REPUBLIQUE DU CAMEROUN

Paix-Travail-Patrie

UNIVERSITE DE YAOUNDE I

******

CENTRE DE RECHERCHE ET FORMATION DOCTORALE EN SCIENCES, TECHNOLOGIE ET GEOSCIENCES

******

UNITE DE RECHERCHE ET FORMATION DOCTORALE SCIENCE DE PHYSIQUES ET APPLICATIONS

******

B.P. Box 812 Yaoundé

Email : crfd_stg@uy1.uninet.cm

REPUBLIC OF CAMEROON

Peace-Work-Fatherland

UNIVERSITE OF YAOUNDE I

******

POSTGRADUATE SCHOOL OF SCIENCE, TECHNOLOGY AND GEOSCIENCE

******

RESEARCH AND POSTGRADUATE TRAINING UNIT IN PHYSICS AND ITS APPLICATIONS

******

P.O. BOX 812 Yaoundé

Email : crfd_stg@uy1.uninet.cm

LABORATOIRE D'ENERGIE, ET DES SYSTÈMES ÉLECTRIQUES ET ÉLECTRONIQUES

LABORATORY OF ENERGY AND ELECTRICAL AND ELECTRONIC SYSTEMS

DEVELOPPEMENT DES CRYPTOSYSTÈMES ROBUSTES A RÉCURRENCE LOGISTIQUE

THÈSE

Présentée en vue de l'obtention du Diplôme de DOCTORAT/ Ph.D de Physique

Option : Systèmes Électriques et Électroniques

Par

NKANDEU KAMDEU Pascal Yannick

Matricule : **99S021**

Master de Physique

Sous la direction de :

Alain TIEDEU

Professeur, Université de Yaoundé I

Année 2019

UNIVERSITÉ DE YAOUNDÉ I
UNIVERSITY OF YAOUNDE I

FACULTÉ DES SCIENCES
FACULTY OF SCIENCE

DÉPARTEMENT DE PHYSIQUE
DEPARTMENT OF PHYSICS

ATTESTATION DE CORRECTION DE LA THÈSE DE DOCTORAT/Ph.D

Nous, Professeur **NDJAKA Jean-Marie Bienvenu** et Professeur **EYEBE FOUDA Jean-Sire Armand**, respectivement Président du jury et Examinateur de la Thèse de Doctorat/Ph.D de Monsieur **NKANDEU KAMDEU Pascal Yannick**, Matricule **99S021**, préparée sous la supervision du Professeur **TIEDEU Alain,** intitulée : « **Développement des cryptosystèmes robustes à récurrence logistique**», soutenue le **Mardi 22 Décembre 2020**, en vue de l'obtention du grade de Docteur/Ph.D en Physique, Spécialité **Systèmes Electriques et Electroniques,** attestons que toutes les corrections demandées par le jury de soutenance ont été effectuées.

En foi de quoi, la présente attestation lui est délivrée pour servir et valoir ce que de droit.

Fait à Yaoundé le .....03 FEB 2021.....

Examinateur

Pr EYEBE FOUDA Jean-Sire Armand

Le Président du jury

Pr. NDJAKA Jean-Marie Bienvenu

Le Chef de Département de Physique

UNIVERSITE DE YAOUNDE I * FACULTE DES SCIENCES * DEPARTEMENT DE PHYSIQUE
Le Chef de Département

Ndjaka Jean-Marie Bienvenu
Professeur

Dédicace

A mon père Kamdeu Pascal.

A ma mère Emakoua Rose.

A mon épouse Mafogang Jeanne.

A toute la grande famille.

Remerciements

Je remercie particulièrement le professeur Alain Tiedeu pour avoir accepté d'encadrer ce travail jusqu'au bout.

Toute ma gratitude au chef de département de physique, le professeur Ndjaka Jean-Marie Bienvenu, pour son leadership bienveillant, son accompagnement, et ses conseils.

Je remercie les enseignants du laboratoire dont le professeur Essimbi Zobo, le professeur Biya Motto, le professeur Bertrand Bodo, et le professeur Jean-Sire Armand Eyebé pour leurs instructions, leurs expertises et leurs encouragements.

Je remercie les ainés de Laboratoire : Professeur Poné, docteur Kenfack Gutenbert, docteur Yannick Abanda, docteur Mbieda Duplex ; et les camarades Mbieda Franck, Yepdia mariel, Makem Mimosette, et bien d'autres, pour avoir profité de leurs expériences ou de leurs interventions.

Un merci particulier au Docteur Tchomgo qui n'a cessé de me pousser vers l'avant avec ses conseils.

Je remercie toute la communauté éducative de Polytechnique, de l'école doctorale, de l'université de Yaoundé I, et tous les professeurs qui ont participé à améliorer la qualité de ce document de quelques institutions qu'ils soient.

Merci à mon père Kamdeu Pascal et ma mère Kamdeu Rose pour avoir fortement impulsé ce travail.

Merci à mon épouse Mafogang Jeanne pour avoir cru que cela était possible.

Merci également aux chefs et aux membres des familles Chito, Matiké, Adjomo, Chutchua, Ndeugoue qui ont fortement contribué à la finalisation de travail.

Je remercie mes amies et connaissances, Ngueng Zacharie, Dr. Mama Keita, Nkemdji Hugh, Ngeunga Alain Brice, Messi Martin, Fansi Jean-claude, Ela Emmanuel, Nana Abel, Tsomberg Bernadin, Nkolo Thomas, Dontsop Flora, Deukeng Gladys, Dongmo Yannick Feudjo Emmanuel, Mfepet Zounedou, pour leur encouragement.

Enfin, un grand merci à tous ceux qui auraient d'une manière ou d'une autre permis la réalisation de cette thèse.

Tables des matières

Résumé

L'utilisation de la cryptographie basée sur les systèmes chaotiques, mieux adaptée aux images numériques, a mis en avant la récurrence logistique comme l'un des atouts dans la conception des algorithmes de chiffrements d'images. Malheureusement, les lacunes liées à la nature intrinsèque de cette dernière ont généré des failles de sécurité dans les algorithmes proposés. Dans ce document, nous nous sommes alors fixés comme objectifs principaux de bâtir des algorithmes utilisant la récurrence logistique sans que ses manquements ne provoquent des failles de sécurité. A cet effet, nous avons proposé deux nouvelles méthodes de chiffrement utilisant pour la première: une technique de fusion de la récurrence logistique à d'autres récurrences similaires (May, Gompertz, Gaussienne), suivie d'une auto-substitution de l'image à crypter, et complétée par une technique conjuguée de masquage et de relocation des pixels, avec pour objectif final de totalement rendre l'image indiscernable. La seconde méthode proposée permet de masquer chaque pixel de l'image, au moyen d'un nombre pseudo-aléatoire extrait d'une séquence (de nombres pseudo-aléatoires) issue de l'itération de la récurrence logistique, dont les paramètres ont été préalablement altérés au moyen des valeurs de pixels de l'image à crypter. L'ensemble des tests standards effectués sur les deux systèmes cryptographiques démontrent leurs robustesses face aux attaques, leurs rapidités en temps d'exécution et leur supériorité devant de bons systèmes cryptographiques de la littérature, offrant l'espoir d'une solution à implémenter en situation réelle.

Mots clés : Récurrence logistique, générateur de nombres pseudo-aléatoires (GNPA), mixage, cryptographie, image.

Abstract

The use of chaos-based cryptography better suited to image data has put forward the logistics map as the best asset in the design of image ciphering algorithms. Unfortunately, deficiencies related to the intrinsic nature of the latter have generated security vulnerabilities in the proposed algorithms. In this document, we have set ourselves as main objectives to build algorithms using the logistic map without its deficiencies tainting security flaws. To this end, we have proposed two new encryption methods using for the first: a technique of mixing of the logistic map with other similar maps (May, Gompertz, Gaussian), followed by a self-substitution of the image to be encrypted, and complemented by a combined technique of masking and relocating the pixels, with the ultimate goal of totally making the image indistinguishable. The second method proposed uses for image pixels masking, a pseudo-random number value extracted from a pseudo-random number sequence obtained of logistic map iteration, which parameters have been previously altered by the mean of pixel values of the original image. The set of standard tests performed on the two cryptographic systems demonstrate their robustness against attacks, their speed in execution time and their superiority in front of good cryptographic systems of literature, offering the hope of a solution to be implemented in real situation.

Key words: Logistic map, Pseudo-random number generators, mixing, cryptography, image.

Liste des abréviations

AES : Advanced Encryption Standard

ADN : Acide DésoxyriboNucléique

ASCII : America Standard Code for Information Interchange

CBC : Cipher Block Chaining

CFB : Cipher FeedBack

DES : Data Encryption Standard

ECB : Electronic Code Book

GAGOS : Gaussian-Gompertz-System

GNPA : Générateur (trice) de Nombres Pseudo-Aléatoires

LOMAS : Logistic-May-System

LOGOS : Logistic-Gompertz-System

LOGAS : Logistic-Gaussian-System

MAC : Message Authentication Code

MAGAS : May-Gaussian-System

MAGOS : May-Gaussian-System

MD5 : Message Digest 5

MDC : Manipulation Detection Code

MIC : Message Integrity Check

NPCR : Number of Pixels Change Rate

OFB : Output FeedBack

PBC : Plainext Block Chaining

PCBC : Propagating Cipher Block Chaining

PFB : Plaintext FeedBack

PGP : Pretty Good Privacy

PIST : Plain Image Substitution Technique

P-Box: Permutation Box

RC4 : Rivest Cipher 4

RGB : Red Green and Blue

RVB : Rouge Vert Bleu

S-Box: Substitution Box

SHA : Secure Hash Algorithm

SSH : Secure SHell

SSL : Secure Socket Layer

UACI : Unified Average Change Intensity

XOR : exclusive or

XSL: eXtended Sparse Linearization

Liste des tableaux

Liste des figures

Introduction générale

La cryptographie permet à travers les systèmes de communication numérique, l'accès et l'interprétation exclusive du contenu d'un message par l'unique destinataire à qui cela était adressé. De nos jours les images photos et vidéos des entreprises (banques, transports aérien, services des eaux et d'électricité, structures privées, sociétés de télécommunication…), des opérations militaires (forces de défenses ou de sécurités...), des examens médicaux (clichés radio, scanner…), des organisations étatiques (présidentielles, diplomatiques, services secrets…) ou des individus (document photos et vidéos,…) transitent sur internet. Un accès facile à ce type de données pourrait causer un désagrément allant de simples dommages dans la vie d'un individu à un désastre d'une ampleur planétaire. Il est donc plus que nécessaire de pourvoir, ou de renouveler les algorithmes de cryptographie nécessaires au bon fonctionnement et au maintien du mode de vie ultra-numérisé de notre siècle.

Quelques méthodes formelles de cryptographie ont été retracées dans l'antiquité en Égypte, et lors de l'invasion de la Gaulle avec Jules César (-65 avant JC). Cependant, l'usage intensif de la cryptographie se fera pendant la première et la deuxième guerre mondiale, et sera déterminante pour la victoire des alliés. Cette forme de cryptographie était appelée cryptographie classique. Après la deuxième guerre mondiale, la cryptographie s'est trouvée une place importante au sein de tous les systèmes analogiques et numériques, suite à l'invention des ordinateurs et des satellites, qui étaient alors les moyens ultimes de communications de ce temps. Cette cryptographie d'après-guerre connue sous le nom de cryptographie moderne à la différence de ceux d'avant l'ère numérique (cryptographie classique), fut encadrée par des lois mathématiques comme celles de Kerckoffs [1] et de Shannon [2]. Elle engendra des méthodes ou algorithmes de cryptographie comme DES, triple DES, AES, etc. Dans les années 90, le développement et l'expansion d'internet, des applications multimédia, des transferts en nombre et volume de données images et vidéos, et surtout de l'ouverture de ces technologies au grand public ont rendu totalement obsolète la cryptographie moderne. En fait, les temps d'opérations des algorithmes de la cryptographie moderne provoquaient la lenteur des systèmes de communications qui les exploitaient [3, 4]. Par ailleurs, ils étaient véritablement incapables de satisfaire la confidentialité des images photos et vidéos du fait de leurs constitutions caractérisées par une forte redondance des bits de données [4, 5, 6]. Pour remédier aux inconvénients et aux limites de la cryptographie moderne, certains

chercheurs ont proposé d'utiliser les équations (ou systèmes) chaotiques à des fins cryptographiques [3, 4, 6]. Les systèmes chaotiques présentent des propriétés telles que : ergodicité, sensibilité aux conditions initiales et aux paramètres de contrôles, comportement aléatoire, imprédictibilité et déterminisme [7]. Ces propriétés sont similaires à celles exigées d'un excellent système cryptographique à savoir : l'ergodicité, la sensibilité de la clé et du message crypté, l'aléa dans le processus de brouillage ou de diffusion, l'imprédictibilité et la réversibilité [8]. Ainsi paru la cryptographie par chaos dont l'atout majeur sembla être celui de pouvoir brouiller efficacement et rapidement les images numériques et vidéos [4, 6, 9, 10].

Question de la recherche et contribution

Bien des auteurs ont proposé des systèmes cryptographiques chaotiques [10-100], mais seuls les systèmes à une dimension (1D) dont la plus fameuse connue sous le nom de récurrence logistique, ont montré une grande facilité d'implémention et un temps rapide d'exécution [10-30]. Il est rapporté que plus de deux cents (200) articles ont été publiés sur l'application de la récurrence logistique en cryptographie entre 1998 et 2014 [11]. Cependant, certains défauts dans ses propriétés chaotiques tels que la périodicité dans la génération des nombres aléatoire, la non-uniformité des fréquences des nombres générés, la faible ergodicité, le faible intervalle des valeurs du paramètre de control, ont eu comme conséquence désastreuse, la conception de systèmes cryptographiques avec d'énormes failles de sécurités [10-53].

Ainsi, l'algorithme de cryptographie proposé par Wang et al. [13] et utilisant l'ergodicité basé sur la récurrence logistique, a été cryptanalysé (décrypté sans clé) par Arroyo et al. [14]. Ils démontrèrent que l'on pouvait faire une estimation exacte de la condition initiale et du paramètre de contrôle utilisés comme clés de l'algorithme de cryptage. De même Pareek et al. [15-19] ont proposé une multitude d'algorithmes de cryptographies efficaces et rapides en temps d'exécution, utilisant la récurrence logistique comme récurrence chaotique de base. Mais, bien des cryptanalyses [20, 21] ont démontré que ces systèmes cryptographiques présentaient d'énormes défauts dont : des clés invalides; des clés équivalentes ; la faible longueur de la clé ; la possibilité de crypter un message et d'utiliser le résultat pour décrypter un autre (chosen plaintext attack); la possibilité d'utiliser un message supposé crypté pour décrypter un autre message (chosen ciphertext attack). Plus tard, l'algorithme de permutation des valeurs de pixels encodées en binaire utilisant les séquences de nombres aléatoires générée par la récurrence logistique et proposé par Ye et al. [22], a été examiné par Li et al. [23] et trouvé complètement faillible. En effet les valeurs issues de la séquence aléatoire de la récurrence

logistique présentaient des suites de valeurs périodiques. Cette faille (de la récurrence logistique) a également permis en 2011 à Li et al. [24] de casser l'algorithme de cryptage d'image de Rao et al. [25], proposé plus tôt cette année-là et utilisant la récurrence logistique comme fonction chaotique principale. Quelques deux années plus tard, Lui et al. [26] ont défait l'algorithme de « cryptage d'image basé sur l'hybridation de la récurrence logistique et de la courbe elliptique » de EL-latif et al. [27]. Pour cela, ils ont exploité le faible espace de clé lié au paramètre de contrôle pour progressivement révéler l'image confuse. A la suite de ces manquements, des auteurs ont proposé de ne pas utiliser la récurrence logistique toute seule, c'est à dire comme unique fonction chaotique dans un cryptosystème [28, 29]. D'autres ont recommandé d'associer la génération de sa séquence des données de la récurrence logistique aux caractéristiques du message original [8,14, 20, 24, 26]. Malgré ces suggestions, l'algorithme de permutation-substitution proposé par Panduranga et al. [30] utilisant les séquences de la récurrence logistique en combinaison avec les fonctions en treillis, fut brisé par Ahmad et al. [31]. Ils démontrèrent que la phase de permutation et de diffusion pour une même clé restaient inchangée pour différentes images. L'algorithme de Wang et al. [32] utilisant le cryptage alterné d'images au moyen de la récurrence logistique proposé en 2014, fut brisé un an plus tard par la technique de « diviser pour conquérir » de Yap et al. [33]. Wang et al. [34] exploitèrent les faiblesses de la récurrence logistique pour casser l'algorithme de Mirzaei et al. [35], qui associaient pourtant trois systèmes chaotiques dont, le système de Lorenz et un hyperchaos en diffusion ; et la récurrence logistique en permutation.

En 2017, le pseudo-générateur de nombre aléatoire basé sur la technique d'amélioration pseudo-aléatoire de la récurrence logistique et proposé pour la cryptographie d'images par Murillo et al. [36], a été faillible à l'attaque de force brute (test exhaustif de toutes les possibilités de clés) en raison du faible espace des clés [37]. Cette même année-là, un autre algorithme de Murillo et al [38] conçu en exploitant les caractéristiques de l'image à crypter pour générer les éléments d'itérations de la récurrence logistique, échoua les tests courants de sécurités effectués par Fan et al.[39].

Il parait dès lors très explicite que l'utilisation efficiente et efficace de la suite logistique, pour la conception des systèmes chaotiques robustes (sans faiblesse) et rapide, demeure une véritable difficulté.

D'où la question : Peut-on bâtir un cryptosystème (algorithme de cryptographie) à base de récurrence logistique qui résiste aux attaques ?

Nous allons proposer dans ce document deux approches pour la solution :

- La première consiste à combiner la récurrence logistique à d'autres récurrences pour améliorer ses propriétés. A cet effet nous avons proposé un algorithme de cryptographie, basée sur une méthode de mixage de la récurrence logistique à des récurrences qui lui sont similaires.
- La deuxième consiste à bâtir un cryptosystème performant en utilisant la récurrence logistique telle qu'elle est.

Dans le premier chapitre, nous ferons un rappel des notions de base en cryptographie, ensuite une revue détaillée de certains crypto-systèmes à base de récurrence logistique sera effectuée. L'on continuera avec une étude approfondie de la récurrence logistique, de ses manquements liés à la cryptographie, et terminerons avec l'annonce des solutions éventuelles aux problèmes.

Dans le chapitre deux, l'on présentera l'organisation générale d'un crypto-système suivi des étapes détaillées des deux cryptosystèmes mis au point. Le chapitre continuera sur le rappel des métriques d'évaluation des crypto-systèmes, et terminera sur le rappel des attaques à effectuer pour tester la robustesse.

Dans le troisième chapitre après la présentation de la base de données d'images, l'évaluation des cryptosystèmes proposés sera réalisée par le biais des métriques en vigueur. Le chapitre s'achèvera sur la réalisation des attaques les plus efficaces permettant d'évaluer la robustesse de nos cryptosystèmes proposés.

Chapitre 1: Revue de la littérature de la cryptographie d'image basée sur la récurrence logistique

Introduction

Dans ce chapitre, notre objectif est de rappeler les notions, principes, algorithmes et protocoles utilisés en cryptographie en général, de mettre en avant quelques cryptosystèmes basés sur la récurrence logistique, et d'étudier cette dernière. Nous allons donc pouvoir analyser la récurrence logistique dans ses défauts et survoler les quelques solutions existantes pour sa parfaite exploitation.

1.1 Notion de base en cryptographie

1.1.1 Terminologie et objectif

a) Terminologie

Comme dans toutes disciplines scientifiques, la cryptographie possède un registre de mots et de termes dédié à son usage. Parmi ces mots on distingue :

- *Cryptologie :* Il s'agit d'une science mathématique comportant deux branches : la cryptographie et la cryptanalyse
- *Cryptographie :* La cryptographie est l'étude des méthodes donnant la possibilité d'envoyer des données de manière confidentielle sur un support donné.
- *Chiffrement :* Le chiffrement consiste à transformer une donnée (texte, message, ...) afin de la rendre incompréhensible par une personne autre que celui qui a créé le message et celui qui en est le destinataire.
- *Déchiffrement* : est l'opération inverse du chiffrement. Il a pour but de récupérer l'information masquée.
- *Texte en clair :* c'est le message clair ou le message original non chiffré, lisible, compréhensible pouvant être interprété par tous. Il peut être de type caractère de texte, photo ou vidéo.
- *Texte chiffré :* Appelé également cryptogramme, le texte chiffré est le résultat de l'application d'un chiffrement à un texte clair. Il est confus, indéchiffrable et incompréhensible.

- *Clef (ou clé) :* Il s'agit du paramètre impliqué et autorisant des opérations de chiffrement et/ou de déchiffrement.
- *Cryptanalyse :* Opposée à la cryptographie, elle a pour but de retrouver le texte clair à partir de textes chiffrés en déterminant les failles de sécurités des algorithmes utilisés. Lorsque ce but est atteint on parle d'un algorithme (cryptosystème) cryptanalysé ou cassé.
- *Cryptosystème :* désigne la description d'un procédé chiffrement/déchiffrement : la méthode, l'algorithme et son utilisation

b) Objectif

De nos jours le rôle de la cryptographie s'étend bien au-delà du chiffrement et du déchiffrement. Elle fournit aussi des services et des procédures de confidentialité, d'intégrité, d'authentification, d'identification, de non-répudiation, et de contrôle d'accès. L'ensemble de ces procédures peuvent être réalisés en une seule opération.

- Confidentialité

  Toutes personnes interceptant un message chiffré doit être incapable de l'interpréter ou de deviner le message original s'il ne dispose pas de la clé de chiffrement

- Intégrité

  Cette procédure permet de garantir au destinataire que le message n'a pas été substitué ou modifié par une tierce personne (malicieuse). Pour réaliser cette procédure on utilise les fonctions de hachage.

- Authentification

  Elle est utilisée pour s'assurer que le message vient du correct émetteur dont l'identité peut être usurpée par une personne malicieuse. On l'implémente grâce à la fonction « message identification code » MAC ou la signature digitale.

- Identification

  Le but de cette procédure est d'authentifier l'interlocuteur et non le message. Le protocole exécutant cette procédure est nommé "challenge-response"

- Non-répudiation

La non-répudiation empêche de nier la participation à un échange ou traitement de données.

- Access Contrôle

Cette procédure prévient l'accès non autorisé a une ressource, elle détermine le type d'accès, le temps d'accès, les restrictions, et les niveaux d'autorisation.

1.1.2 Principes généraux de la cryptographie moderne

a) Principe de base

Pour obtenir un message chiffré *(C)* (également pour « *Ciphertext* »), le chiffrement (*E*) (également pour « *Encryption* ») est effectué en utilisant un opérateur ou une opération ou encore une fonction mathématique sur message numérique en clair (*M* ou *P* « *Plaintext* ») (codé en ASCII) au moyen d'un paramètre qui initialise les opérations et qui représente la clé *(K)* (correspondant à « *Key* »). Toutes les opérations et fonctions utilisées doivent être inversibles mathématiquement et doivent absolument être dépendantes de la clé. Ainsi le récepteur pourra au moyen de la clé, inverser le processus de chiffrement et retrouver le message original. Le message déchiffré sera noté *(D)* (correspondant aussi à « *Decryption* »). Les équations (1) and (2) expriment le principe mathématique.

- *Principe de chiffrement*

$$C = F(M,K), \text{ ou } C = E_K(M), \text{ ou } C = E_K(P) \quad \text{avec} \quad F = E_K \tag{1.1}$$

- *Principe de déchiffrement*

$$M = F^{-1}(C,K), \text{ ou } M = D_K(C) = D_K(E_K(M)), \text{ ou } P = D_K(C) \quad \text{avec} \quad F^{-1} = D_K \tag{1.2}$$

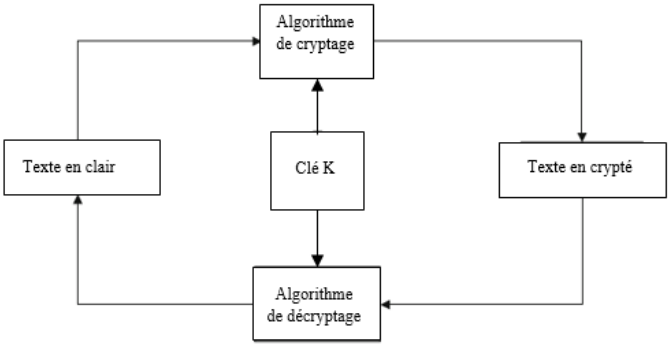


Figure 1.1 *Organigramme d'un processus de cryptage /décryptage*

- *Opérations de bases utilisées en cryptographie*

Etant donné que l'outil principal de la cryptographie moderne c'est l'ordinateur, les opérations arithmétiques et logiques sont utilisées sans restriction pour

concevoir les algorithmes et les systèmes cryptographiques. Parmi ces opérations les plus en vue sont :

- l'addition et la soustraction arithmétique
- la multiplication et la division arithmétique
- le ou exclusif « XOR » connu aussi sur le nom de « bitwise OR » en anglais
- les opérations modulaires (« modulo » ou congruence) associées ou non à l'addition, la soustraction, la multiplication ou la division.
- les opérations logiques : et « AND », ou « OR », non ou « NOR »…
- le décalage par rotation cyclique à droite « bit-shift right » ou à gauche « bit-shift left »

Tableau 1.1 *Symboles des opérations généralement utilisées en cryptographie moderne*

Notation	Opération effectuée
$X+Y$	Addition avec complément de deux mots binaires
$X-Y$	Soustraction avec complément de deux mots binaires
$X \oplus Y$	OU exclusif XOR entre deux mots binaires
$X >>> Y$	Décalage vers la droite du mot binaire X de Y bits par rotation cyclique
$X <<< Y$	Décalage vers la gauche du mot binaire X de Y bits par rotation cyclique
$X \times Y$	Multiplication binaire c'est-à-dire modulo $2^{\text{mot binaire}}$
$X \bmod Y$	Opération modulaire entre X et Y, le résultat est le reste de la division de X par Y

b) Principe de Kerckhoffs

Avant la cryptographie moderne la sécurité des cryptosystèmes conçus reposait entièrement sur le fait que l'algorithme de cryptographie était tenu secret. Au 19 siècle Auguste Kerckhoffs [1] proposa six règles qui devinrent les précurseurs des fondements de la cryptographie moderne :

1- Le système doit être matériellement, sinon, mathématiquement indéchiffrable.

2- Il faut qu'il n'exige pas d'être secret, et qu'il puisse sans inconvénient tomber entre les mains de l'ennemi.

3- La clef doit pouvoir en être communiquée et retenue sans le secours de notes écrites et être changée ou modifiée au gré des correspondants.

4- Il faut qu'il soit applicable à la correspondance télégraphique.

5-Il faut qu'il soit portatif, et que son maniement ou son fonctionnement n'exige pas le concours de plusieurs personnes.

6- Enfin, il est nécessaire, vu les circonstances qui commandent l'application, que le système soit d'un usage facile, ne demandant ni tension d'esprit, ni la connaissance d'une longue série de règle à observer.

Le second et le troisième principe sont la base de la cryptographie moderne, ils imposent que l'algorithme conçu peut être publié à tout vent pour cela il doit dépendre exclusivement de la clé dont le nombre de possibilité (appelé espace de clé) doit être très grand sinon idéalement infini.

c) Théorie de Shannon [2]

Dans les années d'après-guerre, Claude Shannon stipula qu'un bon algorithme de cryptographie devait être une succession d'opération de « confusion » et de « diffusion ». La confusion étant un ensemble d'opération ayant pour but de brouiller les relations visibles entre éléments à crypter (position des caractères dans un message). Et la diffusion ayant pour principale but d'altérer par propagation et de manière non-uniforme les valeurs des caractères d'un message.

Quel que soit le système cryptographique conçu, au moins l'une de ces deux techniques doit être implémentée.

1.1.3 Protocoles et les algorithmes cryptographiques [41]

a) La cryptographie à clé privée

Encore connu sous le nom de chiffrement symétrique ou cryptographie symétrique, elle se caractérise par le fait que la clé utilisée pour le chiffrement est identique à celle utilisée pour le déchiffrement. Ce type de chiffrement est connu pour être très rapide et approprié à la manipulation du chiffrement de large quantité de donnée comme des images vidéo. Cependant, la nécessité de transmettre la clé durant la communication constitue un très grand inconvénient car cette dernière peut être interceptée. Les exemples de cryptosystèmes les plus connus sont : DES, triple DES, AES, RC4, masque jetable…

Il existe deux grandes familles de chiffrements symétriques :

- Les chiffrements par flots (« stream ciphers ») : le chiffrement se fait bit par bit et de manière indépendante pour un message numérique à chiffrer.
- Les chiffrements par Blocs (« Block ciphers ») : le chiffrement se fait sur des bits regroupés en mot binaire de 2, 8, 16, 64 bits ou bien plus, avec ou non une relation de dépendance.

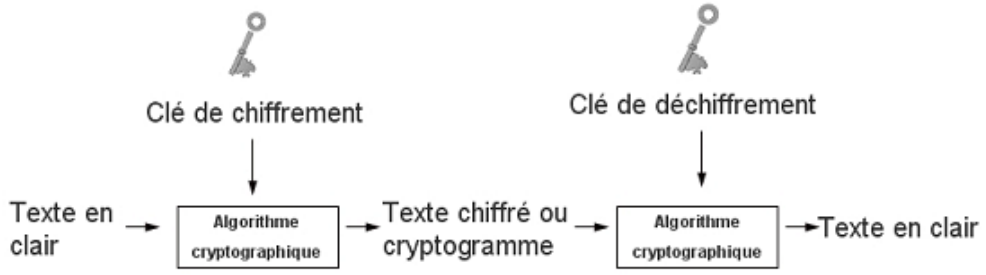


Figure 1.2 *Schéma du protocole d'un algorithme à clé privée (clé identique)*

b) La cryptographie à clé public

Lorsque deux clés différentes sont utilisées dans le processus de chiffrement et de déchiffrement on parle de cryptosystème asymétrique ou de chiffrement asymétrique, ou encore de cryptographie à clé public. Ce principe a été introduit en cryptographie par Diffie et Hellman [78], il utilise une fonction à sens unique (clé public) dont la connaissance préalable de quelques paramètres choisis permet de générer la clé de déchiffrement. Ce protocole permet de palier aux inconvénients de la clé privé mais se trouve être 100 à 1000 plus lent que le chiffrement symétrique. Comme exemple d'algorithme utilisant ce type de chiffrement nous avons le SSL, SSH, PGP.

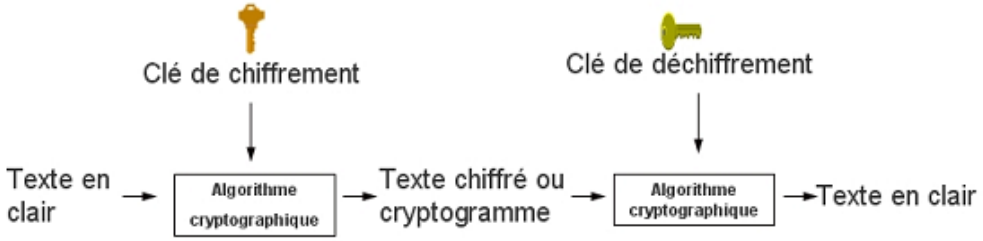


Figure 1.3 *Schéma du protocole d'un algorithme à clé publique [77]*

c) Les algorithmes de chiffrement par bloc

Il existe un grand nombre d'algorithmes de base associé à la cryptographie à clé public selon que l'on effectue un chiffrement par bloc ou par flot. En général le bloc est considéré à partir de 1 octet soit 8 bits.

Les plus utilisés sont :

- *L'« Electronic Code Book » ou mode ECB*

Dans ce mode chaque bloc constitué du message clair (ou texte clair) est chiffré et stocké indépendamment comme dans une base de données. Ainsi un registre ou livre de correspondance électronique ou numérique peut être généré pour le chiffrement. Cette méthode bien que rapide a l'inconvénient d'avoir toujours une paire identique bloc de message en clair/bloc de message chiffré, après chaque chiffrement pour une même clé. Cela provoque des redondances (dans le message chiffré) pouvant être exploitées par un attaquant pour casser l'algorithme.

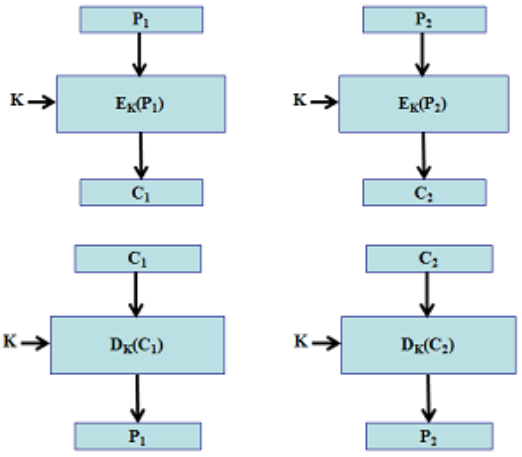


Figure 1.4 *Logigramme du chiffrement/déchiffrement du mode ECB [43].*

- *Le « Cipher block chaining » ou mode CBC*

Dans ce mode, chaque bloc chiffré de la séquence contribue au chiffrement du bloc qui le suit créant ainsi un lien entre blocs chiffrés appelé chainage. Mathématiquement, on effectue une operation XOR entre le bloc de texte en clair suivant P_i et le bloc de texte en clair précédent chiffré C_{i-1} enfin d'obtenir le bloc chiffré suivant C_i, i étant le nombre de bloc du message.

$$\begin{cases} C_i = E_K(P_i \oplus C_{i-1}) \\ P_i = C_{i-1} \oplus D_K(C_i) \end{cases} \tag{1.4}$$

Cette méthode a l'avantage de produire des textes chiffrés différents résultant des textes clairs presque identiques. Cependant elle présente l'inconvénient de propager l'erreur d'un bloc sur tout le reste.

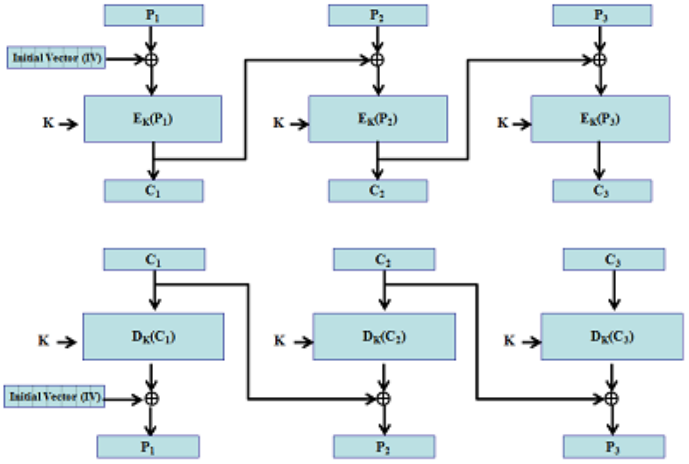


Figure 1.5 *Logigramme du chiffrement/déchiffrement du mode CBC [43].*

- *Le « Propagating Cipher Block Chaining » ou mode PCBC*

  Ce mode est similaire au mode CBC, à la différence que le bloc de texte clair précédent (P_{i-1}) et son correspondant chiffré (C_{i-1}) sont impliqués dans le chiffrement/déchiffrement du suivant.

$$\begin{cases} C_i = E_K(P_i \oplus C_{i-1} \oplus P_{i-1}); \\ P_i = C_{i-1} \oplus P_{i-1} \oplus D_K(C_i) \end{cases} \tag{1.5}$$

  Ce mode amplifie la moindre différence lors du cryptage, ainsi deux textes en clairs ayant des différences seulement au niveau des bits d'un bloc de même rang dans leurs séquences, produirons deux textes chiffrés complètement différents. Cependant, ce mode amplifie et propage d'avantage les erreurs.

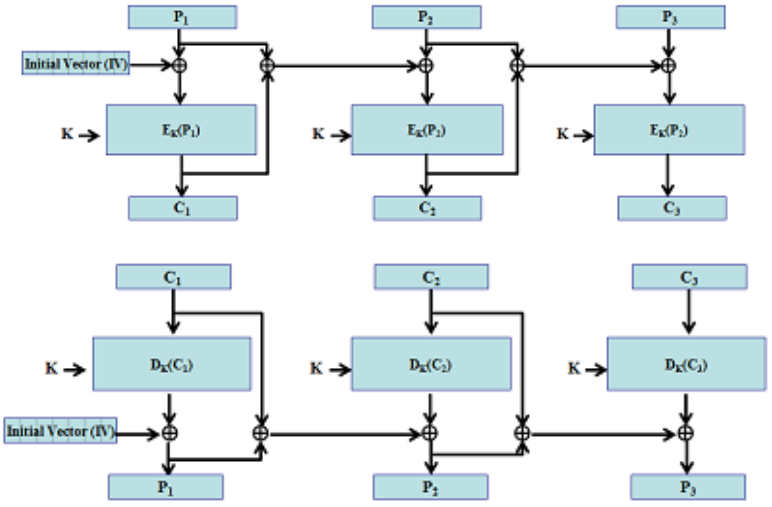


Figure 1.6 *Logigramme du chiffrement/déchiffrement du mode PCBC [43].*

- *Le « Cipher-Feedback » ou mode CFB*

  Le mode CFB permet de travailler sur des blocs considérés d'un ou de plusieurs bits (n-bit CFB) tout en effectuant le chainage. Il dispose aussi de l'avantage de pouvoir transmettre le bit ou le bloc chiffré.

$$\begin{cases} C_i = P_i \oplus E_K(C_{i-1}) \\ P_i = C_i \oplus D_K(C_{i-1}) \end{cases} \tag{1.6}$$

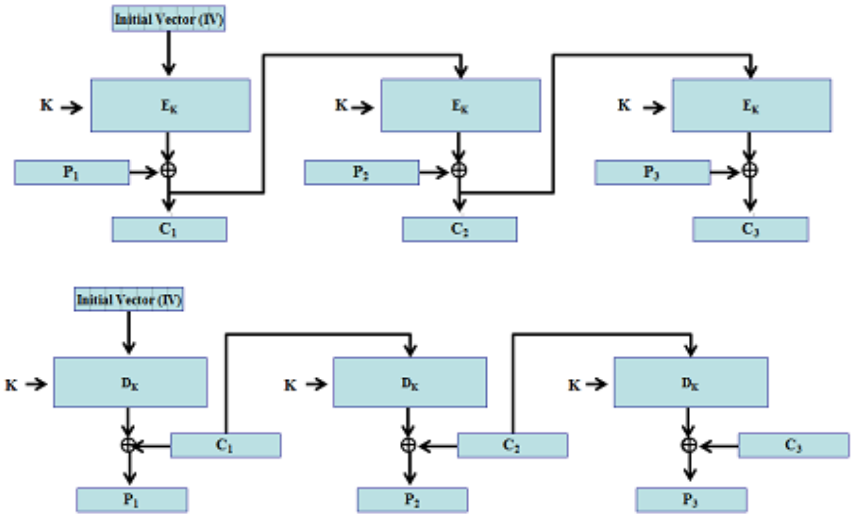


Figure 1.7 *Logigramme du chiffrement/déchiffrement du mode CFB [43].*

- *Les autres modes*

  Bien de modes ont été proposés ou publiés, mais ils ne sont pas particulièrement sollicités. Dans cette catégorie on peut citer : le « plaintext block chaining » ou mode PBC, le « output-feedback » ou mode OFB, le « plaintext feebback » ou mode PFB,…

d) Les algorithmes de chiffrement par flot

Ce type d'algorithme fait le chiffrement bit par bit (contenu dans des blocs ou non) ordonné dans une séquence. Quoi qu'il en soit chaque bit est chiffré en utilisant une sous-clé particulière et indépendante au moyen de l'operation XOR. Il existe trois classes de chiffrement par flot :

- *Le « One-time-pad » (ou chiffrement de Vernon)*

  Dans le principe numérique du chiffrement de Vermon, la clé K produit une séquence $k_0, k_1, k_2, \ldots k_{n-1}$ de sous-clés de longueur n independantes et distribuées de manière uniforme et aléatoire, qu'on « XOR » avec une séquence $m_0, m_1, m_2, \ldots m_{n-1}$, des bits du texte en clair pour obtenir un texte chiffré C de sequence $c_0, c_1, c_2, \ldots c_{n-1}$ et de longueur n. On exprime ce chiffrement / déchiffrement par l'équation suivante :

$$c_i = m_i \oplus k_i \tag{1.7}$$

$$m_i = c_i \oplus k_i \tag{1.8}$$

  Il a été démontré que le « one-time-pad » procure une parfaite sécurité, mais ne peut être implémenté tel quel du fait de la difficulté à trouver une clé de même longueur de bit que le message et remplissant les conditions de disposition aléatoire, de distribution uniforme et d'indépendance.

- *Le chiffrement par flot synchrone (« Synchronous Stream Ciphers »)*

  Ce chiffrement est celui pour lequel la séquence des sous-clés (ou clés dynamiques) est générée indépendamment du texte clair et du texte chiffré, cependant il nécessite que l'émetteur et le récepteur synchronisent leurs générateurs de sous-clé (clés dynamiques) et leurs ordres séquentiels de chiffrement. C'est pourquoi l'algorithme de chiffrement par flot est dit synchrone. Ils ont l'avantage de ne pas propager les erreurs, mais sont vulnérables aux attaques actives comme l'insertion, la suppression et la copie de digits du texte chiffré (par un adversaire actif). En fait ces attaques produisent une perte de synchronisation.

  Le processus de cryptage d'un chiffrement par flot synchrone (Figure 1.8) est décrit par l'équation (1.9) où $f(S_i, K)$ est la fonction qui détermine l'état suivant S_{i+1}, $g(S_i, K)$ est la fonction génératrice de la clef dynamique et $h(z_i, m_i)$ la fonction de chiffrement.

$$\begin{aligned} S_{i+1} &= f(S_i, K), \\ z_i &= g(S_i, K), \\ c_i &= h(z_i, m_i), \end{aligned} \tag{1.9}$$

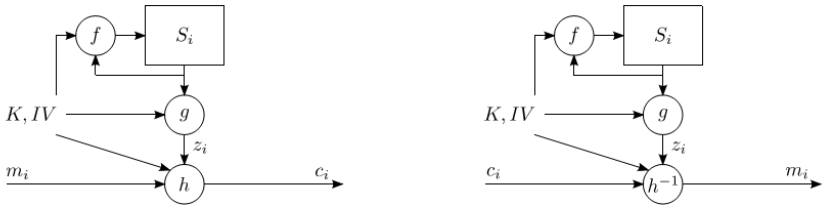


Figure 1.8 *Schéma de l'algorithme de chiffrement/déchiffrement synchrone par flot: à droite processus de chiffrement, à gauche, celui de déchiffrement [44]*

- *Le chiffrement par flot asynchrone ou auto-synchrone (« Asynchronous Stream Ciphers »)*

  Pour l'algorithme de chiffrement asynchrone chaque bit de clé dynamique généré est une fonction d'un nombre déterminé par l'ensemble des bits des textes chiffrés précédents. Etant donné que l'état interne du générateur de clé dynamique du déchiffrement dépend entièrement des n bits du texte chiffré précédemment, il va automatiquement se synchroniser avec le générateur de clé dynamique du chiffrement après avoir reçu les n bits du texte chiffré. Ce type de chiffrement bien que robuste aux attaques se trouve être très sensible aux erreurs de propagation qui ont pour effet de modifier la clé dynamique et le déchiffrement.

$$\begin{aligned} \beta &= (c_{i-t}, c_{i-t+1}, ..., c_{i-1}), \\ z_i &= g(\beta_i, K), \\ c_i &= h(z_i, m_i), \end{aligned} \tag{1.10}$$

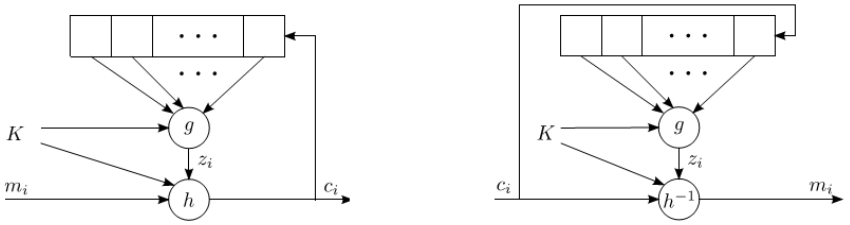


Figure 1.9 *Schéma de l'algorithme de chiffrement (à droite)/déchiffrement (à gauche) par flot auto-synchrone [44].*

e) Les algorithmes de hachage

Les fonctions de hachage sont aussi connues sous bien d'autres noms : les fonctions de compression, les fonctions de contraction, le « message digest », l'empreinte électronique, le « checksum » de chiffrement, le MIC « message integrity check », le MDC « manipulation detection code ». Une fonction de hachage est une fonction

mathématique ou autre, qui prend en entrée une chaine de caractère alphanumérique de dimension variable et la transforme en une chaine caractère alphanumérique de dimension fixe mais plus petite appelée valeur de hachage « hash value ». Le processus de hachage doit être irréversible et extrêmement sensible au moindre changement de bit d'une séquence du message original enfin produire une valeur de hachage complètement différente. La fonction de hachage permet de vérifier l'authenticité ou l'intégrité d'un message en comparant les valeurs de hachages de celui de l'émetteur avec celui du récepteur. Les algorithmes de hachage les plus connus sont MAC (« message authentication code), MD5 (« message digest 5 »), SHA (« secure hash algorithm »).

1.1.4 Cryptanalyse [45]

Il s'agit de l'étude des mécanismes théoriques ou techniques visant à briser (casser) un algorithme de déchiffrement. C'est le fait de retrouver le message M à partir du cryptogramme C, sans connaître la clé K a priori. Dans certains cas, il s'agira également de retrouver cette clé K.

a) Cryptanalyse usuelle

La cryptanalyse traditionnelle associe l'application d'outils mathématiques, à la recherche de motifs, et à la résolution analytique. La patience, la détermination et la chance peuvent être parmi les ingrédients de réussite d'un cryptanalyste. Les cryptanalystes sont également appelés des pirates ou hackers.

Les techniques de cryptanalyse peuvent se résumer en cinq niveaux d'attaques liés aux données utilisées:

– L'attaque par force brute (Brute-force attack) : Le cryptanalyste teste toutes les combinaisons de clés possibles jusqu'à l'acquisition du texte clair.

– L'attaque sur texte chiffré seul (Ciphertext-only attack) : Le cryptanalyste ne connaît que le message chiffré par l'algorithme et essaye de déduire la clé ou le texte clair. Le manque d'information rend cette cryptanalyse plus délicate.

– L'attaque à texte clair connu (Known-plaintext attack) : Le cryptanalyste possède le texte ou des parties du texte en clair et leurs correspondants chiffrés.

– L'attaque à texte clair choisi (Chosen-plaintext attack) : Le cryptanalyste peut choisir le texte en clair, et il peut produire la version chiffrée de ce texte (il a accès à la machine à crypter) avec l'algorithme considéré dès lors comme une

boîte noire. Les techniques de cryptage asymétrique sont notamment sensibles à ce type d'attaque.

– L'attaque à texte chiffré choisi (Chosen-ciphertext attack) : Le cryptanalyste possède le texte chiffré et peut obtenir le texte en clair associé.

Pour vérifier la sécurité d'un cryptosystème nouvellement conçu, quelques tests de cryptanalyses doivent être réalisés. Les algorithmes de cryptanalyse précédemment cités ont été conçus et développés pour évaluer la robustesse et les caractéristiques des algorithmes de cryptage proposés.

b) Cryptanalyse différentielle [46]

Il s'agit de l'étude (modélisation) des transformations subies par le message durant son passage dans l'algorithme de chiffrement. Le principe est de modéliser ce qu'une modification en entrée induira sur le résultat de l'algorithme

c) Cryptanalyse linéaire [47]

Le but est d'effectuer une approximation linéaire de l'algorithme de chiffrement. Il n'y a ici aucune possibilité de choisir le texte clair à chiffrer, on dispose tout au plus d'un ensemble de couples message clair, et équivalent chiffré. Son principe tire profit des probabilités élevées des occurrences des expressions linéaires déduites du texte clair et du texte chiffré. Ces expressions linéaires sont construites à partir d'approximation linéaire de l'algorithme à crypter.

d) Cryptanalyse algébrique [47]

La plupart des algorithmes de cryptage modernes sont conçus de sorte qu'ils résistent aux attaques classiques (linéaire et différentielle). Courtois et Pieprzyk ont étudié la sécurité de ces algorithmes en évoquant une autre hypothèse : le système peut être écrit sous forme d'équations algébriques. Ils ont résolu que l'AES contient 23 équations quadratiques linéairement indépendantes qui peuvent être résolues à l'aide de leur nouveau algorithme de cryptanalyse "XSL" [47]. XSL est une méthode de cryptanalyse pour les chiffrements par blocs. Cette attaque s'avère plus rapide que la recherche exhaustive, mais sa mise en pratique est encore sujet de controverse vue la puissance de calcul qu'elle nécessite.

1.2 Revue de la littérature des cryptosystèmes d'images à base de récurrence logistique

1.2.1 Eléments sur l'image numérique

a) Notion d'image numérique

Les images numériques sont des matrices de nombres représentées comme des tableaux orientés par deux vecteurs dont l'un est « vertical » (colonne), l'autre « horizontal »(ligne). Le couple de valeur de ces deux vecteurs (ligne-colonne) détermine la position d'une troisième valeur comprise en général entre 0 et 255 appelée niveau de gris. L'ensemble des deux valeurs de position et d'un niveau gris est communément appelé pixel. Pour une image noir et blanc le pixel affiche un niveau de gris allant de 0 pour la couleur noir à 255 pour la couleur blanche, les valeurs intermédiaires étant des intensités graduelles de couleur grise. L'image numérique couleur s'obtient en superposant trois « tableau » de valeurs de pixels différents, chaque tableau codant respectivement du premier au dernier pour le rouge, le vert et le bleu (RVB en français et RGB en anglais). Cette superposition permet au moyen de la synthèse additive des couleurs de percevoir d'autres nuances de couleur selon les combinaisons des valeurs des niveaux de couleur des pixels codant pour le rouge, le vert et le bleu.

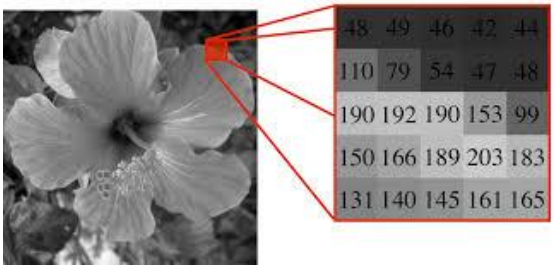


Figure 1.10 *Location des valeurs de niveau de gris dans une image à niveau de gris.*

b) Caractéristique de l'image numérique

Parmi les caractéristiques de l'image numérique on distingue principalement : la définition, la taille, la résolution, le format.

- La définition de l'image est le nombre de pixels qui la compose, elle est donnée en indiquant le produit du nombre de pixels sur une ligne par celui du nombre de pixel sur une colonne (Exemple : 256×256). Il arrive que ce produit soit multiplié par trois pour les images couleurs (Exemple : 512×512×3).

- La taille d'une image s'obtient en multipliant la définition de l'image par le nombre d'octet par pixel.
- La résolution d'une image est le nombre de pixels (ou point lumineux) par pouce (1pouce=2.4cm) considéré sur la diagonale du « tableau » de l'image. L'unité de cette mesure est le *ppi* (« pixels per inch » en anglais) ou *dpi* (dot per inch). Plus la résolution d'une image est forte plus elle est apte à reproduire des détails fins.

c) Format de l'image

Le format de l'image détermine la manière dont elle a été encodée. Dans cet ordre, l'encodage se fait à deux niveaux : sur la valeur numérique d'intensité lumineuse accordée aux pixels (exemple 8bits, 16bits ou 32bits), et sur la compression ou non de l'image. Ainsi on distingue les formats sans compression tel le format TIF (« Tag Image Format »), PSD (PhotoShop Document), XCF (eXeperimental Computing Facility) ; les formats à compression non-destructif tel le format GIF (« Graphic Interchange Format »), PNG (« Portable Network Graphic ») ; et les formats à compression destructif tel le format JPG (déclinaison de JPEG « Joint Photographic Expert Group »). On rappelle que la compression a pour objectif de rendre les fichiers images plus petits en réduisant le nombre de couleur, la résolution ou la taille de l'image. En outre les images ont la particularité d'être constituées de données binaires fortement redondantes et volumineuses du fait de leur taille. Cette dernière caractéristique fait des images numériques un type de donnée particulièrement difficile à crypter.

1.2.2 Algorithme de chiffrement d'image de Wang et al. [48]

a) Description de l'algorithme

Wang et al. [48] Proposèrent un algorithme de chiffrement d'image couleur utilisant uniquement la récurrence logistique comme récurrence génératrice de nombres pseudo-aléatoires (GNPA). Il consistait à décomposer l'image en trois matrices en R,V et B, qui étaient ensuite assemblées en une nouvelle matrice en les juxtaposant dans la direction verticale. La matrice obtenue était brouillée en transposant ses colonnes au moyen d'une séquence de nombre pseudo-aléatoire obtenue de la récurrence logistique par itération. Ce processus de brouillage était repris dans la direction horizontale avec une nouvelle séquence pseudo-aléatoire après que la matrice brouillée précédemment soit recomposée en R, V et B. Pour modifier les valeurs de pixels, Wang généra deux séquences de nombres dont la première était constituée des nombres comprises entre 0 et 2, et la seconde entre

0 à 255, toutes aléatoirement distribuées et conçues à partir d'une même séquence pseudo-aléatoire dérivant de la récurrence logistique. Ensuite il utilisa la valeur rencontrée en parcourant la première séquence pour décider l'ordre de chiffrement des lignes de même rang des trois matrices R, V et B. Le chiffrement se fait par addition modulo 256 d'une valeur du pixel concerné augmenté d'un nombre aléatoire de la deuxième séquence, et ceux du couple pixel précédemment et son correspondant chiffré.

b) Performance du chiffrement d'image de Wang

Le chiffrement de Wang est de bien loin plus rapide qu'un chiffrement similaire utilisant une récurrence 2D ou supérieur. Il est aussi facile à implémenter, cependant il est sans effet sur les images binaire, et peut être victime d'une attaque en texte clair choisi, car les permutations effectuées sont sans effets sur une image constituée de zéros, et la diffusion effectuée révèlerai les sous-clés. De plus il est inapproprié aux images de type gris (noir et blanc).

1.2.3 Algorithme de chiffrement de Sui et al. [49]

a) Description de l'algorithme

Sui proposa un chiffrement monocanal d'images couleurs basées sur l'algorithme de recouvrement de phase et utilisant deux récurrences logistiques couplées. Il utilisa les trois composantes de l'image couleur pour constituer une seule image en niveau de gris dont les pixels furent permutés en exploitant une séquence pseudo-aléatoire du couple de récurrence logistique. Ensuite, il décomposa à nouveau l'image brouillée en trois nouvelles images et encoda chacune sous forme de fonction de phase (POF « phase only fonction ») dans le domaine de Fourier fractionnel (d'ordre α et/ou β) grâce à un algorithme de recouvrement de phase basé sur la transformé fractionnel itérative de Fourier (FrFT). Les phaseurs obtenues des trois images furent combinés en amplitude et en phase, ensuite la matrice de l'amplitude de phase obtenue fut chiffrée au moyen des nombres pseudo-aléatoire des récurrences logistiques, dans une opération d'addition modulo 256 impliquant aussi les valeurs claires et chiffrées précédemment. Dans le processus final, la phase finale (ϕ) obtenu servit de clé, et le phaseur chiffré fut transformé en image par inversion des fonctions (figure 1.11).

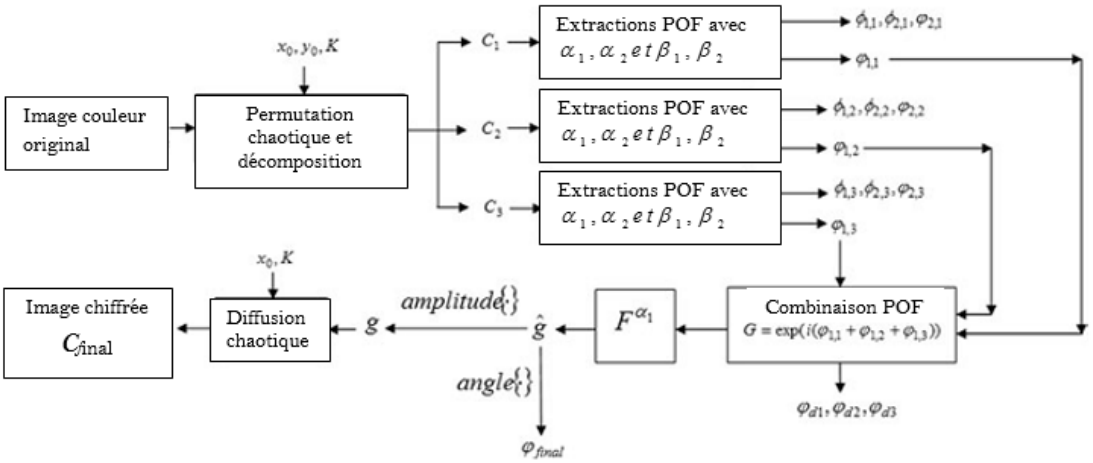


Figure 1.11 *Schéma du chiffrement de Sui et al. [49]*

b) Analyse de sécurité de l'algorithme de Sui

La complexité de l'algorithme de sui et al. [49] le rend certainement invulnérable. De plus, les ordres α et β des Transformées fractionnelles de Fourier et les phases des phaseurs des images intermédiaires (ϕ_1 ϕ_2 ϕ_3 ϕ_{final}) constituent des clés de décryptage. Une attaque exhaustive des clés serait extrêmement confuse. Cependant le processus de chiffrement consomme trop de ressource mémoire (virtuel) et temps.

1.2.4 Algorithme de chiffrement d'image de Murillo-Escobar et al. [50]

a) Description de l'algorithme

Murillo-Escobar utilisa les séquences pseudo-aléatoires de la récurrence logistique combinées à l'ensemble des valeurs de pixels de l'image en claire pour générer un nombre Z empreinte de ce dernier. Ensuite il permuta les valeurs de colonne et de ligne séparément au moyen de deux autres séquences issues de la même récurrence. Enfin il effectua une addition modulo 1 entre les valeurs de pixels permutés et celles d'une séquence aléatoire dont les valeurs étaient augmentées de Z modulo 1. En ramenant les valeurs de pixels dans l'intervalle 0-255 il obtient une image cryptée (figure 1.12).

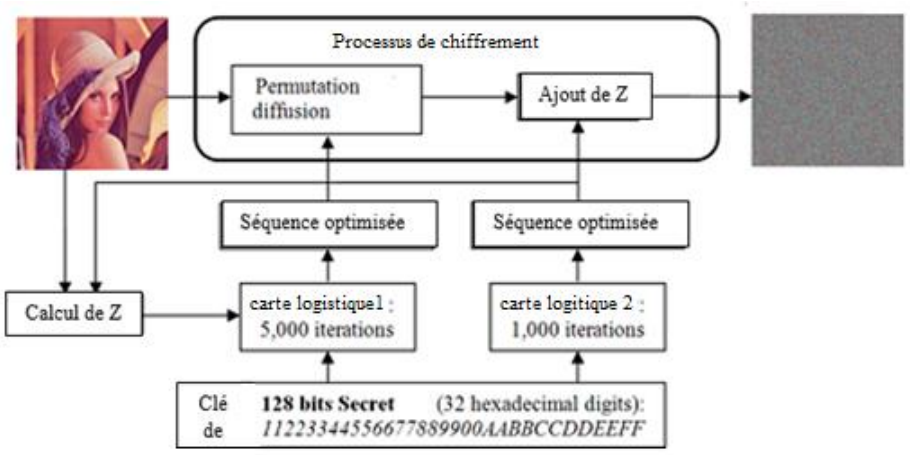


Figure 1.12 *Schéma de chiffrement de Murillo-Escobar [50]*

b) Evaluation de l'algorithme de Murillo-Escobar

Bien que proposant un algorithme robuste et ultrarapide, produisant une image chiffrée caractéristique de l'image claire, Murillo-Escobar était contraint d'insérer l'empreinte Z dans l'image chiffrée pour exploitation lors du déchiffrement. Cette lacune permit à Fan et al.[39] de casser l'algorithme proposé en identifiant Z et en menant les attaques classiques de texte clair et texte chiffré choisi.

1.2.5 Algorithme de chiffrement d'image de Seyedzadeh et al.[51]

a) Description de l'algorithme

L'algorithme de chiffrement de Seyedzadeh utilise les nombres pseudo-aléatoires obtenus d'une récurrence aléatoire quantique, pour altérer les valeurs de pixels d'une image couleur, au moyen d'une opération complexe de modulo 256 et de XOR dans un mode de type CBC. Ensuite il utilise les séquences de nombres aléatoires d'une récurrence logistique 2D pour effectuer à nouveau une diffusion sur les valeurs de pixels complétée par une transposition. En dernier ressort, il effectue une rotation circulaire des bits de pixels sans tenir compte de leur appartenance aux composantes R, V et B (figure 1.13).

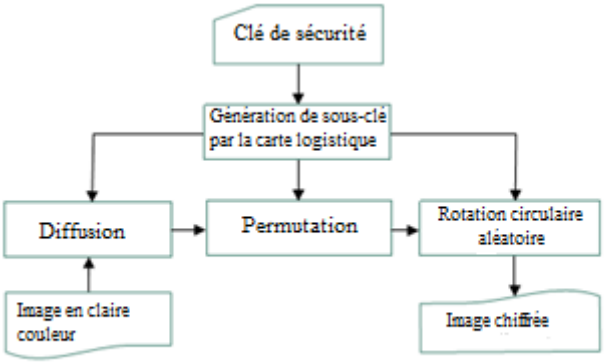


Figure 1.13 *Schéma du chiffrement de Seyedzadeh et al.[51]*

b) Analyse des performances

La complexité du cryptosystème de Seyedzadeh est bonne et son temps d'exécution est pratique. Seulement, son espace de clé de l'ordre 10^{38} est à la limite de ce qui est requis (10^{30}).

1.2.6 Algorithme de chiffrement de Belazi et al. [52]

a) Description de l'algorithme

Pour concevoir un algorithme de chiffrement incassable, Belazi a combiné deux phases de diffusion avec une phase de substitution et une de permutation. Pour cela, il conçut une nouvelle récurrence 1D à partir de la récurrence logistique et l'utilisa en parallèle avec ce dernier dans toutes les phases. La première phase est une diffusion par masquage des valeurs de pixels en utilisant le XOR et les valeurs d'une séquence aléatoire issue de la nouvelle récurrence. La deuxième phase est une substitution des valeurs de pixels utilisant des boites de substitutions (SB) de taille 8×8 créées grâce au deux récurrences. La troisième, une autre phase de diffusion par masquage (XOR) avec des nombres aléatoires issus de la récurrence logistique. Enfin la dernière, une permutation contrôlée par une fonction de nombre premier, de l'image pré-chiffrée transformée en block de matrice de taille carrée aux dimensions identiques (figure 1.14).

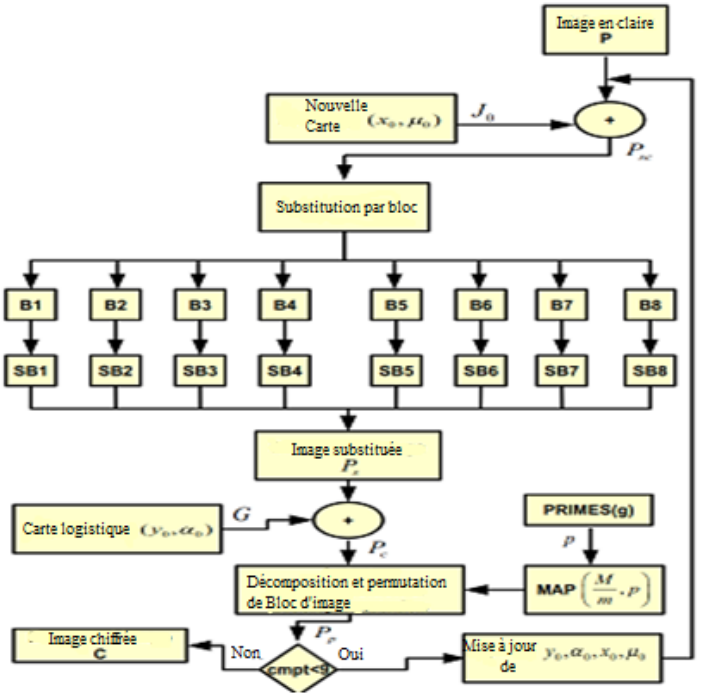


Figure 1.14 *Schéma du chiffrement de Belazi et al. [52]*

b) Analyse des performances

L'algorithme de Belazi et al. [52] est véritablement robuste, mais alors extrêmement lent à l'exécution compte tenu du nombre procédure à compiler. Par ailleurs, il ne peut être utilisé pour le chiffrement des images dont la matrice n'est pas carrée c'est-à-dire ayant une définition m ×n avec m ≠n. Ces manquements constituent un très grand handicap.

1.2.7 Algorithme de chiffrement d'image de Rostami et al.[53]

a) Description de l'algorithme

Rostami a conçu un algorithme dans lequel il exploite les valeurs de pixels de l'image en clair, pour élaborer les paramètres de la récurrence à utiliser comme générateur des nombres aléatoires. Ces nombres aléatoires regroupés dans des matrices 16×16 à la taille totale de l'image et en quatre groupes, sont pour les deux premiers utilisés comme masque par opération XOR suivie d'une transposition de pixels. Et les deux derniers sont utilisés pour répéter le processus de masquage par XOR après la transposition. Il exploita aussi le parallélisme des tâches profitant d'un environnement multiprocesseur, et du fait que le chiffrement s'exécutait par bloc indépendant de pixels au sein d'une même image.

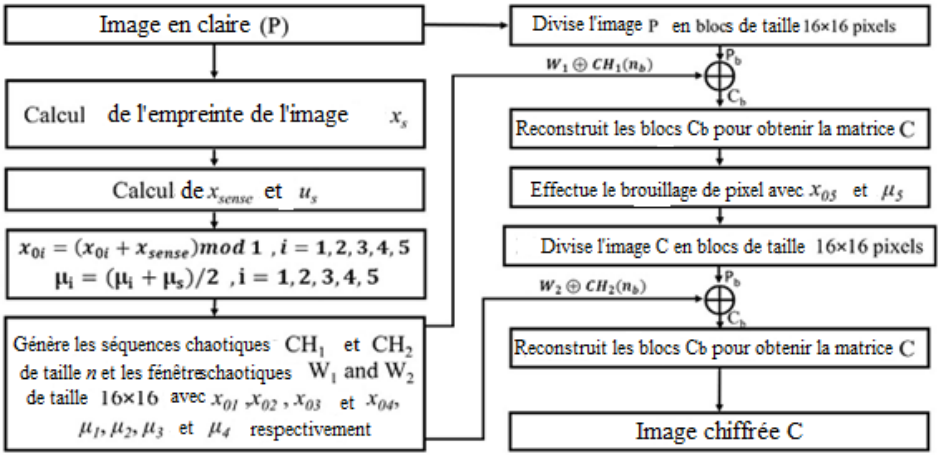


Figure 1.15 *Schéma du chiffrement de Rostami et al. [53]*

b) Analyse des performances de l'algorithme de Rostami

L'espace de clé est très grand 10^{152} (contre 10^{30} au minimum), et l'algorithme est robuste du fait que la clé dépend entièrement de la donnée à chiffrer. Le défaut principal de son cryptosystème réside dans le fait qu'il est conçu pour être exécuté uniquement en environnement multiprocesseurs, parce qu'il serait lent à l'exécution dans un environnement monoprocesseur.

1.3 Récurrence logistique et ses défauts

1.3.1 Origine mathématique de la Récurrence logistique [60-71]

L'équation de la logistique est une modélisation de la croissance démographique d'une population ayant des ressources limitées. Elle a été publiée pour la première fois par le mathématicien et sociologiste Pierre-francois Verhulst [60, 61] et exprimée sous forme différentielle comme suit :

$$\frac{dN}{dt} = aN(1-\frac{N}{K}) \tag{1.12}$$

Ici a est le paramètre de Malthus (taux de croissance maximum de la population), K est la capacité porteuse, N est l'effectif de la population.

En divisant les deux membres de l'équation (1.12) par K et en définissant $z = N/K$ on obtient l'équation différentielle $\frac{dz}{dt} = az(1-z)$, et en considérant la méthode d'Euler pour une équation différentielle, elle devient :

$$\frac{dz}{dt} \approx \frac{z_{n+1} - z_n}{h} \tag{1.13}$$

Avec, $z_{n+1} = z_n + ha(1 - z_n) = z_n(1 + ha - haz_n)$

Prenons $x_n = ahz_n/(1+ah)$, et $r = 1+ah$, alors on obtient :

$$x_{n+1} = r \cdot x_n \cdot (1 - x_n) \tag{1.14}$$

Cette équation aux différences est appelée récurrence (ou suite) logistique, c'est la forme discrète du modèle logistique publié pour la première fois par Robert May [64-67]. $x_n \in [0,1]$, et $r \in [0,1]$

1.3.2 Analyse numérique et graphique de la récurrence logistique

a) Diagramme de bifurcation de la récurrence logistique

Le diagramme de bifurcation de la récurrence logistique résume l'ensemble des comportements dynamiques lorsque r croit. Pour tracer ce diagramme, l'équation (1.14) subit une itération ayant pour condition initiale x_0=0.5 sur l'ensemble des valeurs de r. Les orbites sont enregistrées séquentiellement dans l'ordre croissant des valeurs de r et tracées dans un graphisme (figure 1.16).

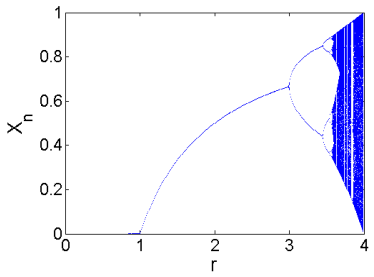


Figure 1.16 *: Diagramme de bifurcation de la récurrence logistique*

Une analyse détaillée de ce diagramme nous mènent aux conclusions suivantes :

(1) Pour $r \in [0, 3]$, la suite logistique est asymptotiquement stable;

(2) $r \in [3, 1+\sqrt{6}]$, la suite logistique devient instable et des bifurcations périodiques d'ordre 2 apparaissent et deviennent asymptotiquement stable;

(3) $r \in [1+\sqrt{6}, r_4]$, $r_4 \approx 3.54409$: les orbites d'ordres 2 deviennent instables et les orbites d'ordres 4 apparaissent et deviennent asymptotiquement stables. Pour r=3.569946, ce

processus de dédoublement de période progressive va aboutir à une cascade de dédoublement de période : c'est le chemin vers le chaos.

(4) $r \in [r_6, r_7]$, $r_6 \approx 3.569946$ et $r_7 = 3.85$, Le système est alterné entre comportement périodique et pure comportement chaotique.

(5) $r \in [r_7, 4]$, la suite logistique a un comportement purement chaotique.

b) Portait de phase ou « cobweb plot »

La récurrence logistique est un système unimodal, son portrait de phase est impossible à tracer, cependant pour visualiser ces orbites pour une valeur donnée du paramètre de contrôle, on utilise la technique de la « toile d'araignée » ou ''cobweb plot'' (en anglais). Elle consiste à prendre comme condition initiale de la prochaine itération d'ordre n, la dernière valeur issue de la précédente itération du même ordre. La figure (1.17) nous montre quelques exemples de tracés en toile d'araignée pour différentes valeurs de r (2 ; 3.2 ; 3.5 ; 3.8). Elle nous montre la densité des trajectoires relative à la dynamique du système.

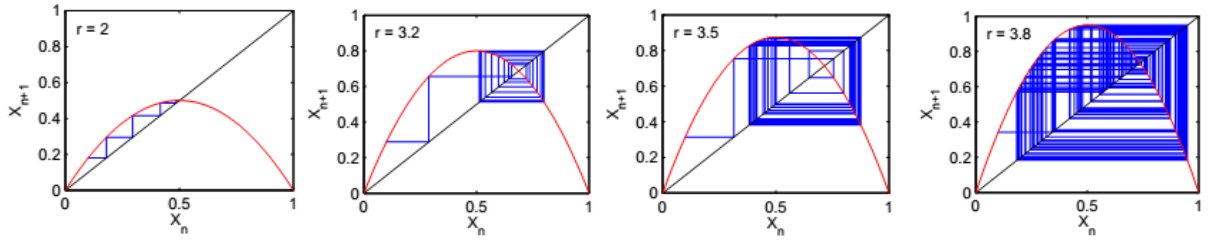


Figure 1.17 : *Graphe de l'orbite de la récurrence chaotique pour différentes valeurs du paramètre de contrôle r (2 ; 3.2 ; 3.5 ; 3.8).*

c) Sensibilité à la condition initiale

Aussi connu sous le nom d' « effet papillon », on peut l'observer en faisant le graphe variable vs itération de deux conditions initiales très proches (figure 1.18-(a)). Cependant, cette méthode n'est pas très adéquate pour mettre en exergue l'effet papillon.

d) Exposants de Lyapunov de la récurrence logistique

Plus rigoureusement pour un espace de phase défini sur IR^m, il existe m exposants de Lyapunov et les propriétés aléatoires apparaissent lorsqu'au moins l'une de ses exposants est positive [54-60].

Considérons la fonction discrète GNPA 1D f, et soit x_0 une condition initiale. On définit l'exposant de Lyapunov pour la trajectoire débutant en x_0 par :

$$\mu = \lim_{n\to\infty}\left[\frac{1}{n}\sum_{i=0}^{n-1}\ln\left|f'(x_i)\right|\right] \tag{1.15}$$

Le graphe donnant les exposants de Lyapunov de la récurrence logistique (figure1.18-(b)) montre clairement les ensembles de valeurs des exposants en dessous de zéro et au dessus pour toutes les valeurs de r. Ainsi on peut identifier sans ambiguité les intervalles de r sensibles aux conditions initiales ou non (ou la récurrence est chaotique ou non)

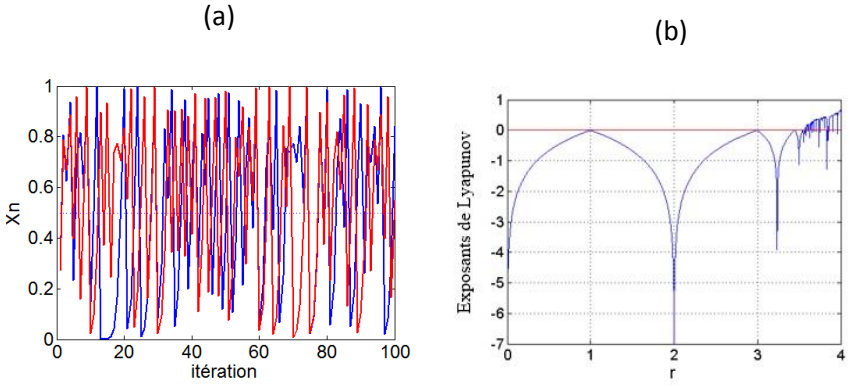


Figure 1.18 : (a) Graphe d'itération de deux conditions initiales très proches ; (b) Graphe des exposants de Lyapunov.

1.3.3 Analyses des défauts de la récurrence logistique et conséquence en cryptographie

La récurrence logistique malgré ses atouts de simplicité d'analyse, d'exploitation ou d'implémentation présente des défauts qui la rendent vulnérable en cryptographie. Parmi ces défauts on peut citer la distribution non uniforme des données, les intervalles de périodicité dans la zone chaotique, les problèmes de synchronisation des conditions initiales, la faible largeur de l'intervalle chaotique.

a) Probabilité de distribution non uniforme

La distribution des variables pseudo-aléatoires issues des itérations est non uniforme quelques soit la valeur du paramètre de contrôle. La fréquence d'apparition des nombres (histogramme) dans toutes les séquences est déséquilibrée donnant avantage aux chiffres ou nombres des extrémités (figure 1.19-(a)). Ce manquement exprime la faible ergodicité de la récurrence logistique.

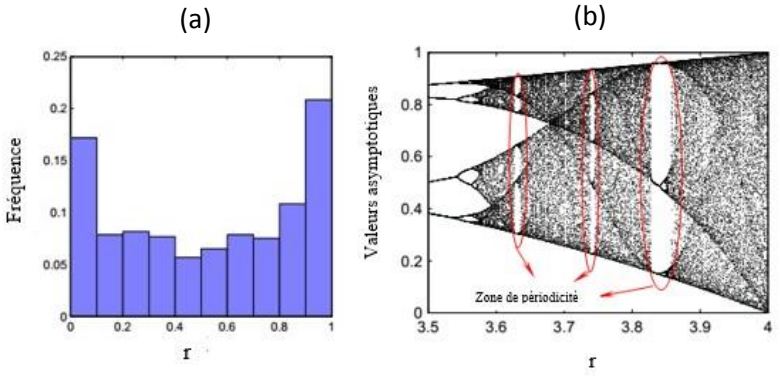


Figure 1.19 *: (a) Graphe de fréquence de distribution de la récurrence logistique, (b) Diagramme de bifurcation avec identification des zones de périodicité de la récurrence logistique.*

b) Fenêtre ou zone de périodicité

Dans l'intervalle [3.56, 4] du paramètre de contrôle ou le chaos apparait et s'établit, il y a des séquences de valeurs périodiques qui atténuent le caractère aléatoire de la récurrence (Figure 1.19-(b)).

c) Synchronisation des conditions initiales

La synchronisation des conditions initiales s'observe en général lorsque deux valeurs très proches de condition initiale sont numérisées avec une faible précision (1 ou 2bit), ou lorsque la somme de ces variables donne 1. Exemple du deuxième cas : x_{01}=0.3 et x_{02}=0.7 auront la même séquence de nombre pseudo-aléatoire pour un même paramètre de contrôle$\left(r \cdot x_{01} \cdot (1-x_{01}) = r \cdot x_{02} \cdot (1-x_{02})\right)$.

d) Largeur de l'intervalle chaotique

La récurrence logistique se défini sur l'intervalle [0, 4] du paramètre de contrôle, mais ne commence à manifester un comportement chaotique que sur l'intervalle [3.569, 4], soit à peine une largeur de 0,431. Pire en encore cette largeur contient des intervalles de périodicité.

1.3.5 Autres récurrences 1D

Il existe bon nombre de récurrence à 1D ayant des propriétés presque identique à celui de la récurrence logistique. Parmi les plus en vue on peut citer : La récurrence sine, la récurrence de Gompertz la récurrence de May, la récurrence de Bernoulli…

Cependant le besoin de proposer une récurrence 1D fiable en tout point en cryptographie, a conduit à la recherche des solutions pouvant permettre d'annihiler les défauts de la récurrence logistique.

1.4 Quelques solutions d'améliorations de la récurrence logistique en cryptographie

Quelques chercheurs en cryptographie ont mis au point des stratagèmes pour contourner les faiblesses de la récurrence logistique avant ou pendant son usage, on cite : le confinement à l'intervalle restreint [3.99, 4] des paramètres de contrôles aux bonnes propriétés aléatoire ; l'extension de l'intervalle du paramètre de contrôle ; l'amélioration de la récurrence en la transmutant en une autre récurrence, l'association de la récurrence à d'autres par couplage, la création de nouvelles récurrences 1D aux excellentes propriétés, en utilisant la récurrence logistique comme l'un des parents, dans un processus de combinaison mathématique ou de mixage.

1.4.1 Confinement du paramètre de contrôle de la récurrence logistique dans l'intervalle [3.99, 4]

Presque tous les auteurs [10-53] qui se servent de la récurrence logistique optent pour son utilisation dans l'intervalle [3.99, 4]. Hors de cet intervalle, il est pratiquement impossible de se fier aux séquences obtenues car leur aléa est transitoire et sujet à tous sortes de défauts (périodicité, non-uniformité…).

1.4.2 Extension de l'intervalle du paramètre de contrôle

Il est possible d'étendre l'intervalle chaotique [3.99, 4] à un intervalle [0, 4] du paramètre de contrôle de la récurrence logistique, mais son équation sera modifiée comme suit :

$$(0.0025\times r+3.99)\cdot x_n\cdot\left(1-x_n\right) \tag{1.16}$$

Avec $x \in [0,1]$, $r \in [0,4]$.

Cette équation semble résoudre le problème du faible espace de clé de la récurrence logistique, mais elle n'améliore aucun de ses défauts (périodicité, ergodicité, synchronisation…).

1.4.3 Transmutation de la récurrence logistique

C'est une technique qui consiste à modifier l'équation de la logistique en la modélant de façon à accentuer sa non-linéarité. Gao et al. [70] ont dans ce principe modifié la récurrence logistique, et ont proposé une nouvelle récurrence baptisée non linear chaotic map (NCA).

$$x_{n+1} = r \cdot tg(\alpha x_n) \cdot (1 - x_n)^{\beta} \tag{1.17}$$

Avec $x_n \in [0,1]$, et le paramètre de contrôle $r \in [0,4]$; $\alpha \in [0,1.4]$; $\beta \in [5,43]$

1.4.4 Associations des récurrences par couplage

Supposons que nous ayons une récurrence quelconque 1D pouvant s'écrire sous la forme $x_{n+1} = f(r, x_n)$, r étant son paramètre de contrôle. Pour coupler deux récurrences 1D identiques ou différentes $x_{n+1}^{(1)} = f(r_1, x_n^{(1)})$ et $x_{n+1}^{(2)} = f(r_2, x_n^{(2)})$, on les associe selon les équations suivantes :

$$\begin{cases} x_{n+1}^{(1)} = (1-\varepsilon) f\left(r_1, x_n^{(1)}\right) + \varepsilon f\left(r_2, x_n^{(2)}\right) \\ x_{n+1}^{(2)} = (1-\varepsilon) f\left(r_2, x_n^{(2)}\right) + \varepsilon f\left(r_1, x_n^{(1)}\right) \end{cases} \tag{1.18}$$

ε représente le coefficient de couplage, il est toujours inférieur à 1.

Les auteurs [71, 72, 108] ont exploité ce principe pour utiliser des récurrences GNPA couplées dans leurs algorithmes.

1.4.5 Créations de nouvelle récurrence par mixage

Il existe plusieurs méthodes [73, 74, 109] permettant de mixer plusieurs récurrences pour en créer une nouvelle. Pour les cas d'espace Pak et al. [74] ont utilisé l'équation (1.19) pour la création de leur nouvelle récurrence, alors que Zhou et al. [73], l'équation (1.20).

$$x_{n+1} = f(r, x_n) \times 2^k - floor(f(r, x_n) \times 2^k) \tag{1.19}$$

Avec « floor (x) » étant une fonction qui donne la partie entière de x, $8 \le k \le 20$

$$x_{n+1} = \left(f(r_1, x_n) + g(r_2, x_n) \right) \bmod 1 \tag{1.20}$$

Les fonctions f et g sont des récurrences à une dimension ayant respectivement r_1 et r_2 comme paramètre de contrôle.

Pour la suite, nous allons exploiter la méthode de Zhou et al.[73], car elle fait intervenir plusieurs récurrences aux propriétés similaires offrant un large éventail de possibilité sur la conception de GNPA robustes.

Conclusion

Nous avons étudié dans ce chapitre les grands principes théoriques de la confusion et de la diffusion utilisés en cryptographie. La revue de la littérature faite dans le cadre de la cryptographie basé sur la récurrence logistique, nous a bien démontré que les méthodes et les algorithmes sont encore loin de satisfaire les exigences. Par ailleurs, l'étude de la récurrence logistique nous a montré qu'il est un outil appréciable de cryptographie d'image du fait de son implémentation facile. Seulement, ces lacunes à haut risque telles les fenêtres de périodicités et les distributions non uniformes observées, nous imposent la recherche de solutions à son amélioration ou l'adoption de celles existantes.

Chapitre 2: Méthodologie de chiffrement et élaboration de deux algorithmes basés sur la récurrence logistique

Introduction

Ce chapitre nous initie dans un premier temps aux méthodes cryptographiques les plus utilisées. Nous y présentons également et de manière détaillée, nos méthodes développées pour solutionner le problème d'algorithmes de chiffrements faillibles, basés sur la récurrence logistique. Nous rappelons en fin de ce chapitre, l'ensemble des notions liées aux métriques les plus utilisées pour évaluer les systèmes cryptographiques.

2.1 Organisation générale des cryptosystèmes d'images utilisant les récurrences GNPA

La conception d'un algorithme de chiffrement d'images par récurrence GNPA s'organise autour de trois principaux points dont : le choix de la ou des récurrence (s) à utiliser ; la génération et la gestion de la clé de chiffrement ; les méthodes de confusion et de diffusion à combiner.

2.1.1 Choix du Générateur de nombre pseudo-aléatoire à utiliser

Le choix du GNPA est déterminant pour la qualité de l'algorithme de chiffrement à implémenter. En effet, la vitesse d'exécution de l'algorithme, l'espace de clé, le désordre dans les séquences aléatoires sont déterminés par le type de système aléatoire utilisé comme GNPA. A défaut des récurrences existantes, le cryptographe peut améliorer, combiner, coupler ou créer de nouvelles récurrences GNPA.

a) La vitesse d'exécution

Si le système est 1D comme la récurrence logistique ou 2D comme la récurrence standard, il est plus facile de les utiliser, car ce sont des récurrences aux équations bien connues (Equations 2.5 et 2.9), et pour le cas d'espèce ils sont très faciles à mettre en œuvre. Les systèmes dont les dimensions excèdent 2D sont plus difficiles à numériser, la méthode Euler n'est plus suffisante, on commence à utiliser les méthodes plus complexes comme Runge-Kutta, ordre 2, 3 ou 4. De plus les systèmes 1D et 2D donnent plus vite

les valeurs aléatoires que les ordres supérieurs plus lent en itération. Alvarez et al. [75] ont ainsi critiqué l'algorithme de Gao et al. [12] utilisant les systèmes GNPA de Chen et de Lorenz. Ils ont démontré que l'algorithme repris tout entier avec la récurrence logistique, dans le même environnement et pour la même image, passait de 5.8 secondes à 1.2 seconde de durée de chiffrement.

b) L'espace des clés

L'espace de clé est le nombre total de paramètre du système aléatoire utilisé pour chiffrer. Il est plus grand avec les systèmes chaotiques d'ordres supérieurs à 2D car, plus l'ordre augmente, plus le nombre de paramètre du système est grand. Ainsi donc pour deux systèmes de dimensions différentes utilisés un même nombre de fois dans un algorithme de chiffrement, celui qui est de dimension plus grande aura bien plus de paramètres à mettre en jeu autant de fois qu'il est sollicité.

c) Les séquences aléatoires

Tous les récurrences GNPA ne sont pas parfaites, c'est pourquoi il important de choisir une récurrence ayant au moins un sous-interval dans lequel ses propriétés aléatoires se rapproche favorablement de celui d'un système chaotique parfait (c'est-à-dire produisant de véritables séquences de nombres aléatoires). Dans le cas contraire, l'algorithme conçu sera fragilisé par les intervalles de périodicités du système.

2.1.2 Génération et gestion de la clé de chiffrement d'image

L'utilisation du chaos en cryptographie a engendré un nouveau protocole de génération et de gestion de clé celui de l'utilisation d'une clé interne ou d'une clé externe pour la conception d'un algorithme de chiffrement.

a) Utilisation d'une clé interne

Le concept de clé interne est fondamentalement simple, il réside dans le fait que pour le cryptosystème à concevoir, les paramètres du GPNA choisi à savoir : le ou les paramètre(s) de contrôle(s), et le ou les condition(s) initiale(s), seront utilisés comme les clés de chiffrement/déchiffrement. Ce protocole est très pratique mais il comporte des risques de sécurité. En effet, il sera plus facile d'identifier le GNPA utilisé et d'isoler les paramètres défaillants de ce système, afin de lancer une attaque pour deviner les clés. Des algorithmes de chiffrement d'images comme ceux des auteurs en [10-15] ont été

victimes de ce genre d'attaque, et ont été cassés par cryptanalyse. Si par l'exemple l'on voulait utiliser la récurrence logistique d'équation $x_{n+1} = rx_n(1-x_n)$, sa condition initiale x_0 et son paramètre de contrôle r seront alors utilisés comme clés de l'algorithme, et seront comptés autant de fois comme clés, autant de fois qu'ils seront sollicités.

b) Utilisation d'une clé externe

La clé externe pallie aux défauts de la clé interne, elle donne peut d'information sur le système de clé utilisé (par conséquent sur le type de GNPA choisi) et de ce fait augmente la sécurité de l'algorithme. Malgré cet avantage, la gestion des clés internes est plus difficile car il faut ajouter à l'algorithme de chiffrement, un algorithme de génération de clé expansive ou compressive. On choisit une clé binaire de taille défini, et on la transforme par des méthodes mathématiques pour en extraire les paramètres du GNPA utilisé. Si la clé est trop grande on la hache, si elle est trop petite elle subit une expansion. A titre d'exemple, Liu et al. [76] ont intégré à leur algorithme de chiffrement l'algorithme SHA-256 (« Secure Hash Algorithm » en anglais) pour transformer leur clé K ($K=K_1, K_2, \ldots K_{32}$) de 256 bits en paramètres de systèmes GNPA. Leurs équations étaient comme suit :

$$x_0 = x_0^{'} + \frac{(K_1 \oplus K_2 \oplus K_3 \oplus K_4 \oplus K_5 \oplus K_6 \oplus K_7 \oplus K_8 \oplus K_9)}{256} \tag{2.1}$$

$$y_0 = y_0^{'} + \frac{(K_9 \oplus K_{10} \oplus K_{11} \oplus K_{12} \oplus K_{13} \oplus K_{14} \oplus K_{15} \oplus K_{16})}{256} \tag{2.2}$$

$$z_0 = z_0^{'} + \frac{(K_{17} \oplus K_{18} \oplus K_{19} \oplus K_{20} \oplus K_{21} \oplus K_{22} \oplus K_{23} \oplus K_{24})}{256} \tag{2.3}$$

$$c_0 = c_0^{'} + \frac{(K_{25} \oplus K_{26} \oplus K_{27} \oplus K_{28} \oplus K_{29} \oplus K_{30} \oplus K_{31} \oplus K_{32})}{70} \tag{2.4}$$

2.1.3 Choix des méthodes de confusion et de diffusion

Le chiffrement d'image se fait sur la base du principe conjugué de diffusion et de confusion. En effet la plupart des cryptosystèmes combinent sous plusieurs couches (tours ou «round ») d'exécutions répétées de chiffrement (figure 2.1) : une méthode de type permutation (confusion), suivie d'une méthode de type masquage par XOR (diffusion) dans un mode choisi (CBC, PCBC…), utilisant des nombres speudo-aléatoires fournis par des récurrences chaotiques. Ce schéma de confusion-diffusion adopté de la cryptographie moderne, et proposé pour la première fois par Fridich [4] pour le

chiffrement d'image par chaos, avait pour but de progressivement briser la redondance des valeurs de pixels enfin de créer une image chiffrée.

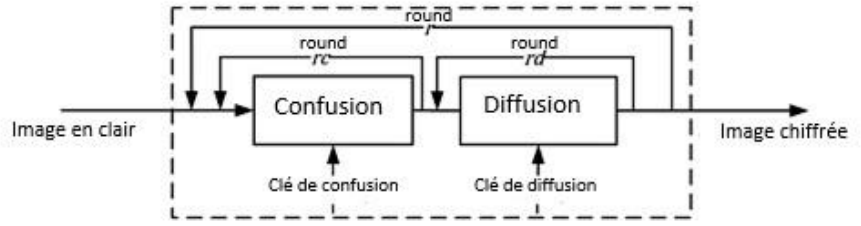


Figure 2.1 *Schéma de chiffrement de Fridich* [4]

Bien que ce schéma ait été adopté par la plupart des cryptographes, il a principalement eu comme inconvénient d'être couteux en temps d'exécution. Un grand nombre de chercheurs se sont lancés dans la recherche de nouvelles méthodes de diffusion et de confusion, leur permettant de chiffrer avec un niveau de sécurité acceptable en une seule couche (un seul tour ou « round »). On catégorise alors des méthodes anciennes et nouvelles utilisées dans le chiffrement d'images par chaos parmi lesquelles les plus utilisées sont : la permutation, le brouillage, les P-boxes (ou boites de permutation), la substitution, les S-boxes (ou boite de substitution), l'encodage ADN (acide désoxyribonucléique) ou « DNA encoding » en anglais, la permutation par niveau de bit (« bit-level permutation »), le couplage masquage-brouillage (« scrambling-masking »).

a) La permutation

La permutation est une méthode de confusion, elle a pour but de rendre une image confuse en relocalisant les pixels de l'image (comme un puzzle) sans altérer leurs valeurs. On l'exécute en général en utilisant des systèmes chaotiques 2D qui préservent l'espace (c'est-à dire bijective et inversible) tel la récurrence standard (Equation 2.5), la récurrence de chat (Equation 2.6), la récurrence du boulanger (Equation 2.7). Le niveau de sécurité d'une permutation est très faible, mais combiné à une méthode de diffusion, le cryptosystème conçu devient alors très robuste [79-84].

$$\begin{cases} x_{i+1} = (x_i + y_i) \bmod N \\ y_{i+1} = (y_i + K \sin \dfrac{2\pi x_{i+1}}{N}) \bmod N \end{cases} \tag{2.5}$$

$$\begin{bmatrix} x_{i+1} \\ y_{i+1} \end{bmatrix} = \begin{pmatrix} 1 & p \\ q & pq+1 \end{pmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} (\bmod N) \tag{2.6}$$

$$\begin{cases} x_{i+1} = \dfrac{N}{n_j}\left(x_i - N_j\right) + y_i \bmod \dfrac{N}{n_j} \\ y_{i+1} = \dfrac{k_j}{N}\left(y_i - y_i \bmod \dfrac{N}{n_j}\right) + N_j \end{cases}, \text{ with} \begin{cases} n_0 + n_1 + ... + n_t = N \\ N_j = n_0 + n_1 + ... + n_j \\ 0 \le j \le N \\ N_j \le x_i \le N_j + n_{j+1} \\ 0 \le j \le t-1 \\ n_0 = 0 \end{cases} \tag{2.7}$$

b) Le brouillage

Le brouillage est une méthode de confusion similaire à la permutation, toutefois, il peut se faire avec n'importe quel générateur de nombre aléatoire, de système chaotique ou de séquence de nombres aléatoires. Il se fait par transfert du désordre existant dans les valeurs des nombres pseudo-aléatoires d'une séquence, aux positions des pixels d'une image. Son niveau de sécurité est identique en tout point à celui d'une permutation, mais il est plus utilisé que ce dernier.

c) Les « P-boxes »

Cette méthode de type confusion de même niveau qu'une permutation est réalisée par une table précompilée de correspondance entre valeur de position et valeur de position aléatoire (figure 2.2 a). Elle a l'avantage de permettre de gagner en temps lors d'une opération de permutation (car déjà préconçu) et se met automatiquement à jour après utilisation grâce aux générateurs de nombres aléatoires (ou aux systèmes chaotiques) qui lui sont affectés.

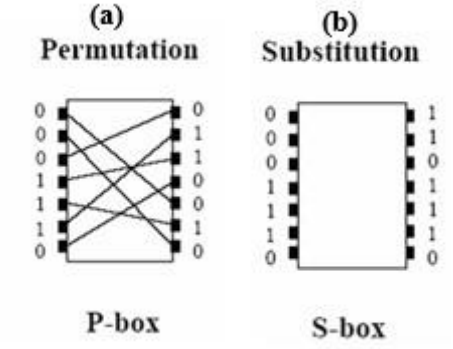


Figure 2.2 : *(a) boîte de permutation, (b) boîte de substitution.*

d) La substitution

Cette méthode peut être de type confusion ou diffusion : lorsqu'on fait un décalage à droite ou à gauche de valeurs de pixels sur leurs positions par rotation circulaire (figure 2.3), alors la substitution est une permutation car les valeurs de pixels ne sont

pas altérées. Mais si on remplace une valeur de pixel par celui d'un nombre aléatoire (cas S-boxes), ou si on effectue une rotation circulaire entre les bits d'une valeur de pixel (figure 2.2 b), alors la substitution devient une diffusion. Son niveau de sécurité dépend donc du type d'opération effectuée.

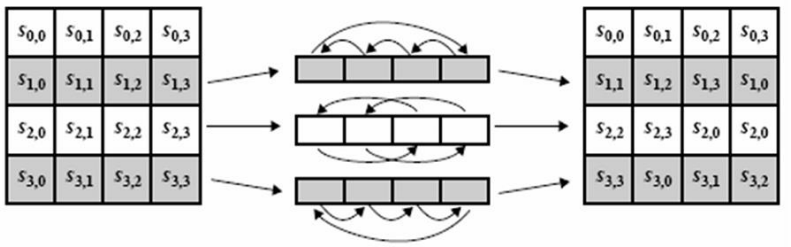


Figure 2.3 *Illustration d'une rotation circulaire entre les pixels d'une image.*

e) Les « S-boxes »

Ce sont des tableaux permettant de substituer une valeur de pixel au vue d'un certains nombres de critères liés à sa codification numérique. Les « S-boxes » sont des méthodes de diffusion avec un haut niveau de sécurité, mais provoquent d'énormes lenteurs dans l'exécution des algorithmes de chiffrement. Par exemple, pour une valeur de pixel codé sur 8 bits $P = b_1 b_2 b_3 b_4 b_5 b_6 b_7 b_8$, l'opération de substitution de ce pixel à travers une boîte S-box peut se faire en considérant les valeurs $b_5 b_3 b_8 b_1$ et $b_2 b_4 b_6 b_7$ (les b_i étant reclassés aléatoirement), pour la ligne et pour la colonne respective de la boite de substitution (figure 2.4).

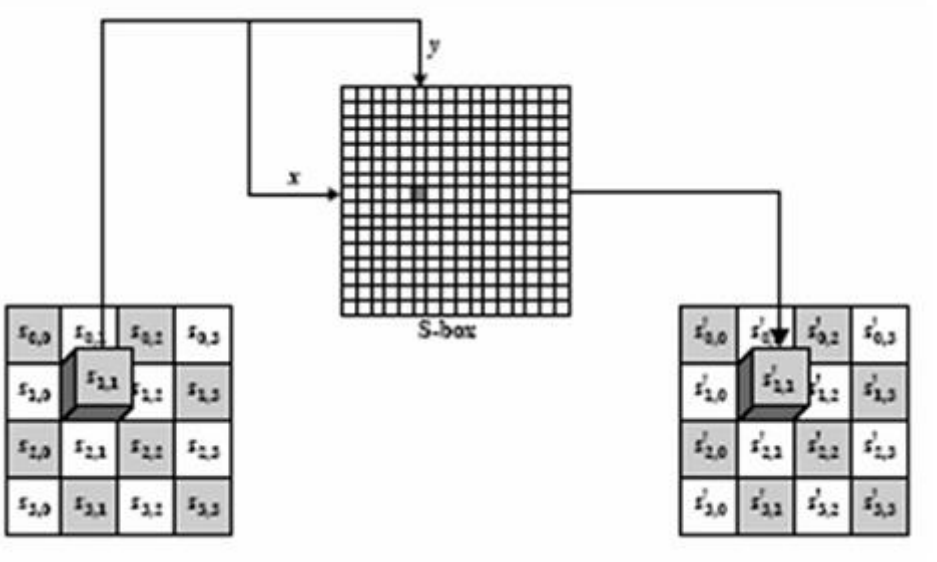


Figure 2.4 *Illustration de l'utilisation d'une boite S-box*

f) L'encodage ADN

Comme nous le savons, ils existent quatre bases nucléiques A, C, G et T dans la séquence de l'ADN tel que, A est le complément de T, et C est le complément de G. Deux bits d'une séquence binaire 00, 01, 10, et 11 sont utilisés pour les représenter. Etant donné que le bit 0 est le complément du bit 1, la séquence binaire 00 est traitée comme le complément de 11 et 01 comme celui de 10. Ainsi donc, on peut encoder 00, 01, 10 et 11 en utilisant respectivement les bases de l'ADN C, A, T et G. Avec cet encodage, si le niveau de gris d'une image est 135, sa valeur binaire 10000111 peut être encodée comme TCAG, et la valeur de pixel peut être restaurée par décodage en remplaçant les bases de l'ADN par leurs séquences binaires. Une fois l'encodage effectué, on peut réaliser l'addition, la réplication ou une nouvelle règle de complémentarité (Tableau 2.1). L'encodage ADN est une méthode de type diffusion avec un niveau de sécurité très élevé mais un temps d'exécution extrêmement médiocre.

Tableau 2.1: *table d'addition d'une séquence ADN proposé par Jain et al. [85]*

+	A	C	T	G
A	C	A	G	T
C	A	C	T	G
T	G	T	C	A
G	T	G	A	C

g) La permutation par niveau de bit

C'est une méthode de type confusion et/ou diffusion selon la manière d'exploiter l'information contenue dans un pixel. Dans une valeur de pixel codée entre 0 et 255 c'est-à-dire sur 8 bits (ou 1 octet), un bit peut être représentatif d'une certaines quantités d'informations selon sa position dans l'octet. Le tableau 2.1 établit l'importance d'un bit en terme de quantité d'information contenu dans l'octet en fonction de sa position. Les quatre premiers bits (du $8^{\text{ième}}$ au $5^{\text{ième}}$) déterminent 94.15% de la valeur du pixel alors que le reste ($4^{\text{ième}}$ au 1^{er}) détermine 6% seulement. L'algorithme « bit-level permutation » sépare les bits des valeurs pixels en ces deux groupes, et les chiffres séparément en confusion ou en diffusion avec les méthodes classiques. Le niveau de sécurité d'un algorithme de « bit-level permutation » bien conçu est très élevé, et son temps d'exécution est acceptable.

Tableau 2.2 *Contribution en pourcentage d'information par différent bit dans un pixel.*

Position de bit *(i+1)* dans un pixel	Pourcentage *p(i)* de l'information du pixel
1	0.3922
2	0.7843
3	1.5686
4	3.137
5	6.275
6	12.55
7	25.10
8	50.20

h) Le masquage-brouillage

La plupart des cryptosystèmes d'images par récurrence GPNA exécutent séquentiellement une confusion suivie d'une diffusion (ou l'inverse) répété ou non un certain nombre de fois (figure 2.1). En 2014 Eyebe et al. [86, 87] proposèrent d'exécuter les deux en même temps, ainsi paru le « scrambling-masking ». Cette méthode consiste à exercer simultanément sur un pixel une diffusion (masquage) et une relocalisation (brouillage). Elle est donc du type fusion « diffusion-confusion » (figure 2.5), et permet de gagner énormément en temps de chiffrement avec un bon niveau de sécurité.

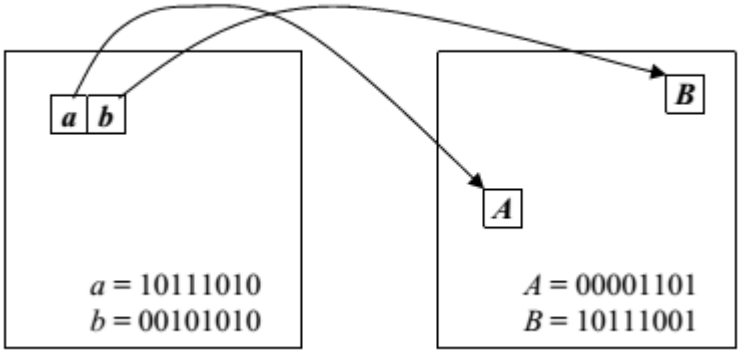


Figure 2.5 *Illustration d'un processus de masquage brouillage entre deux pixels directement voisins.*

2.2 Première contribution au chiffrement d'image : « un algorithme de chiffrement d'image basé sur la technique de substitution et le mixage des récurrences GNPA » [88].

Dans cette section, nous présentons un algorithme de chiffrement qui utilise les architectures S-boxes et de masquage-brouillage, et qui exploite pour cela de nouvelles récurrences conçues à par partir des récurrences logistique, May, Gompertz, gaussienne.

2.2.1 Présentation des récurrences GNPA choisies

Nous avons choisi de créer nos récurrences 1D par mixages des récurrences : logistique, May, Gompertz et gaussienne, avec pour objectif de profiter de leurs avantages et d'optimiser leur comportement chaotique en corrigeant leurs défauts.

a) Rappel des équations des récurrences 1D génératrices

- *La récurrence logistique*

La récurrence logistique (Equation 2.8) avec ses atouts de simplicité et de facilité d'implémentation, sera exploitée comme récurrence génératrice, en dépit du fait que ces propriétés ne soient pas idéales.

$$x_{n+1} = r x_n \left(1 - x_n\right) \tag{2.8}$$

x_n est la valeur obtenue après n itération dans l'intervalle [0, 1], r est le paramètre de contrôle dont les valeurs appartiennent à l'intervalle [0, 4]

- *La récurrence de May [89]*

Robert May a proposé en 1976 la modélisation de la croissance démographique d'une population continuellement décimée par une maladie. L'équation aux différences de ce modèle s'écrit :

$$x_{n+1} = x_n \exp\left(a\left(1 - x_n\right)\right) \tag{2.9}$$

Avec $x_n \in \left[0,\ 10.9\right]$ et le paramètre de contrôle a appartient à l'intervalle [0, 5].

Le diagramme de bifurcation de la récurrence de May (figure 2.6-a) montre une évolution aléatoire similaire dans les grandes lignes à celui de la logistique. En effet, à la figure 2.6-a, on peut observer le problème récurrent de la récurrence logistique à savoir la distribution non-uniforme des données et les zones de périodicités. Par

ailleurs, son tracé des exposants de Lyapunov (figure 2.6-b) révèle des valeurs faibles et positives dans un intervalle du paramètre de contrôle restreint à [2.5, 5].

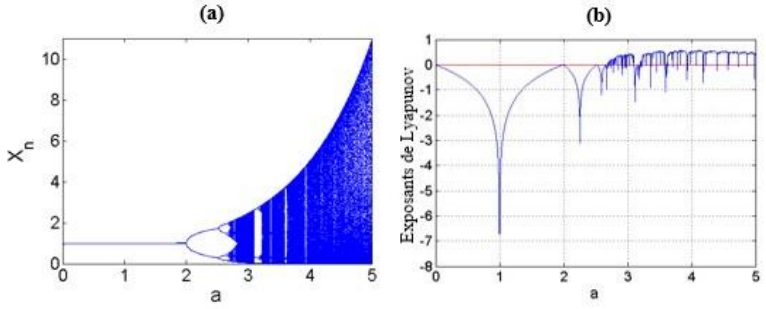


Figure 2.6 *: (a) Diagramme de bifurcation de la récurrence de May ; (b) graphe des exposants de Lyapunov de May.*

- *La récurrence de Gompertz [90]*

L'équation de la récurrence de Gompertz à utiliser est la suivante :

$$x_{n+1} = -bx_n \ln x_n \tag{2.10}$$

$b \in [0, e\ ]$, e =2.71829…est le paramètre de contrôle. Et la variable $x_n \in [0,\ 1]$.

La figure 2.7 montre que ses propriétés chaotiques ont des lacunes également caractérisées par les fenêtres de périodicité, une distribution non-uniforme (figure 2.7-a) et un très faible intervalle aux propriétés purement aléatoires (figure 2.7-b).

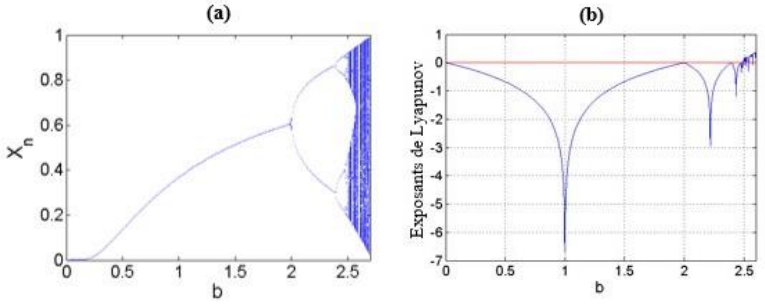


Figure 2.7 *: (a) Diagramme de bifurcation de la récurrence de Gompertz ; (b) graphe des exposants de Lyapunov de Gompertz.*

- *La récurrence gaussienne [91]*

La récurrence gaussienne dérive de l'une des équations du bruit gaussien, et a pour expression :

$$x_{n+1} = \exp\left(-\alpha x_n^2\right) + c \qquad (2.11)$$

Le paramètre $\alpha \in [4.7, 17]$, et le paramètre de contrôle est $c \in [-1, 1]$. La variable $x_n \in [0, 1]$.

La récurrence Gaussienne présente des valeurs en sortie d'itération positives ou négatives, de plus, les propriétés chaotiques son obtenues pour les valeurs négatives du paramètre de contrôle. Cependant les graphes de la figure 2.8 démontrent que la récurrence gaussienne est aussi médiocre que la logistique, car elle présente des valeurs non-uniformes, des fenêtres de périodicités (figure 2.8-a), et un faible intervalle de paramètre contrôle aux propriétés aléatoires (figure 2.8-b).

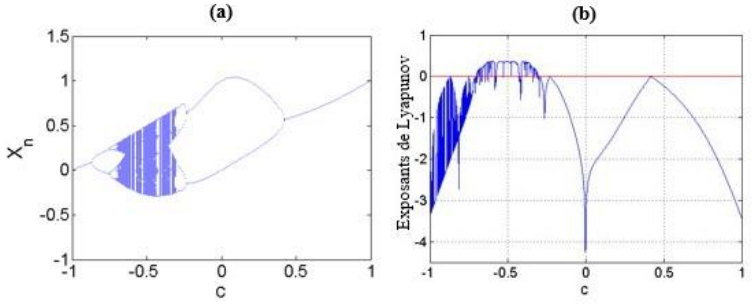


Figure 2.8 *: (a) Diagramme de bifurcation de la récurrence gaussienne ; (b) graphe des exposants de Lyapunov de la récurrence gaussienne.*

b) Méthode de mixage des récurrences GNPA

Comme démontré précédemment, les propriétés de récurrences GNPA choisies ne conviennent pas à la conception d'un algorithme de chiffrement robuste. Nous allons de ce fait fabriquer de nouvelles récurrences ayant un niveau de non-linéaire adéquate pour un algorithme de chiffrement. Nous allons pour cela adopter la méthode de mixage proposé par Zhou et al. [73] exprimé par le schéma de la figure 2.9. Ce schéma indique le moyen par lequel une nouvelle récurrence 1D peut être créée à partir d'une combinaison non-linéaire de deux récurrences 1D. Ainsi, à partir des quatre récurrences GNPA choisies, nous allons construire par mixage et par synchronisation des paramètres de contrôles, six nouvelles récurrences GNPA.

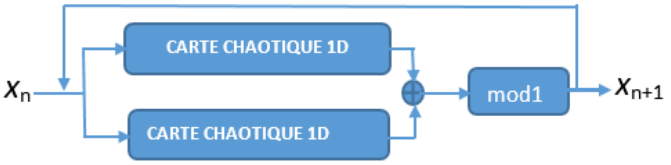


Figure 2.9 *Schéma de mixage des récurrences GNPA 1D*

c) La récurrence logistique-may

La première récurrence est obtenue à partir des récurrences logistique et May, elle est nommé LOMAS (« logistic-may system »), son équation est donnée par l'expression suivante :

$$x_{n+1} = \left(x_n \exp\left((r+9)(1-x_n)\right) - (r+5)x_n(1-x_n)\right) \bmod 1 \qquad (2.12)$$

$x_n \in [0,1]$ and $r \in [0,5]$.

Son diagramme de bifurcation et ses exposants de Lyapunov sont donnés par le graphe de la figure 2.10-a et d. Au vue de ces graphes on peut constater que ses propriétés aléatoires sont meilleures, car le diagramme de bifurcation est uniformément dense; et tous les exposants de Lyapunov sont positifs avec un maximum à 8,3.

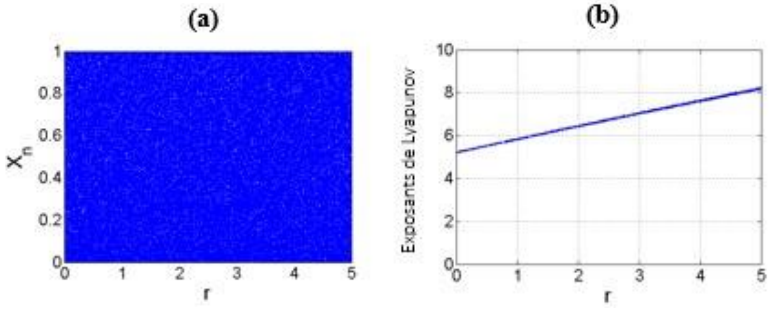


Figure 2.10 *: (a) Diagramme de bifurction de LOMAS ; (b) Graphe des exposants de Lyapunov de LOMAS.*

d) La récurrence logistique-gompertz

La seconde récurrence appelée LOGOS (« Logistic-gompertz-system ») est construite à partir de logistique et gompertz , elle a pour équation :

$$x_{n+1} = \left(-(r-31)x_n(1-x_n) - (r+35)x_n \log x_n\right) \bmod 1 \qquad (2.13)$$

$x_n \in [0,1]$ and $r \in [0,5]$.

Quoique la récurrence de Gompertz et celle de la logistique aient des propriétés chaotiques limitées, celle de leur combinaison a de très bonnes propriétés de non-linéarité, si l'on considère son diagramme de bifurcation uniforme (figure 2.11-a) et ses exposants de Lyapunov tous positifs (figure 2.11-b).

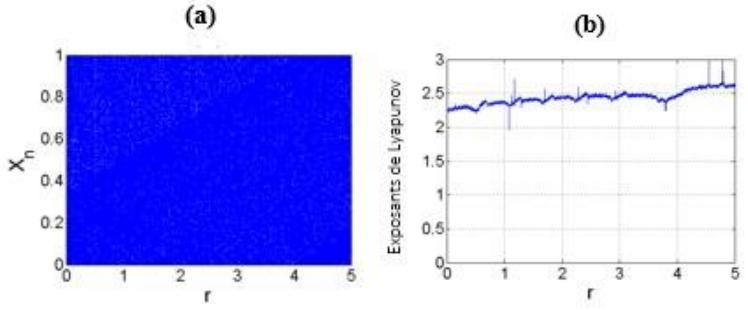


Figure 2.11 *: (a) Diagramme de bifurcation de LOGOS ; (b) Graphe des exposants de Lyapunov de LOGOS.*

e) La récurrence logistique-gaussienne

La récurrence logistique et la récurrence gaussienne ont été utilisées pour construire la troisième récurrence nommée LOGAS et donnée par l'équation suivante :

$$x_{n+1} = \left(-(r-33)x_n(1-x_n) + \frac{(r+37)}{4} + \exp(-\alpha x_n^2) \right) \bmod 1 \qquad (2.14)$$

$x_n \in [0,1]$, $r \in [0,5]$, $\alpha \in [4.7,17]$.

Le graphe du diagramme de bifurcation (figure 2.12-a) et celui des exposants de Lyapunov (figure 2.12-b) de LOGAS (Logistic-gaussian-system) démontrent à suffisance que sa dynamique est parfaitement chaotique. En effet, toutes les valeurs de son paramètre de contrôle sont tous denses et ont leurs valeurs d'exposants de Lyaponuv positifs.

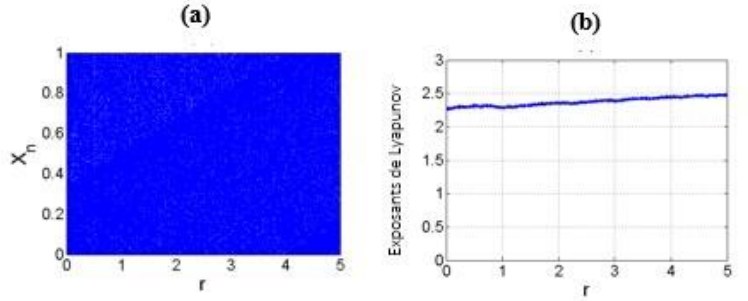


Figure 2.12 *: (a) Diagramme de bifurcation de LOGAS ; (b) Graphe des exposants de Lyapunov de LOGAS.*

f) La récurrence may-gompertz

La quatrième récurrence est le résultat du mixage de la récurrence May et Gompertz (équation 2.15)

$$x_{n+1} = \left(x_n \exp((r+10)(1-x_n)) \ -(r+10)x_n \log x_n \right) \bmod 1 \qquad (2.15)$$

$x_n \in [0,1]$ and $r \in [0,5]$.

La figure 2.13-a et c représentent le diagramme de bifurcation et les exposants de Lyapunov de MAGOS (may-gompertz-system), elles montrent respectivement un diagramme uniforme et des valeurs positives des exposants de Lyapunov ayant maximum à 8.5. Ces résultats prouvent ainsi l'excellente dynamique aléatoire de MAGOS.

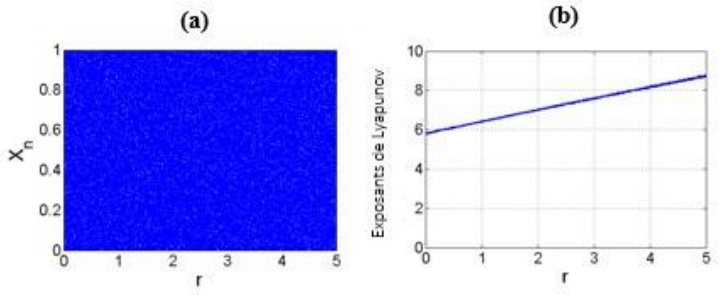


Figure 2.13 *: (a) Diagramme de bifurcation de MAGOS ; (b) Graphe des exposants de Lyapunov de MAGOS.*

g) La récurrence may-gaussienne

Le Mixage de la récurrence May à la récurrence gaussienne engendre la 5ième Récurrence appelé MAGAS (May-gaussian-system) laquelle a pour équation :

$$x_{n+1} = \left(x_n \exp\left((r+10)(1-x_n)\right) + \frac{(r+5)}{4} + \exp\left(-\alpha x_n^2\right) \right) \bmod 1 \qquad (2.16)$$

$x_n \in [0,1]$, $r \in [0,5]$, $\alpha \in [4.7,17]$.

A travers le diagramme de bifurcation de MAGAS à la figure 2.14-a, l'on peut voir que toutes les séquences sont uniformément distribuées dans l'intervalle [0, 1]. Par ailleurs, la figure 2.14-b montrent tous ses exposants de Lyapunov positifs.

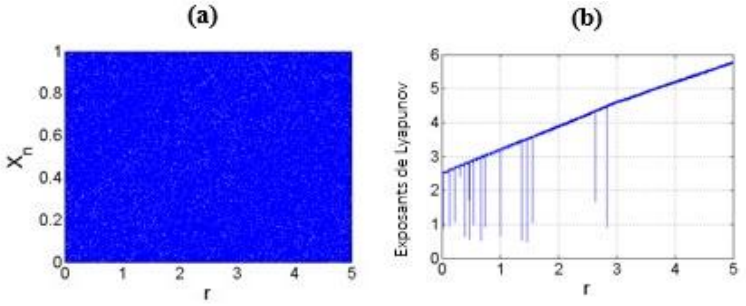


Figure 2.14 *: (a) Diagramme de bifurcation de MAGAS ; (b) Graphe des exposants de Lyapunov de MAGAS.*

h) La récurrence gaussienne-gompertz

La dernière récurrence conçue est le mixage de la récurrence gaussienne et Gompertz. Elle est nommée GAGOS (gaussian-gompertz-system) et s'exprime par la formule :

$$x_{n+1} = \left(\frac{(r/5+26)}{4} + \exp\left(-\alpha x_n^2\right) - (r/5+26) x_n \log x_n \right) \bmod 1 \qquad (2.17)$$

$x_n \in [0,1]$, $r \in [0,5]$, $\alpha \in [4.7,17]$.

Le diagramme de bifurcation (figure 2.15-a) de GAGOS montre une distribution uniforme des séquences, et le graphe des exposants de Lyapunov (figure 2.15-b) ne contient que des valeurs positives supérieurs à 2.

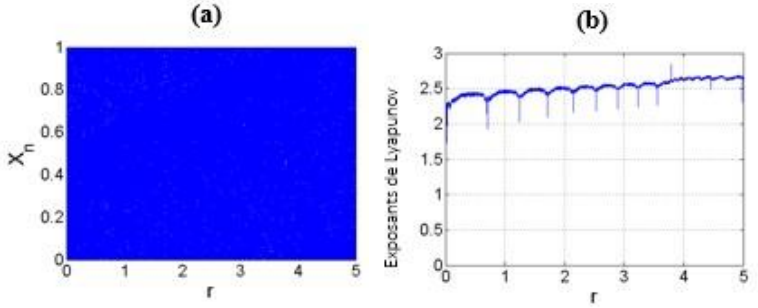


Figure 2.15 *: (a) Diagramme de bifurcation de GAGOS ; (b) Graphe des exposants de Lyapunov de GAGOS.*

2.2.2 Avantages des nouvelles récurrences

Toutes les récurrences chaotiques 1D conçu (LOMAS, LOGOS, LOGAS, MAGOS, MAGAS, GAGOS), ont des propriétés chaotiques meilleures que ceux de leurs génératrices, bien que l'étude de la théorie mathématique en rapport avec ses récurrences

n'ait pas été faite. À travers les principaux outils de détection du nonlinéaire et de l'aléatoire tel que le diagramme de bifurcation et les exposants de Lyapunov, on a pu constater :

- Un intervalle de valeurs de paramètres de contrôles aux propriétés chaotiques bien plus large ;
- Une distribution uniforme des valeurs des variables discrètes ;
- Une absence totale des zones de périodicités ;
- Des exposants de Lyapunov tous positifs ;
- Une densité pleine sur l'ensemble des valeurs du paramètre de contrôle.

Par ailleurs, les valeurs maximales des exposants de Lyapunov de LOMAS, LOGOS, LOGAS, MAGOS, MAGAS, GAGOS sont respectivement de 8.1 ; 2.6 ; 2.5 ; 8.7 ; 5.6 ; et 2.5 (figure 2.10 ; 2.11 ; 2.12 ; 2.13 ; 2.14 ; 2.15). Elles sont toutes supérieures à celles des récurrences logistiques, May, Gompertz et gaussienne dont les valeurs maximales sont : 0.6 ; 0.4 ; 0.5 ; 0.5 et 0.7 (figure 2.6; 2.7; 2.8) [89-91]. Les valeurs des exposants de Lyapunov élevés signifient : peu d'itération pour peu d'effet transitoire, pour avoir à l'opposé deux séquences de nombres aléatoires complètement différentes, à partir de deux conditions initiales sensiblement identiques. Cet avantage fait des nouvelles récurrences, de meilleures génératrices de nombres aléatoires pour la cryptographie que les anciennes.

2.2.3 Choix du type clé

La clé sera interne, c'est-à-dire constituée simplement par les valeurs des paramètres de contrôles et des conditions initiales servant à l'itération des récurrences GNPA, pour la production des nombres aléatoires.

2.2.4 Algorithme de chiffrement proposé

Le nouvel algorithme de chiffrement que nous avons proposé repose sur deux procédures principales dont : une nouvelle technique de substitution de l'image en clair; et un algorithme de chiffrement basé sur la technique de masquage-brouillage.

a) Technique de substitution de l'image en clair

La technique de substitution de l'image en clair (« plain image substitution technique » ou PIST) appliquée à une image, a pour principale but d'augmenter la sensibilité dans la dépendance inter-pixels de l'image, de sorte que la moindre modification d'un pixel de l'image créée l'effet d'avalanche, et modifie substantiellement

tous les autres pixels. Cette technique est un prétraitement de l'image qui ne nécessite pas l'usage d'une clé. Les étapes pour l'appliquer à une image sont :

- Pour chaque colonne d'une image $M \times N$ et partant de l'avant-dernier pixel (en bas) vers le premier (en haut), remplace la valeur d'un pixel en cours de traitement par celle obtenue de l'opération XOR entre ce dernier et la valeur de pixel précédente (équation 2.17).

$$\begin{cases} i = 1,...M-1;\ j = 1,...N \\ I(M-i,j) = I(M-i,j) \oplus I(M+1-i,j) \end{cases} \tag{2.18}$$

- Répète la même opération pour chaque ligne mais en partant du dernier pixel à droite vers le premier à gauche (équation 2.18).

$$\begin{cases} i = 1,...M;\ j = 1,...N-1 \\ I(i,N-j) = I(i,N-j) \oplus I(i,N+1-j) \end{cases} \tag{2.19}$$

Le PIST offre l'avantage à l'image traitée de confiner les caractéristiques de tous les pixels dans les directions horizontales et verticales, et finalement dans le premier pixel. Cette technique serait très avantageuse en mode CBC et PCBC, car dans ces modes, les caractéristique du pixel n° 1 de la chaine (ou séquence à chiffrer) et son équivalent chiffré affectent facilement (à cause du chaînage) le reste lors du chiffrement. On peut donc la recommander à tous les algorithmes de chiffrement souffrant d'insensibilité par rapport à un changement d'image claire [92-107]. A la figure 2.16, on peut observer l'image confuse résultat du traitement de l'image Lena par l'algorithme PIST.

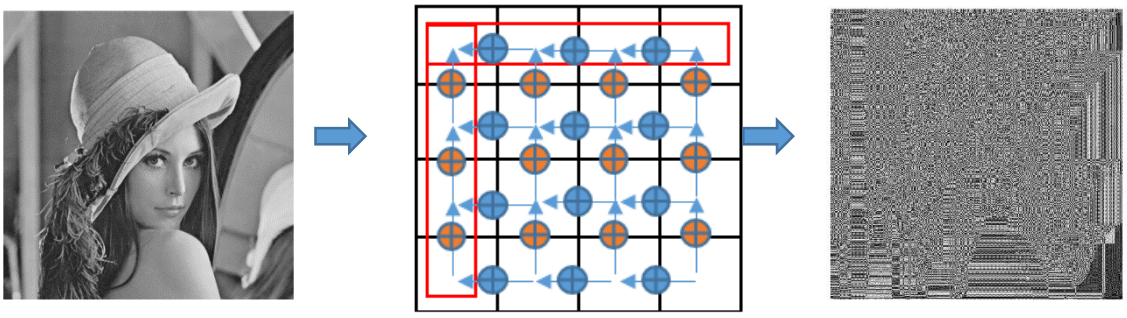


Figure 2.16 *Effet du PIST sur une image Lena à niveau de gris.*

b) Principe de chiffrement basé sur le masquage-brouillage et utilisant les S-box

Le chiffrement proposé utilisera pour le cas d'espèce les couples de clés (w_0=0.4 ; r_1=1), (x_0=0.3 ; r_2=2); (r_3=3 ; y_0=0.2); (z_0=0.1 ; r_4=4) et α=6 .Ces couples sont aussi des couples de paramètres de contrôles et de conditions initiales appartenant respectivement

aux récurrences LOMAS, LOGOS, LOGAS, MAGOS. Par ailleurs, une image en clair I de taille $M\times N$ prétraité à l'algorithme PIST donnera une image chiffrée C après le masquage-brouillage, et au final une image C' après le brouillage de pixels.

Les étapes ayant servies à la conception de l'algorithme sont :

Etape 1 *:* Après 500 itérations pour éviter les effets de transition, continuer les itérations de LOMAS, LOGOS, LOGAS et MAGOS pour créer 4 séquences (vecteurs colonnes et/ou vecteur ligne) respective W, X, Y, Z, de taille $M\times N+100$ fois. Construire ensuite deux matrices de taille $M\times N$ dont la première S_x en utilisant la séquence X, et la seconde S_y, la séquence Y.

Etape 2 : Effectue le chiffrement de la première ligne et de la première colonne de l'image en utilisant les nombres pseudo-aléatoires des séquences X et Y selon les équations :

$$\begin{cases} j = 2,3,...N \text{ and } i = 1,2,...M \\ C(1,j) = I(1,j) \oplus \left\lfloor (X(j+100)\times 10^{15}) \bmod 256 \right\rfloor \oplus \left\lfloor (Y(j+100)\times 10^{15}) \bmod 256 \right\rfloor \\ C(i,1) = I(i,1) \oplus \left\lfloor (X(i+200)\times 10^{15}) \bmod 256 \right\rfloor \oplus \left\lfloor (Y(i+200)\times 10^{15}) \bmod 256 \right\rfloor \end{cases} \quad (2.20)$$

Etape 3 : Pour chaque valeur chiffrée $C(i,1)$ de la première colonne et $C(j,1)$ de la première ligne, calcule les nombres $l(i)$ et $k(j)$ (Equation 2.21), ensuite, utilise chacun d'eux comme indice d'extraction dans les séquences de valeurs de positions $i=\{2, 3,...M\}$ pour les lignes, et $j=\{2, 3,...N\}$ pour les colonnes. Forme deux nouvelles séquences de valeurs de positions aléatoirement distribuées avec les valeurs de positions extraites.

$$\begin{cases} i = 2,3...N \text{ and } j = 2,3...M \\ l(i) = 1 + \left(C(i,1) \oplus \left\lfloor (Z(i+200)\times 10^{15}) \bmod 256 \right\rfloor \times \left\lfloor (W(i+200)\times 10^{15}) \right\rfloor \right) \bmod (M+1-i) \\ k(j) = 1 + \left(C(1,j) \oplus \left\lfloor (Z(j+100)\times 10^{15}) \bmod 256 \right\rfloor \times \left\lfloor (W(j+100)\times 10^{15}) \right\rfloor \right) \bmod (N+1-j) \end{cases} \quad (2.21)$$

Etape 4 : Remplace les pixels d'indices $i = \{2, 3...M\}$ et $j = \{2, 3...N\}$ par ceux d'indices a et b extraits des séquences des valeurs de positions aléatoirement distribuées obtenues à l'étape 3. Exerce un processus de masquage-brouillage utilisant les S-boxes S_x et S_y, en utilisant l'équation suivante :

$$\begin{cases} i = 2,3...M \text{ and } j = 2,3...N \\ C(a,b) = I(i,j) \oplus S_x(i,b) \oplus S_y(a,j) \end{cases} \quad (2.22)$$

Les valeurs des éléments des S-boxes (S_x et S_y) sont des valeurs des niveaux de gris obtenues en faisant respectivement $\lfloor x(n)\times 10^{15} \rfloor \bmod 256$ et $\lfloor x(n)\times 10^{15} \rfloor \bmod 256$, avec $n = \{1,2...M\times N\}$.

Etape 5 : Calcule les séquences *u(i)* et *v(j)* d'indices d'extraction des valeurs de positions respective des lignes $i=\{2, 3...M\}$, et des colonnes $j=\{2, 3...N\}$, en utilisant les séquences (*W, X, Y, Z*) des quatre récurrences et les pixels chiffrés *C(1,1), C(1,2), C(1,3), C(2,1), C(3,1),* selon l'équation (2.23). Effectue un brouillage en remplaçant le couple d'indice *(i, j)* de chaque pixel chiffré de l'étape 4 par celui d'un couple *(c, d)* résultat des valeurs de positions extraites par *u(i)* et *v(j)*.

$$\begin{cases} i=1,2,...M \ \text{ and } \ j=1,2,...N \\ u(i)=1+\left(C(1,1)\oplus\lfloor (W(i+C(1,2))\times 10^{15})\bmod 256\rfloor\times\lfloor (X(i+C(1,3))\times 10^{15})\rfloor\right)\bmod(M+1-j) \\ v(j)=1+\left(C(1,1)\oplus\lfloor (Y(j+C(2,1))\times 10^{15})\bmod 256\rfloor\times\lfloor (Z(j+C(2,3)\times 10^{15})\rfloor\right)\bmod(N+1-i) \end{cases} \quad (2.23)$$

Etape 6 : Associer une copie des valeurs *C(1,1), C(1,2), C(1,3), C(2,1), C(3,1)* aux valeurs de la clé pour le transférer au récepteur.

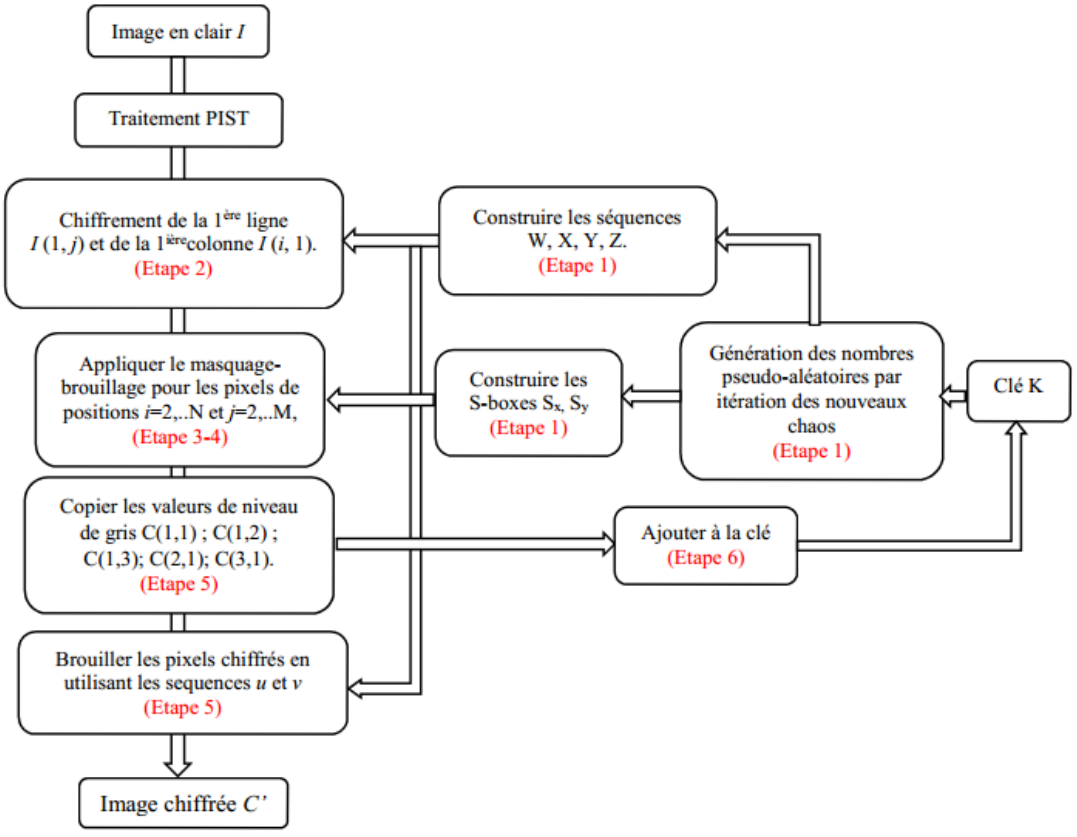


Figure 2.17 *Logigramme de l'algorithme de chiffrement*

c) Chiffrement de l'image couleur

Le principe de chiffrement établit précédemment reste identique pour les images couleur (RVB). Cependant, pour améliorer la sensibilité de l'image couleur par rapport à la fluctuation d'un pixel, les matrices R, V et B de l'image couleur seront combinées pour former une matrice unique avant le chiffrement, et restaurer à l'état R, V et B après le chiffrement.

d) Principe de déchiffrement

Le déchiffrement se fait en trois étapes :

Etape1 : Inverser le processus de brouillage de l'étape 5 de chiffrement en utilisant comme clé les conditions initiales, les paramètres de contrôles, les valeurs de clé *C(1,1), C(1,2), C(1,3), C(2,1), C(3,1),* ainsi que les équations de cette étape.

Etape 2 : Inverser le processus de masquage-brouillage de la sous-matrice ayant pour indices $i=\{2, 3,\ldots M\}$ et $j=\{2, 3,\ldots N\}$; en utilisant la clé, les séquences (W, X, Y, Z) générées des quatre récurrences, les valeurs de pixels de la première ligne et de la première colonne, selon l'équation suivante :

$$\begin{cases} i = 2,3...M \text{ and } j = 2,3...N \\ I(i,j) = C(a,b) \oplus S_x(i,b) \oplus S_y(a,j) \end{cases} \tag{2.24}$$

Etape 3 : Effectuer le déchiffrement de la première ligne et de la première colonne et inverser le processus du PIST.

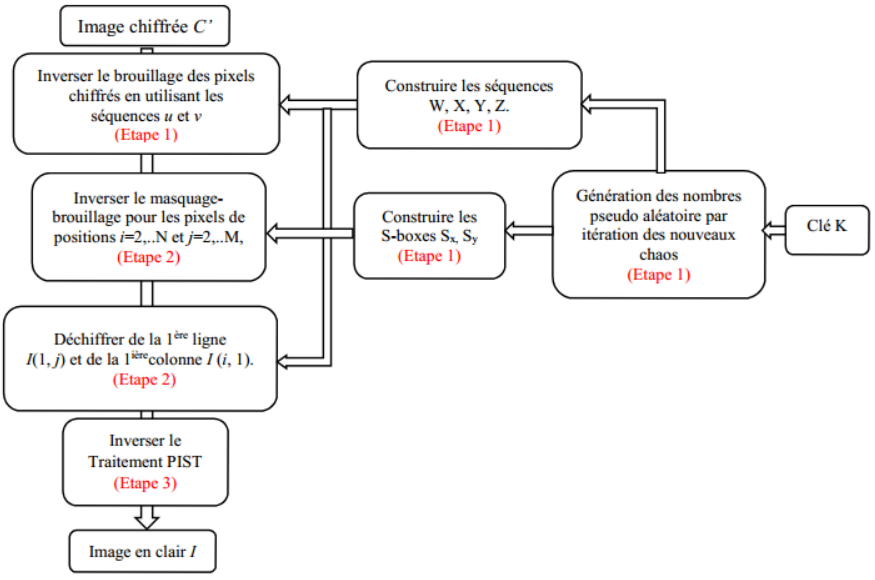


Figure 2.18 *Logigramme de l'algorithme de déchiffrement*

2.3 Deuxième contribution : Utilisation adéquate de la récurrence logistique dans un chiffrement à flot auto-synchronisé pour le chiffrement d'image.

L'algorithme présenté ici est un algorithme de chiffrement à flot auto-synchrone, utilisant uniquement la récurrence logistique (équation 2.8) dans une architecture de diffusion-permutation. Il repose sur la capacité à renouveler la clé de chiffrement pour chaque pixel, ce renouvellement se fait en utilisant les clés initiales et les paramètres générés aux moyens des valeurs des pixels chiffrés précédemment.

2.3.1 Procédure de génération de la clé

L'algorithme utilise une clé secrète externe K de 256-bits de longueur. Des opérations reposant sur la rotation (« circular-shift ») et les algorithmes sécurisés de hachage (« secure hash algorithm » en anglais et codé *SHA*), sont appliquées à la clé avec pour objectif d'augmenter la sensibilité de cette dernière, à la moindre variation de sa valeur (figure 2.18). À l'issue de cette opération une autre est effectuée pour générer les conditions initiales et les paramètres de contrôles devant être utilisés pour le chiffrement.

a) Augmentation de la sensibilité de la clé

Dans le processus de rehaussement de la sensibilité de la clé K, l'opération de rotation circulaire à gauche ($circshift_l$) sur les bits d'une clé K produit une clé K' (équation 2.24). Et l'expression $K= k_i , k_j$, signifie : extrait deux nombres k_i et k_j du nombre K, le premier étant constitué de la moitié de ses bits les plus significatifs, et le second de l'autre moitié les moins significatif.

$$K' = circshift_l(K,q) = \sum_{i=0}^{q-1} b_i.2^{(q-(i+1))} \tag{2.25}$$

q est la longueur binaire de K, la fonction $circshift_l$ déplace un bit de la position la moins significative à la position la plus significative dans un mot binaire.

$K = K_1, K_2$	$K_3 = K_1^{'} \oplus K_2$	$K_4 = K_1 \oplus K_2^{'}$
$K_3 = K_5, K_6$	$K_4 = K_7, K_8$	$K_9 = K_5^{'} \oplus K_6$
$K_{10} = K_5 \oplus K_6^{'}$	$K_{11} = K_7^{'} \oplus K_8$	$K_{12} = K_7 \oplus K_8^{'}$
$K_{9} = K_{13}, K_{14}$	$K_{10} = K_{15}, K_{16}$	$K_{11} = K_{17}, K_{18}$
$K_{12} = K_{19}, K_{20}$	$K_{21} = K_{13} \oplus K_{14}^{'}$	$K_{22} = K_{13}^{'} \oplus K_{14}$
$K_{23} = K_{15}^{'} \oplus K_{16}$	$K_{24} = K_{15} \oplus K_{16}^{'}$	$K_{25} = K_{17}^{'} \oplus K_{18}$
$K_{26} = K_{17} \oplus K_{18}^{'}$	$K_{27} = K_{19} \oplus K_{20}^{'}$	$K_{28} = K_{19}^{'} \oplus K_{20}$

Figure 2.19 *: Processus de rehaussement de la sensibilité de la clé.*

b) Calcul des paramètres de la récurrence logistique

Les conditions initiales nécessaires aux chiffrements sont les nombres x_0, y_0 appartenant à l'intervalle [0, 1] et sont calculées par les équations (2.26) à (2.29). Par ailleurs, l'unique paramètre de contrôle λ_0, sera généré par l'équation (2.30), il devra être toujours supérieur à 3.90 ($\lambda_0 \geq 3.90$). Tous ces paramètres serviront à l'itération de la récurrence logistique.

$$\alpha = \frac{(K_{22} \oplus K_{23} \oplus K_{25} \oplus K_{26} \oplus K_{27} \oplus K_{28})_{10}}{2^{32}} \tag{2.26}$$

$$\beta = \frac{(K_{21} \oplus K_{22} \oplus K_{23} \oplus K_{24} \oplus K_{26} \oplus K_{27})_{10}}{2^{32}} \tag{2.27}$$

$$x_0 = \frac{(K_{21} \oplus K_{22} \oplus K_{24} \oplus K_{25} \oplus K_{27} \oplus K_{28})_{10}}{2^{32}} \tag{2.28}$$

$$y_0 = \frac{(K_{21} \oplus K_{23} \oplus K_{24} \oplus K_{25} \oplus K_{26} \oplus K_{28})_{10}}{2^{32}} \tag{2.29}$$

$$\lambda_0 = 3{,}9 + \frac{(K_{21} \oplus K_{22} \oplus K_{23} \oplus K_{24} \oplus K_{25} \oplus K_{26} \oplus K_{27} \oplus K_{28})_{10}}{2^{32}} \times 0.1 \tag{2.30}$$

2.3.2 Procédure de chiffrement de l'image

a) Procédure de diffusion

La diffusion utilise les clés générées par la clé externe dans une procédure dont les étapes sont les suivantes :

Etape 1 : Une image (matrice 2D) I de taille $M \times N$ sera transformée en une séquence (vectorielle) de valeurs de pixel, $p_1, p_2, ..., p_{MN}$ de longueur MN avec $i = \{1, 2, 3, ..., MN\}$.

Etape 2 : Cette étape est constituée de deux sous-étapes a) et b)

a) Pour le chiffrement du premier pixel de l'image I, calcule le paramètre de contrôle λ_1 et la condition initiale $x_0(1)$ selon l'équation (2.31).

$$\begin{cases} \lambda_1 = \lambda_0 + \left((-1)^{\lfloor x_0 \times 10^{16} \rfloor} \times \left[(\lfloor (\beta + y_0) \times 256 \rfloor) \bmod 256 + 1 \right]^{-\alpha} \times x_0 \right) \times 2 \times 10^{-4} \\ x_0(1) = (x_0 + y_0)/(2 + (\alpha + \beta) \bmod 1) \end{cases} \quad (2.31)$$

b) Utilise λ_1, $x_0(1)$ pour effectuer les itérations de la récurrence logistique N=101 fois et calculer le pixel à chiffrer avec la 101ième valeur comme suit :

$$\begin{cases} Sk_1 = \left(\lfloor x_{101}(1) \times 10^{16} \rfloor \right) \bmod 256 \\ c_1 = p_1 \oplus Sk_1 \end{cases} \quad (2.32)$$

Etape 3 : Pour chaque valeur de pixel P_{i+1} à chiffrer, actualise la condition initiale $x_0(i+1)$ et le paramètre de contrôle λ_1 suivant l'équation (2.33). Effectue 21 itérations et utilise le résultat de la 21ième itération dans l'équation (2.34) pour calculer le pixel chiffré.

$$\begin{cases} \lambda_{i+1} = \lambda_i + \left((-1)^{\lfloor x_N(i) \times 10^{16} \rfloor} \times \left[(p_i + c_i) \bmod 256 + 1 \right]^{-\alpha} \times x_N(i) \right) \times 2 \times 10^{-4} \\ x_0(i+1) = (x_0(i) + c_i / 2^8)/(2 + \beta) \end{cases} \quad (2.33)$$

$$\begin{cases} Sk_{i+1} = \left(\lfloor x_{21}(i+1) \times 10^{16} \rfloor \right) \bmod 256 \\ c_{i+1} = p_{i+1} \oplus Sk_{i+1} \end{cases} \quad (2.34)$$

le symbole $\lfloor x \rfloor$ arrondi l'élément x à l'entier naturel directement inférieur ou égal à x, mod est l'opération modulo et le symbole $\oplus$ est celui de l'opération XOR. La variable $x_{21}(i)$ représente la valeur obtenu de la récurrence logistique après la 21ième itération,

$x_0(i)$ and λ_i sont les conditions initiales et les paramètres de contrôle de chaque valeur de pixel p_i.

Etape 4 : Transforme la séquence 1D des pixels chiffrés $c_1, c_2, ..., c_i$ obtenus en une matrice image 2D de taille $M \times N$.

b) Procédure de permutation

La permutation utilise les clés générées par les équations (2.28)-(2.30). Les étapes de la procédure de permutations sont les suivantes :

Etape 1 : Génère les conditions initiales et les paramètres de contrôles pour les lignes et pour les colonnes comme suit :

$$\begin{cases} x_0{}^V = \left[\left(c_1 \oplus c_2 \oplus .... \oplus c_{MN} + \lfloor \alpha \times 256 \rfloor\right) \bmod 256 + 1\right]^{-\beta} \\ x_0{}^H = \left(y_0 + c_{MN}/2^8\right)/\left(2 + x_0\right) \\ \lambda_V = 3.90 + \left[\left((\lambda_0 - 3.90) + (x_0{}^W)\right) \bmod 1\right] \times 0.1 \\ \lambda_H = 3.90 + \left[\left((\lambda_0 - 3.90) + (x_0{}^H)\right) \bmod 1\right] \times 0.1 \end{cases} \tag{2.35}$$

$x_0{}^V, \lambda_V$ et $x_0{}^H$, λ_H sont respectivement la condition initiale et le paramètre de contrôle dans les directions verticales (colonne) et horizontales (ligne).

Etape 2 : Pour éviter les effets de transition, effectue l'itération de la récurrence logistique au moins 100 fois pour les lignes et pour les colonnes, continue l'itération M fois pour les lignes et N fois pour les colonnes.

Etape 3 : Ordonner les valeurs des séquences aléatoires obtenues à l'étape 2 dans l'ordre croissant.

Etape 4 : Pour chaque couple ligne-colonne *(i, j)* de la position d'un pixel en cours de traitement, cherche la position précédente ligne du nombre aléatoire (correspondant) de la séquence ligne ordonnée; et colonne du nombre aléatoire (correspondant) de la séquence colonne ordonnée, constitue un nouveau couple ligne-colonne avec ses positions et attribue les au pixel en cours traitement.

Etape 5 : Transpose le dernier pixel de l'image chiffré avec celui du pixel chiffré et occupant cette position dans l'image en clair.

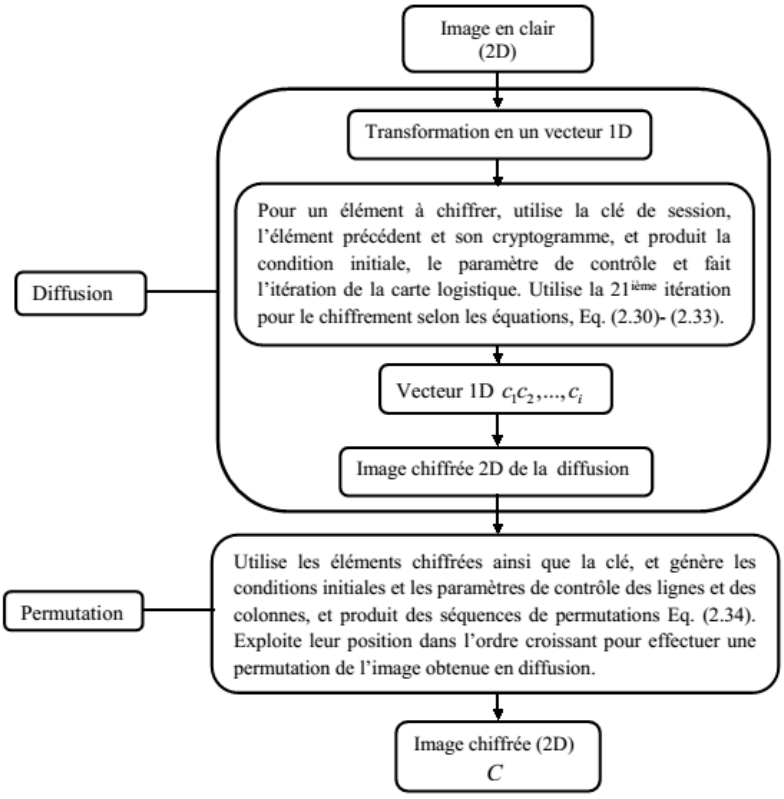


Figure 2.20 *Procédure de chiffrement*

2.3.3 Procédure de déchiffrement

La procédure de déchiffrement en diffusion et en permutation est la même que celle du déchiffrement, à la différence de quelques variantes dans les équations de diffusion. Il est à noter que la permutation se fait avant la diffusion en déchiffrement.

Les équations de déchiffrement en diffusion sont :

Etape 1 : Procédure identique

Etape 2 :

a)

$$\begin{cases} \lambda_1 = \lambda_0 + \left((-1)^{\left\lfloor x_0 \times 10^{16} \right\rfloor} \times \left[\left(\left\lfloor (\beta + y_0) \times 256 \right\rfloor \right) \bmod 256 + 1 \right]^{-\alpha} \times x_0 \right) \times 2 \times 10^{-4} \\ x_0(1) = (x_0 + y_0)/(2 + (\alpha + \beta) \bmod 1) \end{cases} \quad (2.36)$$

b)

$$\begin{cases} Sk_1 = \left(\left\lfloor x_{101}(1) \times 10^{16} \right\rfloor \right) \bmod 256 \\ p_1 = c_1 \oplus Sk_1 \end{cases} \quad (2.37)$$

Etape 3 :

$$\begin{cases} \lambda_{i+1} = \lambda_i + \left((-1)^{\left\lfloor x_N(i)\times 10^{16} \right\rfloor} \times \left[(p_i + c_i) \bmod 256 + 1 \right]^{-\alpha} \times x_N(i) \right) \times 2 \times 10^{-4} \\ x_0(i+1) = (x_0(i) + c_i / 2^8) / (2+\beta) \end{cases} \tag{2.38}$$

$$\begin{cases} Sk_{i+1} = \left(\left\lfloor x_{21}(i+1) \times 10^{16} \right\rfloor \right) \bmod 256 \\ p_{i+1} = c_{i+1} \oplus Sk_{i+1} \end{cases} \tag{2.39}$$

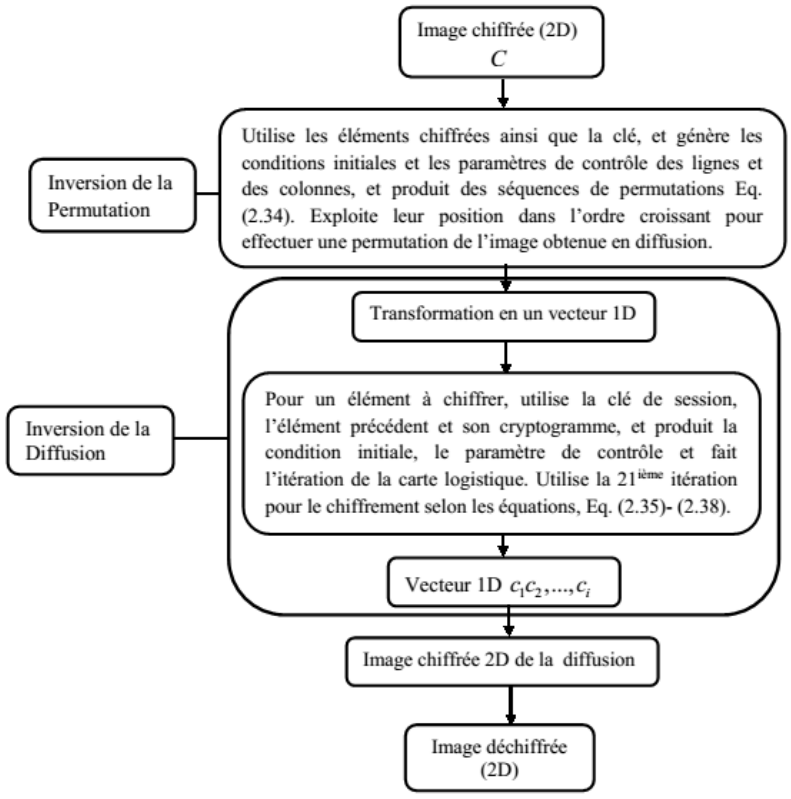


Figure 2.21 *Procédure de déchiffrement*

2.3.4 Procédure de chiffrement/déchiffrement d'une image couleur

La procédure est la même que pour une image monochrome ou en niveau de gris. Les trois matrices R, V et B sont chiffrées séquentiellement comme si elles étaient combinées sur l'horizontale (figure 2.21).

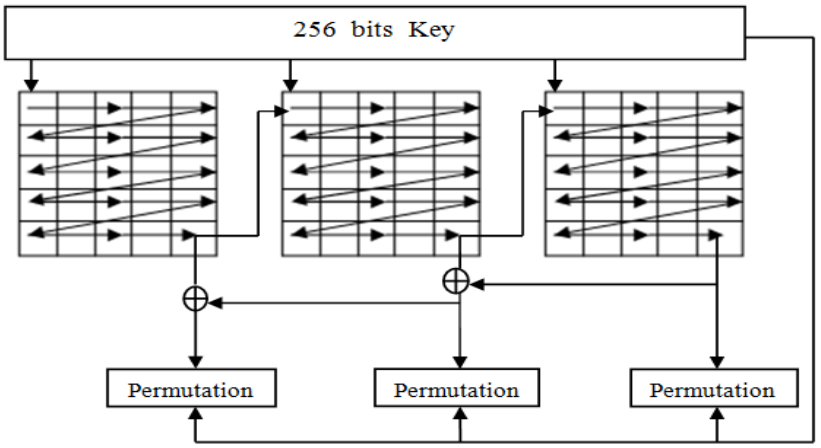


Figure 2.22 *Schéma du processus de chiffrement de l'image couleur*

2.4 Outils et métriques d'analyse de sécurité d'un cryptosystème d'images

En appliquant un algorithme de chiffrement à une image, les pixels de l'image chiffrée doivent être très différents, et totalement indépendants (faible corrélation), en comparaison de ceux de l'image originale. Cela doit être constaté par simple visualisation de l'image cryptée. Cependant La simple inspection visuelle reste insuffisante pour juger de la qualité d'un algorithme de chiffrement d'image. De plus, certaines évaluations comme le temps de chiffrement ou les attaques ne peuvent être quantifiés qu'avec des outils. On classe ainsi les métriques d'évaluation d'un cryptosystème en grand groupe dont quelques-uns sont : l'analyse statistique (histogramme, variance d'histogramme, corrélation entre pixels adjacents, l'entropie de Shannon) ; l'analyse de la clé (espace de clé et la sensibilité de la clé) ; l'analyse différentielle ; la cryptanalyse de base (attaque à texte clair choisi, attaque à texte chiffré choisi) ; le temps d'exécution de l'algorithme).

2.4.1 Analyse statistique

a) Histogramme et la variance d'histogramme

- *L'histogramme*

Un histogramme est une courbe statistique indiquant la répartition des pixels selon les valeurs de niveau de gris ou de monochromie. C'est aussi une métrique visuelle qui permet de contrôler l'exposition d'une image. En d'autres termes, tous les histogrammes d'une image chiffrée doivent avoir des pixels uniformément distribués à tout niveau de gris ou de monochromie (figure 2.22).

Un histogramme d'image en niveau de gris indique pour chaque valeur entre 0 (le noir) et 255 (le blanc), combien il y a de pixels de cette valeur dans l'image. Les pixels sombres apparaissent donc le plus à gauche de l'histogramme, les pixels clairs le plus à droite de l'histogramme et les pixels gris au centre de l'histogramme.

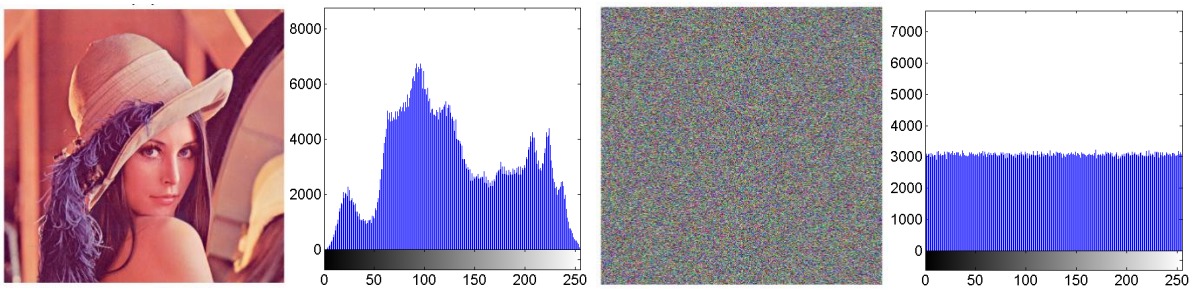


Figure 2.23 *De gauche à droite : l'image en clair Lena ; histogramme de l'image Lena ; image Léna chiffrée ; histogramme de l'image Lena chiffrée.*

- *La variance d'histogramme*

Cette métrique quantifie l'uniformité dans la distribution des pixels d'un histogramme. Elle se calcule avec l'équation suivante :

$$Var(z)=\frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{1}{2}(z_i-z_j)^2 \qquad (2.40)$$

Z est le vecteur des valeurs d'histogramme $Z=\{z_1,z_2,...,z_{256}\}$, z_i et z_j sont les nombres de pixels dont les niveaux de gris sont respectivement égaux à i et à j.

Les valeurs de variances d'histogrammes inférieurs à 6000 sont considérées comme des valeurs caractérisant un histogramme aux pixels uniformément répartis.

b) Corrélation entre pixels adjacents

Une caractéristique intrinsèque de toutes images est la forte redondance des valeurs de pixels localisées. Cela se traduit mathématiquement par une corrélation entre pixels adjacents, et peut contribuée aisément à trouver une relation entre l'image chiffrée et l'image en clair. C'est pourquoi l'analyse de corrélation entre deux pixels adjacents est nécessaire.

- *Le calcul du coefficient de corrélation d'une image chiffrée*

La méthode la plus pratique pour évaluer le coefficient de corrélation *Cr* d'une image consiste à choisir aléatoirement 5000 paires de pixels dans les directions : horizontale, verticale et diagonale, et d'implémenter l'équation (2.41) :

$$Cr = \frac{K\times\sum_{i=1}^{K} X_i Y_i - \sum_{i=1}^{K} X_i^2 \times \sum_{i=1}^{K} Y_i^2}{\sqrt{\left(K\times\sum_{i=1}^{K}(X_i)^2 - \left(\sum_{i=1}^{K} X_i\right)^2\right)\times\left(K\times\sum_{i=1}^{K}(Y_i)^2 - \left(\sum_{i=1}^{K} Y_i\right)^2\right)}} \tag{2.41}$$

X et Y sont les valeurs de niveau d'intensité lumineuse entre deux pixels adjacents de l'image, K est le nombre de paires de pixels. Cr est la valeur de corrélation appartenant à l'intervalle [-1,1].

Une valeur de Cr tendant vers 1 ou -1 signifie une très forte corrélation, mais une valeur tendant vers 0 exprime une très faible corrélation.

- *Le graphique du coefficient de corrélation*

Le graphisme du coefficient de corrélation permet d'observer la proximité des valeurs de pixels adjacents dans une direction donnée. Dans les exemples de la figure 2.24-a et b, on peut voir la différence entre le graphe du coefficient de corrélation d'une image en clair et celle d'une image chiffrée.

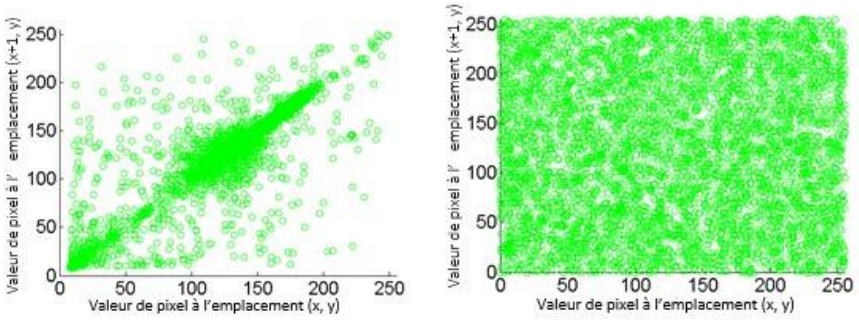


Figure 2.24: *(a) Graphe du coefficient de corrélation d'une image claire; (b) Graphe du coefficient de corrélation d'une image chiffrée.*

c) Entropie de Shannon

Dans le contexte de la cryptographie reposant sur les GNPS, l'état d'un cryptogramme a une entropie déterminée en général par le cryptosystème utilisé. Il est alors très important de s'assurer que les valeurs de l'entropie obtenues après un chiffrement, ne permettent pas à un tiers de deviner un segment du cryptogramme, ou de trouver une relation liée aux paramètres de la récurrence chaotique utilisée.

L'entropie de Shannon d'une image est un critère utilisé pour évaluer le désordre ou l'aléa dans une donnée. Une grande valeur de cette entropie détermine une meilleure distribution des niveaux de gris de l'image. Ainsi, l'entropie H d'un message M sera donnée par :

$$H(M)=\sum_{i=0}^{2^K-1} p(m_i)\log_2\left(1/p(m_i)\right) \tag{2.42}$$

$p(m_i)$ représente la probabilité du symbole m_i K est le nombre de bits du message et 2^K toutes les valeurs binaires possibles du message. Pour une image représentée par 256 niveau de gris, le pixel a 2^8 valeurs possibles, et l'entropie idéale d'une image parfaitement chiffrée vaudra 8.

2.4.2 Analyse de la clé

L'espace de clé et la sensibilité d'une clé sont des critères important pour éviter une attaque force brute naïve ou sélective effective.

a) L'espace de clé

Une condition nécessaire mais pas suffisante pour qu'un cryptosystème soit sûr, c'est qu'il doit avoir un espace de clé suffisamment large pour résister une attaque de force brute. Cette attaque consiste à casser un algorithme de chiffrement par le test exhaustif de toutes les clés possibles. A cet effet, elle utilise la puissance de plusieurs processeurs travaillant en parallèle. Compte tenu de la puissance des processeurs de nos jours, il est admis qu'un espace de clé de taille $K>2^{100}$ est assez sûr. Cependant la taille de la clé de certains cryptosystèmes dépend de leur usage (tableau 2.3).

Tableau 2.3 *: Taille de la clé requise pour différent type d'information*

Type de trafic	**Durée de vie**	**Taille minimum de la clé**
Information militaire tactique	Minutes/ heures	56-64 bits
Annonce de produit, fusion, taux d'intérêt d'entreprise	Jours/semaines	64 bits
Projet d'entreprise à long terme	années	64 bits
Secrets d'entreprises	décennies	112 bits
Le secret de la bombe H	Supérieur à 40 ans	128 bits
Identités des espions	Supérieurs à 50 ans	128 bits
Secret diplomatique	Supérieurs à 65 ans	Au moins 128 bits

b) La sensibilité de la clé

Un cryptosystème doit être sensible à la plus insignifiante valeur décimale de sa clé, afin de pouvoir résister à une attaque à image claire ou image chiffrée choisie. Le moindre changement dans la clé doit produire une image chiffrée complètement différente. On

évalue donc la sensibilité de la clé en utilisant pour chiffrer une même image originale une clé K_1 et ses versions modifiées à 10^{-15} décimale K_2 et K_3. Ensuite le pourcentage de différence entre les pixels des images chiffrées ainsi que leurs coefficients de corrélations sont calculés. Les bonnes valeurs de ces deux métriques sont respectivement 99.60 et 0.001.

2.4.3 Analyse de l'attaque différentielle

Une propriété absolument nécessaire à un bon cryptosystème est la haute sensibilité aux petits changements dans l'image originale. Un adversaire peut faire une légère modification (un seul pixel par exemple) de l'image cryptée, et il observe le résultat de ce changement. De cette façon, il peut être en mesure de trouver une relation significative entre l'image claire et celle cryptée. Si un changement mineur dans l'image en clair peut provoquer un changement significatif dans l'image cryptée à l'égard de la diffusion et la confusion, alors l'attaque différentielle devient inutile.

Pour calculer l'influence du changement d'un seul pixel sur une image cryptée et évaluer ainsi la résistance du cryptosystème aux attaques différentielles, deux critères sont en général utilisées : le critère NPCR (taux de changement du nombre de pixels, en anglais «(Number of Pixel Change Rate») ; et le critère UACI (moyenne unifiée du changement d'intensité, en anglais «Unified Average change intensity») définies par les formules suivantes :

$$NPCR = \frac{\sum_{i,j} D(i,j)}{W \times H} \times 100\% \tag{2.43}$$

$$UACI = \frac{1}{W \times H}\left[\sum_{i,j} \frac{\left|C_1(i,j) - C_2(i,j)\right|}{255}\right] \times 100\% \tag{2.44}$$

C_1 et C_2 sont deux images de taille $W \times H$. Si $C_1(i,j) \neq C_2(i,j)$ alors $D(i,j) = 1$, sinon $D(i,j) = 0$.

Le *NPCR* mesure le pourcentage du nombre de pixels différents par rapport au nombre total de pixels entre deux images. Tandis que l'*UACI* mesure la moyenne de différence d'intensité entre les deux images. Un score *NPCR/UACI* élevé se traduit, généralement, par une forte résistance aux attaques différentielles. Les bonnes valeurs de *NCPR* et de *UACI* sont respectivement 99.60 et 33.45 [12-40].

2.4.4 Cryptanalyses de base

Un algorithme de chiffrement doit réussir les deux tests fondamentaux de cryptanalyse que sont : l'attaque à image claire choisie et l'attaque à image chiffrée choisie.

a) L'attaque à texte clair choisi

Cette attaque repose sur le principe que deux images totalement différentes chiffrées avec la même clé produira deux cryptogrammes différents, mais obtenus avec des sous-clés ou clés dynamiques identiques. Ainsi, si un intrus dispose d'une image chiffrée C et d'une paire d'images claire/ chiffrée correspondante (M, D), il peut retrouver l'image claire P de C en utilisant l'équation (2.45).

$$P^{i,j} = C^{i,j} \oplus \left(M^{i,j} \oplus D^{i,j} \right) \tag{2.45}$$

En général, l'image claire M est l'image tout-zéro (valeur de pixels uniquement des zéros) ou tout-un (valeur de pixels uniquement 255), et l'image chiffrée D est son équivalent chiffré.

b) L'attaque à texte chiffré choisi

Ici l'intrus dispose d'une paire d'images chiffrée / déchiffrée correspondante (D, M), il veut alors retrouver l'image clair P d'une image chiffrée C, pour cela il utilise l'équation suivante :

$$P^{i,j} = C^{i,j} \oplus \left(D^{i,j} \oplus M^{i,j} \right) \tag{2.46}$$

Généralement, l'intrus considère l'image tout-zéro ou tout-un comme chiffrée D et le déchiffre (M), ensuite il utilise les deux pour son attaque.

2.4.5 Temps d'exécution de l'algorithme

L'un des plus importants objectifs de la cryptographie d'image basée sur les systèmes aléatoires c'est la vitesse ou le temps mis dans le processus chiffrement/déchiffrement. Le constant développement en multimédia et son nombre croissant d'utilisateur sont des facteurs nécessitant une communication rapide et sécurisée. Autrement, il serait impossible d'échanger les données à travers le réseau internet à cause des lenteurs des logiciels de cryptographie. Plus rapide est un algorithme de chiffrement, plus rapide sera son logiciel. Bien des chercheurs considèrent la rapidité d'un algorithme comme l'un des critères de sa validation. Cependant, il n'y a pas encore

de standard ou de norme relative au temps minimum de chiffrement /déchiffrement d'une certaine dimension d'image, parce que ce temps dépend du processeur de l'ordinateur utilisé. Quoiqu'il en soit un nouvel algorithme doit indiquer : la plateforme du logiciel de développement, les caractéristiques de l'ordinateur utilisé (Système d'exploitation, Fréquence processeur, la taille de la RAM) et le temps de chiffrement/déchiffrement. La mesure du temps mis pour le chiffrement/déchiffrement est évaluée sur la plateforme considérée en utilisant la commande adéquate (tic et toc pour matlab).

Conclusion

Après avoir étudié les méthodes de cryptographie utilisées, nous avons développé dans ce chapitre nos propres méthodes utilisées en cryptographie d'images, et basées sur la récurrence logistique principalement. L'une des méthodes réalisées produit de nouvelles récurrences GNPA aux excellentes propriétés et les exploite en masquage-brouillage, l'autre renouvelle continuellement la clé de chiffrement. Les métriques également étudiées comme le NCPR, UACI, vont nous permettre d'avoir une base commune d'évaluation et de comparaison avec nos pairs.

Chapitre 3: Résultats et évaluation des algorithmes de chiffrement proposés

Introduction

Dans ce chapitre nous nous appliquons à évaluer profondément les algorithmes proposés pour le chiffrement d'image à base de récurrence logistique. Pour cela, nous allons d'abord présenter une base de données d'images que nous utiliserons dans la suite comme cryptogramme après chiffrement. Ces cryptogrammes subiront tous les tests d'analyses statiques, d'analyses différentielles, de temps de chiffrement/déchiffrement et de cryptanalyses, enfin de certifier la robustesse et la rapidité de nos cryptosystèmes.

3.1 Présentation de la base de données d'images

La base de données d'images utilisées pour évaluer la qualité d'un algorithme de chiffrement reposant sur les récurrences GNPA, est constituée de trois principaux types d'images de tous les formats et de toutes les tailles. On y rencontre donc les images de type binaire, les images en niveau de gris et les images couleurs. Une base de données d'images unique permet de comparer facilement les effets et les résultats obtenus entre différents cryptosystèmes.

3.1.1 Les images binaires

Les images binaires sont des images contenant seulement deux types de valeurs de pixel dont la valeur 0 pour le noir, et la valeur 255 pour le blanc. Dans le traitement d'image, ce type de valeur renvoie soit au 0 (pour 0) soit au 1 (pour 255), c'est pourquoi on parle alors d'image binaire. Les images binaires sont très importantes en cryptographie d'image, ils permettent de constater un chiffrement effectif au niveau visuel de l'image chiffrée obtenue, et au niveau de son histogramme. Cela est un test nécessaire car les images binaires sont très difficiles à crypter, de plus on les utilise en cryptanalyse pour révéler les sous-clés.

Figure 3.1 *Les images binaires ; à gauche l'image fingerprint ; au milieu l'image flower ; et à droite l'image tout-zéros.*

3.1.2 Les images grises (en niveau de gris)

Les images en niveau de gris sont celles dont les intensités lumineuses des pixels sont codées en valeur entière allant de la valeur 0 à la valeur 255. Elles sont plus faciles à chiffrer étant donné qu'elles sont représentées par une seule matrice image dans la plupart des formats. On dénombre bon nombre d'images en niveau de gris utilisées en cryptographie d'image basée sur les systèmes aléatoires (figure 3.2). Les plus célèbres sont : Lena (utilisé dans 99% des articles), cameraman, Baboon (connu aussi sous le nom de mandrill), Barbara, boat, Airport, Pepper, Goldhill, fruit, plane.

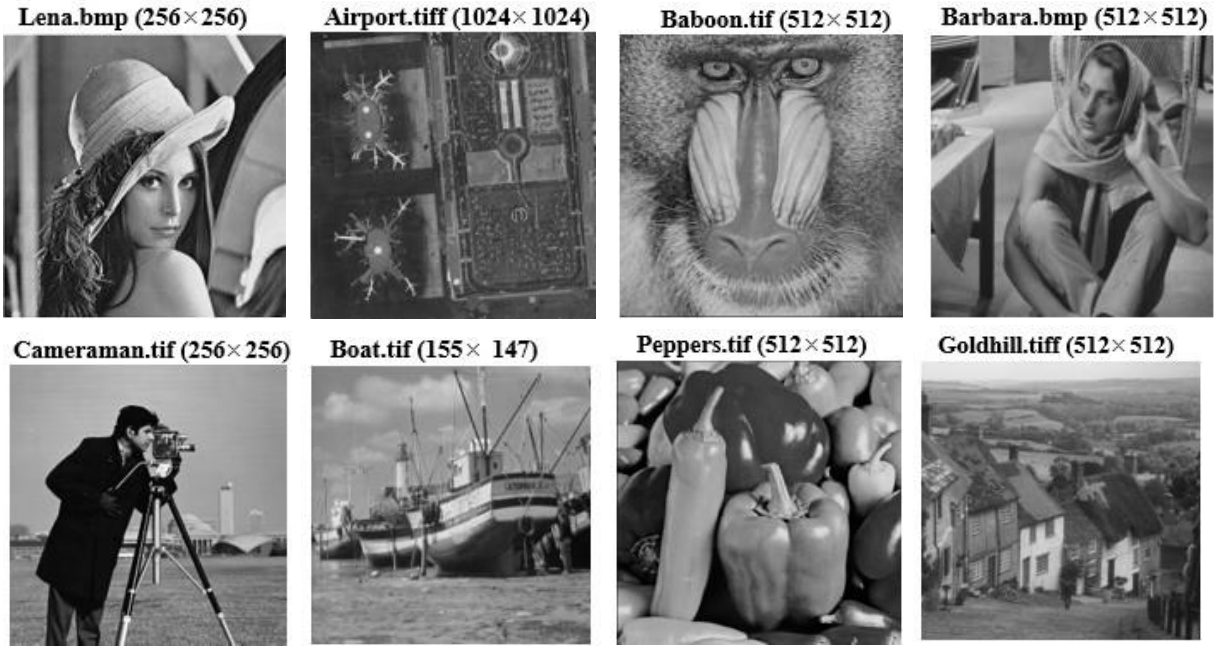


Figure 3.2 *Images en niveau de gris les plus utilisées en cryptographie d'images basée sur les récurrences GNPA.*

3.1.3 Les images couleurs

Le chiffrement des images couleurs constituent également un défi pour la cryptographie d'images. En effet, chiffrer une image couleur revient à chiffrer 3 images en niveau de gris, en un processus. Ainsi donc, les images codées en RVB ou en 3 couches de niveau de gris consomment énormément de ressources mémoires et temps. Leurs évaluations sont faites en utilisant des images visuellement identiques aux images à niveau gris, mais codées en RVB (figure 3.3).

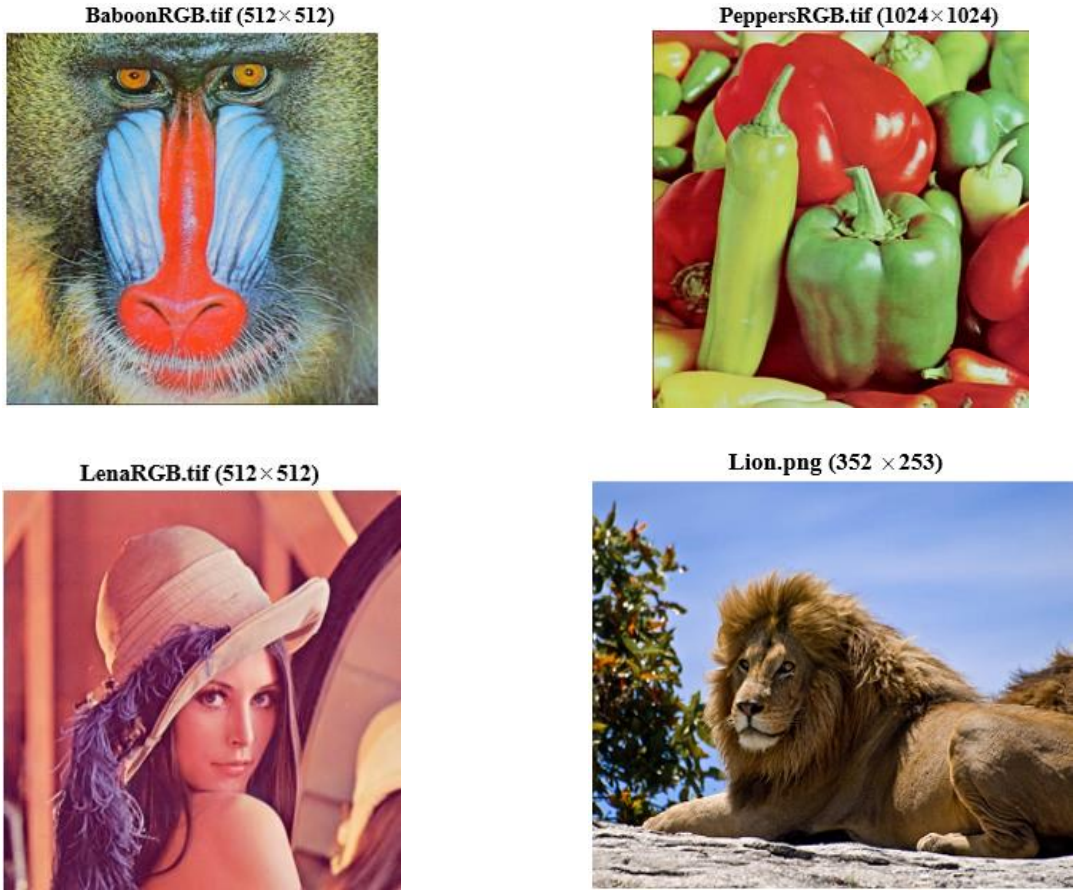


Figure 3.3 *: Les images couleurs les plus utilisées en cryptographie d'images basée sur les récurrences GNPA.*

3.2 Analyse de sécurité de « un algorithme de chiffrement d'image basé sur la technique de substitution et le mixage des récurrences GNPA »

Les tests de sécurités ont été exécutés avec un ordinateur ayant un processeur Core(TM) i5-2430M, sur la plateforme Matlab 2012b. Plusieurs images ont été chiffrées

et analysées aux attaques statistiques, à l'attaque de force brute, à l'attaque différentielle, à l'attaque de texte clair et de texte chiffré choisi, et en terme de rapidité d'exécution.

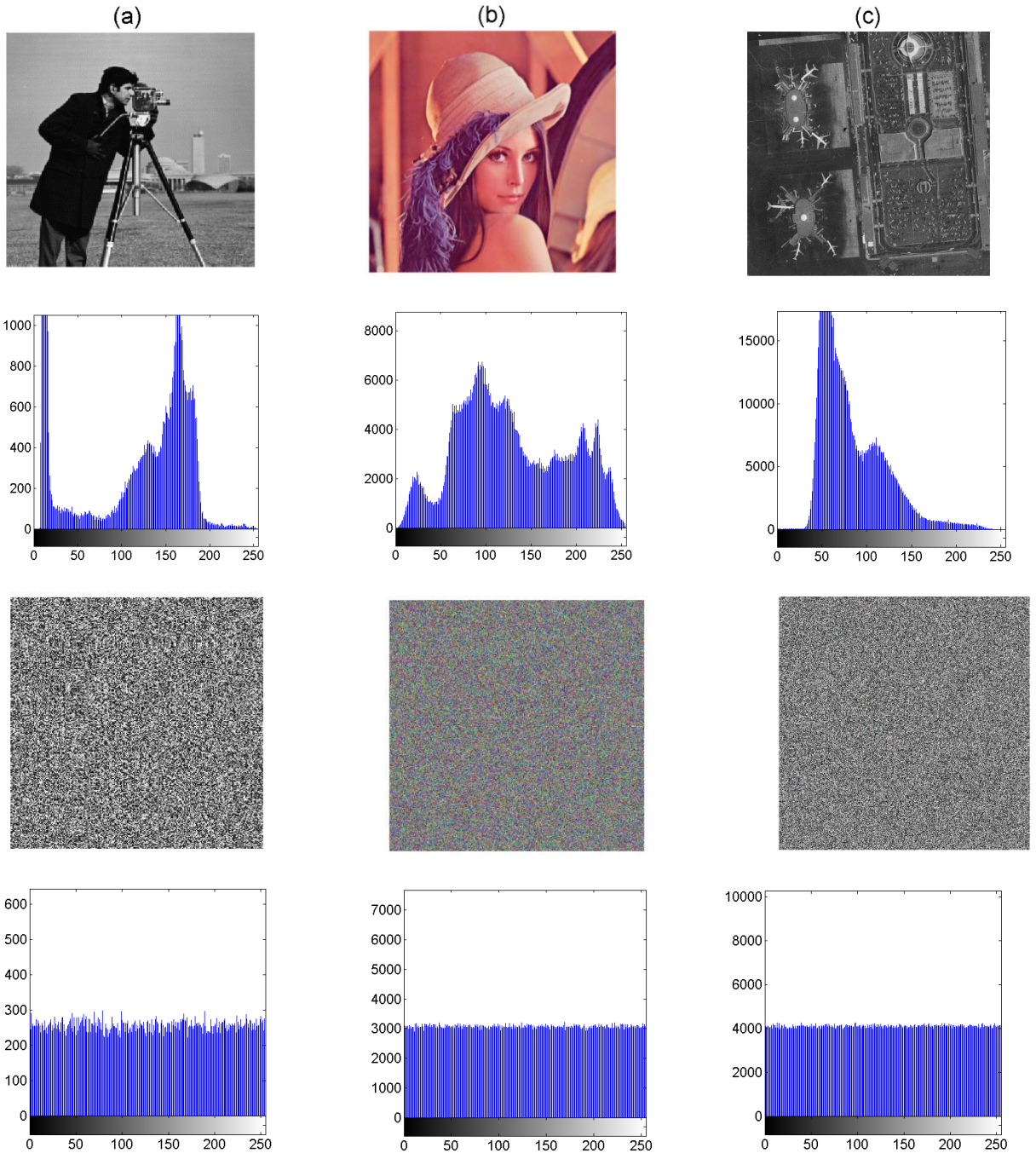


Figure 3.4 *Images en niveau de gris et en couleur ; claires, chiffrées; ainsi que leurs histogrammes respectifs. (a) Cameraman, (b) Lena, (c) Airport.*

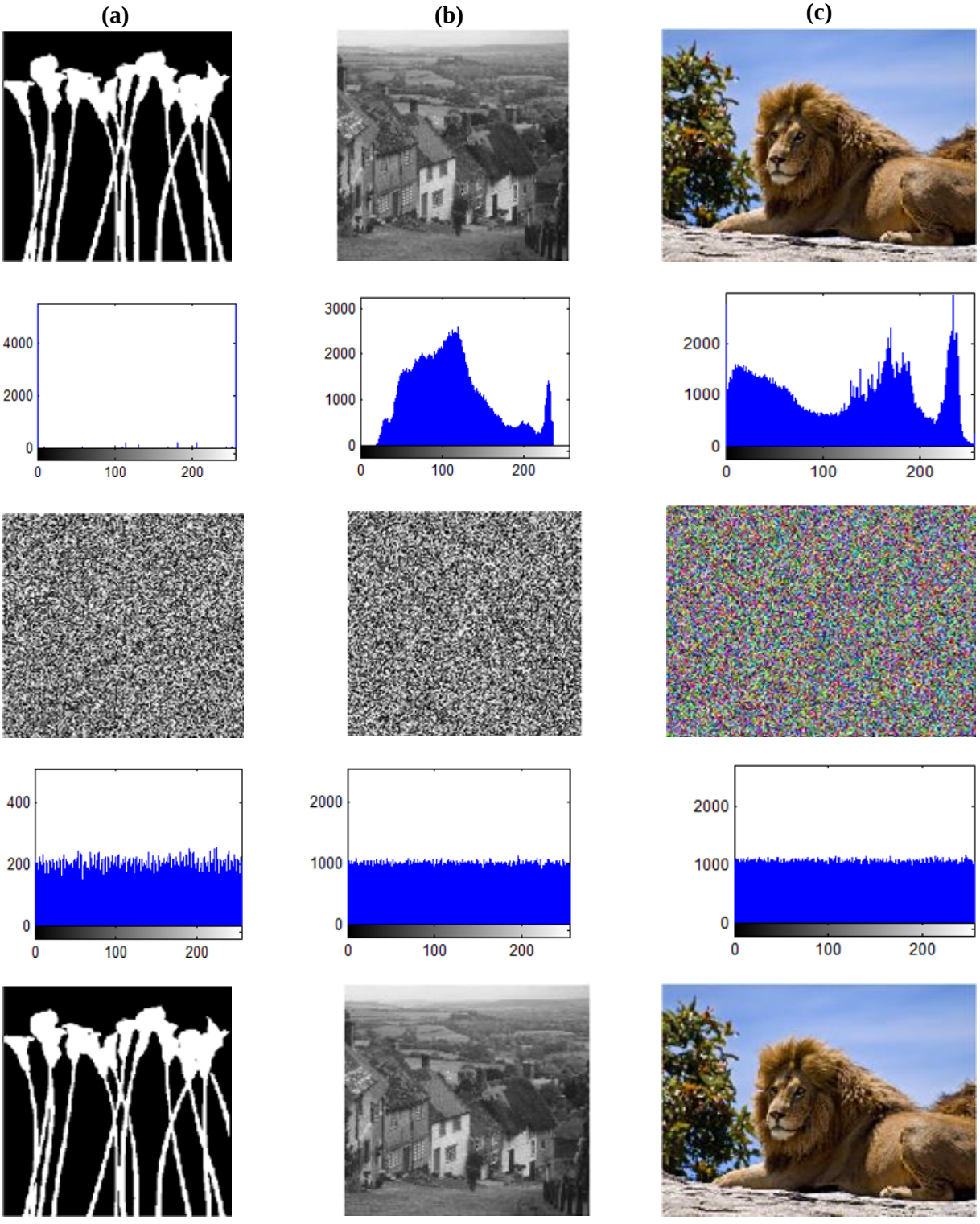


Figure 3.5 *Images en niveau de gris et en couleur ; claires, chiffrées et déchiffrées ; ainsi que leurs histogrammes respectifs. (a) Flowers, (b) Goldhill, (c) Lion.*

3.2.1 Histogramme et la variance d'histogramme

a) L'histogramme

Les figures 3.4 et 3.5 nous permettent de constater que les images de cameraman, Lena, airport, flour, et fingerprint chiffrées sont confuses, brouillées et semblable à un bruit blanc. Cette appréciation est d'avantage confirmée par leurs histogrammes qui affichent une distribution quasi-uniforme des pixels aux différents niveaux de gris. Pour confirmer nos observations, nous allons calculer la variance d'histogramme.

b) La variance d'histogramme

La variance d'histogramme des différentes images chiffrées a été calculée grâce à l'équation (2.40), et a été résumée dans le tableau 3.1. Les valeurs obtenues ont été comparées à ceux des algorithmes de chiffrements récents. Nous avons ainsi des valeurs comprises entre 1000 et 5482, elles sont bien inférieures à celles proposées en [110], et ont une moyenne inférieure à 5000. L'analyse d'histogramme de notre algorithme de chiffrement ne présente aucune faille de sécurité.

Tableau 3.1 *Variance d'histogramme de quelques images chiffrées*

Nom de l'image	Image en clair	Notre image chiffrée	**[110]**	**[111]**	**[112]**
Cameraman	1673908.95	1482.61	5381.26	953.20	946.68
Lena	638716.84	1450.87	5118.09	1068.10	1027.59
Airport	1567841.23	5471.65	5264.70	900.12	1103.156
Peppers	548714.85	1536.31	5403.28	1033.78	941.06
Baboon	745278.25	4625.36	5119.29	965.63	1058.13
Boat	2821212.85	1334.22	5097.98	973.78	926.91

3.2.2 Analyse de la corrélation des pixels adjacents

Il est absolument nécessaire d'avoir une très faible corrélation entre pixels voisins dans un cryptogramme. Pour démontrer la validé de ce test, nous avons calculé le

coefficient de corrélation Cr de nos images chiffrées dans les directions : horizontal (HC), vertical (VC), et diagonal (DC), en utilisant l'équation (2.41).

Les résultats obtenus suite à ses calculs, et consignés dans le tableau 3.2, sont bien tous inférieur à 0.01. La valeur moyenne de coefficient de corrélation par image et dans toutes les directions confondues vaut 0.006.

Une meilleure observation est faite en scrutant les graphes de la figure 3.6. Ils révèlent bien comment les valeurs de gris concentrées autour de certaines valeurs (le long d'une ligne) pour toutes directions de l'image claire, sont dispersées presque uniformément dans les graphes des mêmes directions pour les images chiffrées. L'algorithme proposé ici satisfait à la condition de sécurité permettant d'éviter les attaques de corrélation.

Tableau 3.2 *Coefficient de corrélation de quelques images.*

Image	Test	Cr image en clair	Cr image chiffrée
Cameraman	HC	0.9377	-0.009
	VC	0.9535	0.010
	DC	0.9043	-0.006
Lena	HC	0.9679	0.001
	VC	0.9845	-0.014
	DC	0.9580	-0.006
Airport	HC	0.9090	0.002
	VC	0.8989	-0.004
	DC	0.8610	0.008
Peppers	HC	0.9756	0.005
	VC	0.9736	-0.004
	DC	0.9575	0.011
Baboon	HC	0.9090	0.013
	VC	0.8989	-0.005
	DC	0.8610	0.007
Boat	HC	0.9389	0.008
	VC	0.9425	-0.004
	DC	0.9070	0.008
Flower	HC	0.9989	0.01
	VC	0.9825	0.009
	DC	0.9570	-0.006

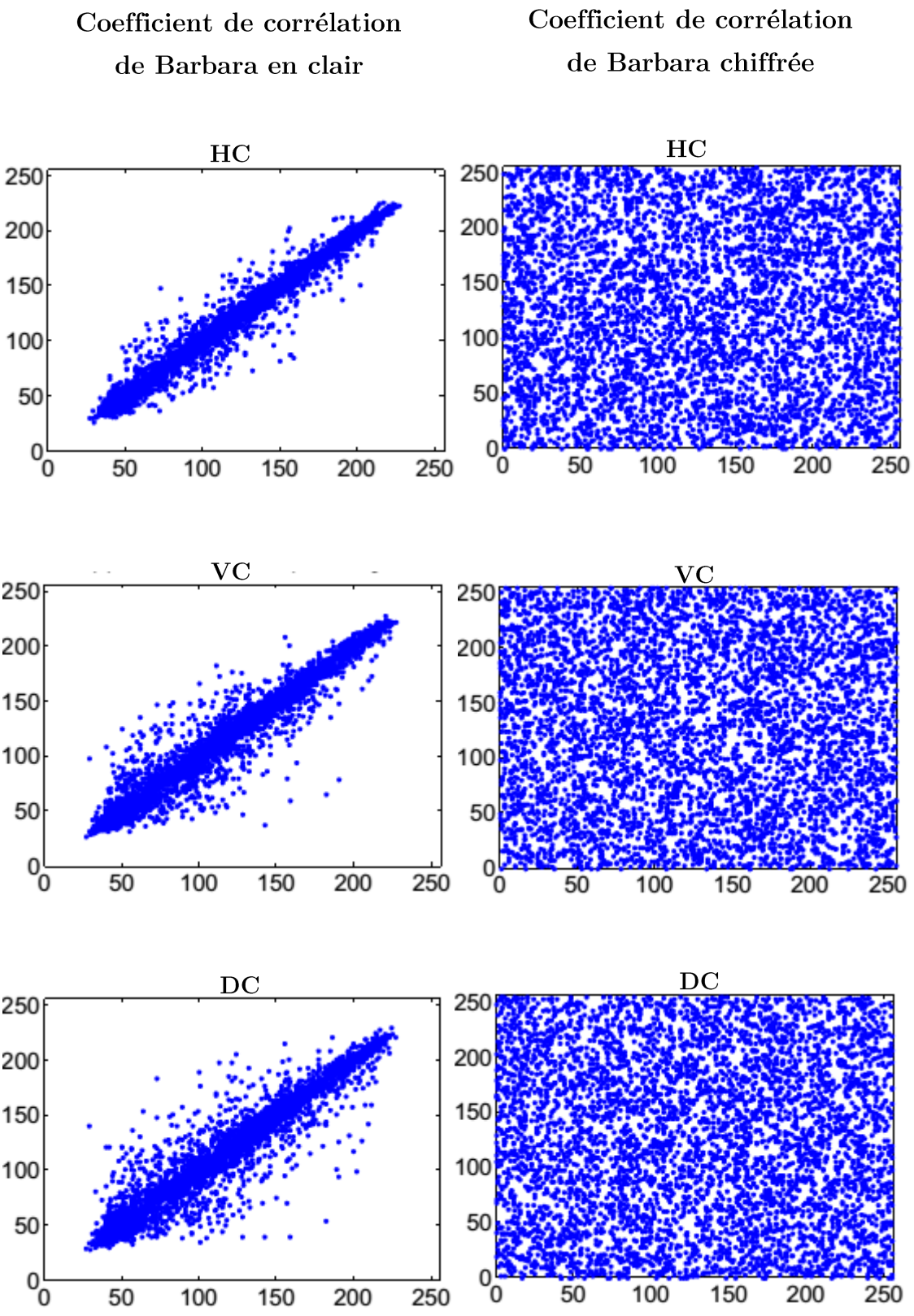


Figure 3.6 *Graphe des coefficients de corrélations horizontale (HC), verticale (VC) et diagonale (DC) de l'image Barbara et de sa version chiffrée.*

3.2.3 Analyse de l'entropie de Shannon

L'entropie de Shannon (ou entropie de l'information) de nos images chiffrées a été évaluée au moyen de l'équation (2.42), et les valeurs obtenues ont été consignées dans le tableau 3.3 et 3.4. Les valeurs obtenues oscillent entre 7,9971 et 7.9998 variant selon l'image, et sont bien supérieures à celles proposées en littérature ([52], [113]). Ces valeurs très proches de 8, nous suggèrent fortement que : quelques soit le type d'image chiffrée (niveau de gris, couleur ou taille), un état de désordre règne dans sa codification en binaire.

Tableau 3.3*: Entropie de l'information de quelques images en niveau de gris et de leurs versions chiffrées*

Image en niveau de gris	Image en clair	Image chiffrée	[113]	[52]	[112]
Cameraman	6.9719	7.9971	7.9969	7.9985	7.9971
Lena	7.5925	7.9994	7.9969	7.9963	7.9994
Baboon	7.4125	7.9993	7.9974	-	7.9992
Airport	7.0896	7.9998	-	7.9960	7.9992
Barbara	7.4562	7.9994	-	7.9978	7.9991
Boat	7.1701	7.9993	7.9973	7.9980	7.9994
Flower	3.256	7.9991	-	-	-
black	0	7.9992	6.9980	-	-

Tableau 3.4*: Entropie de l'information de quelques images couleurs et de leurs versions chiffrées.*

Image en couleur	type	composants		
		R	V	B
Lena	Claire	7.5232	7.2699	7.1985
	Chiffrée	7.9994	7.9991	7.9994
Baboon	Claire	6.9855	7.2584	6.8925
	Chiffrée	7.9994	7.9992	7.9993
Peppers	Claire	6.9872	7.6942	7.6543
	Chiffrée	7.9998	7.9999	7.9997

3.2.4 Analyse de la clé de chiffrement

a) L'espace de clé

La clé secrète de type interne proposé par cet algorithme est constituée de : 4 conditions initiales (w_0, x_0, y_0, z_0), 5 paramètres de contrôles (r_1, r_2, r_3, r_4, α), 5 valeurs de 8 bits ($C(1,1)$, $C(1,2)$, $C(1,3)$, $C(2,1)$, $C(3,1)$), donnant un total d'espace de clé de $(10^{15})^4 \times (10^{15})^5 \times (2^8)^5 = 10^{142} \approx 2^{475}$ si la précision décimale est fixée à 10^{-15}. Cette espace de clé est 10^{112} supérieurs à la valeur d'espace de clé validé comme étant sûr [12-35], donc l'attaque à force brute n'est pas le moyen par lequel cette algorithme de chiffrement sera faillible.

b) La sensibilité de la clé

Lorsqu'un algorithme n'est suffisamment sensible à la moindre variation de la clé, il commence à apparaître des clés équivalentes. Ces clés sont différentes mais une peut déchiffrer le cryptogramme de l'autre, en d'autres termes elles produisent le même cryptogramme lors du chiffrement de la même image. Il devient alors très aisé de mener une attaque à image claire ou à image chiffrée choisie. Ainsi donc, nous avons utilisé pour chiffrer une même image les clés : $K_1 = r_1, r_2, r_3, r_4, \alpha, w_0, x_0, y_0, z_0$ et ses versions modifiées à 10^{-15} près K_2 ($r_2 = r_2 + 10^{-15}$ pour K_2, le reste inchangé) et K_3 ($z_0 = z_0 + 10^{-15}$ for K_3, le reste inchangé). Ensuite, le pourcentage de différence entre images chiffrées (entre leurs pixels) des trois clé est calculé. Les résultats de ce calcul, rapportés dans le tableau 3.5, nous prouvent que les clés K_1, K_2 et K_3 sont différentes les unes des autres car leurs images chiffrées sont différentes les unes des autres d'au moins 99.62%.

Par ailleurs, les images chiffrées Airport, BaboonRGB avec la clé K_1 et déchiffrées avec les clés K_2 et K_3 et représentées à la figure 3.6 et 3.7 restent toujours chiffrées. L'ensemble de ces tests nous démontrent sans ambiguïté que la clé de ce cryptosytème est extrêmement sensible.

Tableau 3.5 *Pourcentages de différences entre images chiffrées avec des clés presque similaires.*

	Différence entre Clés			Coefficient de corrélation entre cryptogramme		
Images	K_1 vs K_2	K_2 vs K_3	K_1 vs K_3	K_1 vs K_2	K_1 vs K_2	K_1 vs K_2
Lena	99.61	99.65	99.58	0.9×10^{-5}	2.2×10^{-5}	5.0×10^{-5}
Peppers	99.62	99.78	99.65	0.1×10^{-4}	7.3×10^{-6}	0.1×10^{-4}
Flower	99.65	99.67	99.77	0.2×10^{-5}	3×10^{-5}	1×10^{-5}

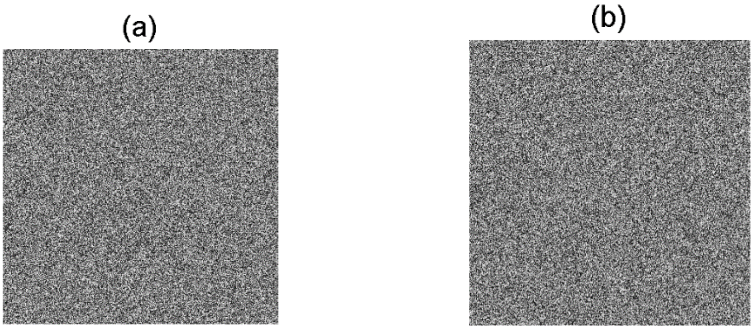


Figure 3.7 *Tentative de déchiffrement du cryptogramme Airport avec des clés légèrement différentes de K_1 : (a) déchiffrement avec K_2, (b) déchiffrement avec K_3.*

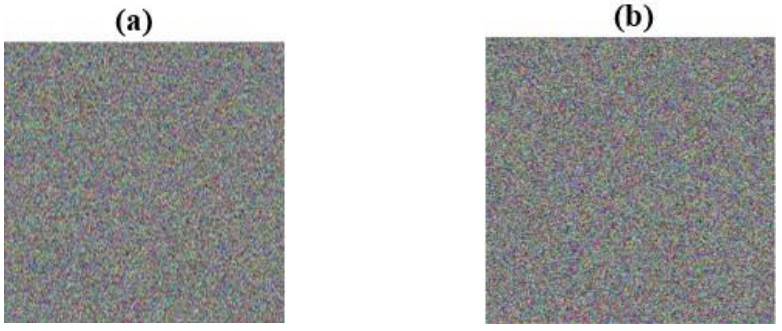


Figure 3.8 *Tentative de déchiffrement du cryptogramme BaboonRGB avec des clés légèrement différentes de K_1 : (a) déchiffrement avec K_2, (b) déchiffrement avec K_3.*

3.2.5 Analyse de l'attaque différentielle

Nous avons évalué la sensibilité de l'algorithme proposé au moindre changement d'une valeur de pixel par la mesure de ses critères *NCPR* et *UACI* (Equation (2.43);

(2.44)). Pour un test d'une extrême précision, nous avons généré pour chaque image à chiffrer trois images jumelles, différentes de l'originale respectivement par la modification du bit le plus insignifiant (LSB) sur le premier pixel, le pixel médian et le dernier pixel.

Le tableau 3.6-3.7 contient les résultats de *NCPR* et de *UACI* obtenus, le minimum de leurs valeurs pour toutes les images testées sont respectivement de 99.61 et 33.52. On rappelle que ces valeurs sont obtenues avec des couples d'images chiffrées dont les originaux sont différents d'un seul bit seulement. Par ailleurs, nos valeurs (Tableau 3.6-3.7) sont équivalentes sinon légèrement meilleures que celles requises en littérature et validées pour les cryptosystèmes [114, 115]. Considérant les résultats obtenus, aucune attaque différentielle ne pourra réussir.

Tableau 3.6 *Evaluation des critères NPCR et UACI de quelques images en niveau gris.*

		Modification du bit le plus insignifiant (LSB) sur :		
Image	Test	Le premier Pixel	le Pixel du milieu	Le dernier Pixel
Cameraman	NCPR	99.63	99.64	99.62
	UACI	33.55	33.56	33.52
Lena	NCPR	99.65	99.66	99.61
	UACI	33.49	33.50	33.46
Airport	NCPR	99.62	99.68	99.62
	UACI	33.47	33.50	33.47
Flower	NCPR	99.61	99.60	99.63
	UACI	33.50	33.51	33.56
Barbara	NCPR	99.60	99.68	99.66
	UACI	33.49	33.56	33.47
Baboon	NCPR	99.59	99.69	99.63
	UACI	33.45	33.46	33.48
Goldhill	NCPR	99.68	99.60	99.59
	UACI	33.59	33.56	33.49
Peppers	NCPR	99.61	99.65	99.65
	UACI	33.45	33.53	33.56

Tableau 3.7 *Evaluation des critères NPCR et UACI de l'image couleur Lena.*

Composant De l'image Lena	Test	Notre algorithme	[114]	[115]
R	NCPR	99.63	99.59	99.63
	UACI	33.52	33.33	33.31
G	NCPR	99.61	99.62	99.60
	UACI	33.55	33.35	33.34
B	NCPR	99.64	99.63	99.61
	UACI	33.45	33.12	33.43

3.2.6 Analyse du temps de chiffrement/déchiffrement

Une évaluation du temps mis pour le chiffrement/ déchiffrement nous est donnée dans le tableau 3.8. Dans ce même tableau on retrouve le temps de chiffrement de quelques algorithmes de chiffrements classés comme rapide. Il apparait très clairement que notre algorithme possède un meilleur temps de chiffrement/ déchiffrement (que ceux de [52, 112-118]) avec une moyenne de 490 ms pour une image 512×512.

Tableau 3.8 *Durée de chiffrement/ déchiffrement.*

Image	Type	Notre algorithme	[52]	[73]	[116]	[117]	[115]	[118]
Cameraman	Gris	0.195	0.223	0.178	1.673	-	-	-
Lena	Gris	0.490	-	0.663	-	-	-	-
Airport	Gris	0.997	-	3.142	-	-	-	-
Lena	Couleur	0.610	-	-	-	-	-	-

3.2.7 Bilan des évaluations, comparaisons et discussion

La totalité des performances de notre algorithme est comparée dans le tableau 3.9 à ceux de certains articles récents. L'image couleur Lena de 512×512 a été choisi comme standard de comparaison, et le temps a été évalué sur une plateforme visual C++ 2010 enfin de simuler les conditions réelles d'applications. Les résultats obtenus dans le tableau 3.9 démontre bien que les métriques comme l'espace de clé, le *NCPR*, l'*UACI*, le temps

de chiffrement/ déchiffrement sont bien meilleurs. Par ailleurs les métriques comme le coefficient de corrélation, l'entropie de Shannon, la sensibilité de la clé ont des valeurs standards.

Tableau 3.9 *Comparaison de notre algorithme à quelques-uns en littérature.*

Tests	Cryptosystème proposé	[82]	[79]	[119]
Espace de clé	$\mathbf{10^{142}}$	10^{96}	10^{143}	10^{42}
Sensibilité de la clé	99.66	-	-	99,61
Coefficient de correlation moyen	0.004	-0.005	0.003	0.004
Entropie de l'information	**7.9994**	7.999	7.9994	7.9993
NCPR	**99.62**	99.58	99.60	99.59
UACI	33.63	33.25	33.50	33.47
Durée de chiffrement en (s)	**0.099**	0.174 (4round)	0.105 (4round)	0.101

3.3. Analyse de sécurité de « utilisation adéquate de la récurrence logistique dans un chiffrement à flot auto-synchronisé pour le chiffrement d'image »

3.3.1 Histogramme et variance d'histogramme

a) L'histogramme

La figure 3.9 montre les images en clair de cameraman, Peppers couleur, Airport, black, leurs équivalents chiffrés, ainsi que leurs histogrammes.

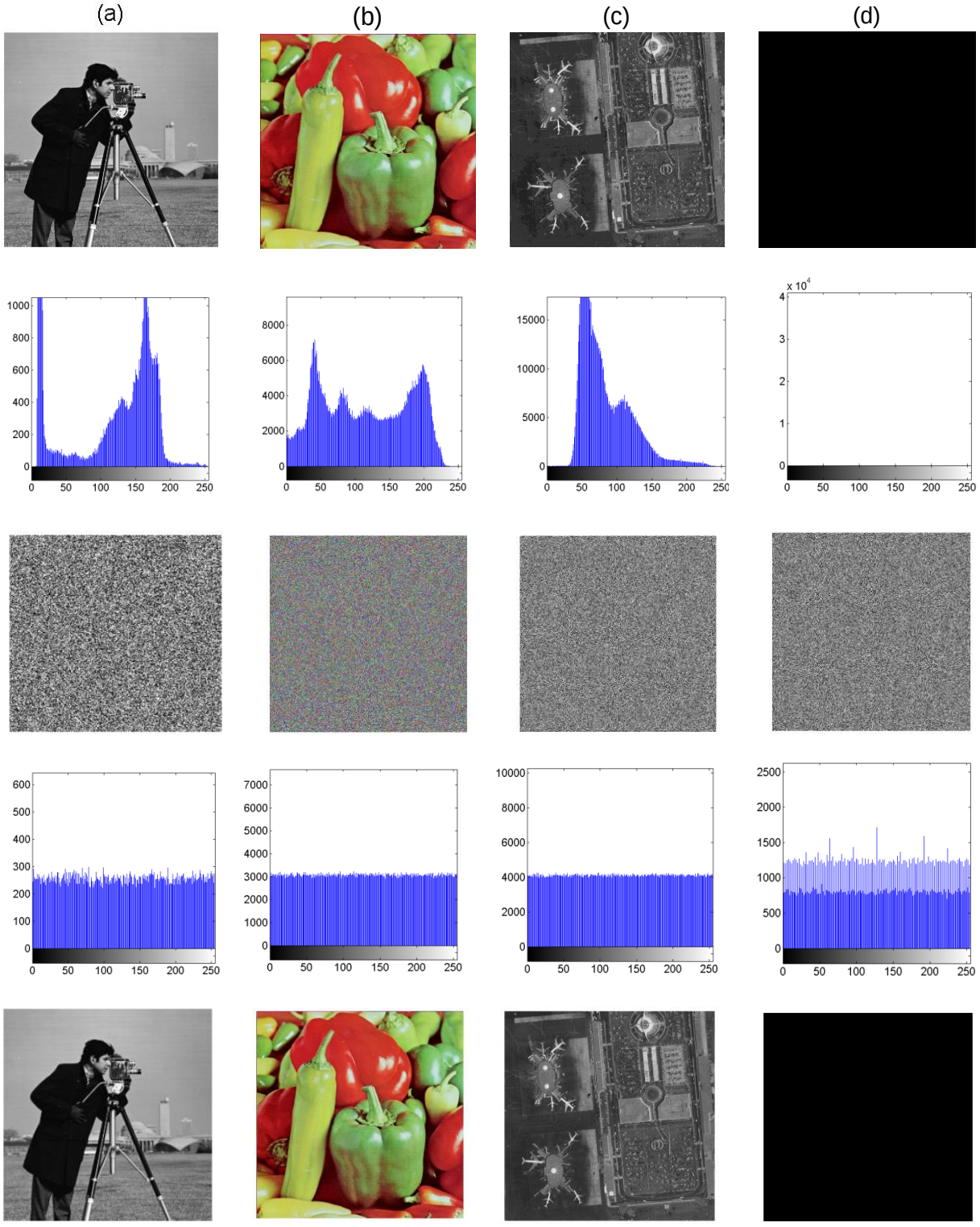


Figure 3.9 *Image en niveau de gris et couleur, claires, chiffrées, déchiffrées et leurs histogrammes : (a) Cameraman, (b) Peppers, (c) Airport, (d) Black.*

On peut clairement observer les histogrammes des images chiffrées complètement uniforme au niveau de l'étalement des pixels par niveau de gris. Cela constitue un très

bon indicateur de chiffrement que nous allons confirmer au moyen de la variance d'histogramme.

b) La variance d'histogramme

Le tableau 3.10 présente la variance d'histogramme de quelques images et leurs versions chiffrées. On peut y constater que celles des images chiffrées sont faibles et sont de l'ordre de 5000, alors que celles des images en clair sont bien plus élevées (ordre du million). Des valeurs de l'ordre de 5000 ou inférieurs sont en accord avec les standards de la littérature [108, 122], et permettent d'assurer que la nature de la distribution des pixels par intensités lumineux des images chiffrées, est aléatoire.

Tableau 3.10 *Variance d'histogramme de quelques images chiffrées*

Nom de l'image	Image en clair	Notre image chiffrée 2ième contribution	Notre image chiffrée 1ère contribution
Cameraman	1673908.95	2393.61	1482.61
Lena	638716.84	961.42	1450.87
Airport	1567841.23	4551.73	5471.65
Peppers	548714.85	847.23	1536.31
Baboon	745278.25	1535.47	4625.36
Barbara	612365.23	3654.28	3654.28
Goldhill	1602343.50	2175. 34	1235.74
Boat	2821212.85	3124. 32	1334.22
Fingerprint	6708965.14	3400.23	1004.33

3.3.2 Analyse du coefficient de corrélation des pixels adjacents

Pour chaque image testée, nous avons aléatoirement choisi 5000 paires de couple de pixels adjacents dans les trois directions HC (horizontal), VC (vertical), DC (diagonal), ensuite nous avons calculé leurs coefficients de corrélations et présenté les résultats dans le tableau 3.11 . Dans ce tableau on peut voir que les coefficients de

corrélations des images en claires sont très proches de 1 (preuve de la forte corrélation entre pixels adjacents), alors que leurs équivalentes chiffrées ont des coefficients de corrélation tous inférieurs à 0.01 par direction, et ont une moyenne de 0.003 par image. Ce résultat de faible corrélation pour les images chiffrées est d'avantage confirmé par la figure 3.10 qui montre les graphes associées aux coefficients de corrélations calculés. L'ensemble de ces résultats persuasifs constituent une preuve de l'efficacité de notre algorithme à des images chiffrées sans faille de corrélation.

Tableau 3.11 *Coefficients de corrélations de quelques images.*

Image	Test	*Cr* image en clair	*Cr* image chiffrée $2^{ième}$ contribution	*Cr* image chiffrée $1^{ère}$ contribution
Cameraman	HC	0.9377	0.010	-0.009
	VC	0.9535	-0.005	0.010
	DC	0.9043	-0.004	-0.006
Lena	HC	0.9679	-0.009	0.001
	VC	0.9845	-0.004	-0.014
	DC	0.9580	0.009	-0.006
Airport	HC	0.9090	0.010	0.002
	VC	0.8989	0.015	-0.004
	DC	0.8610	0.009	0.008
Peppers	HC	0.9756	-0.010	0.005
	VC	0.9736	-0.001	-0.004
	DC	0.9575	0.011	0.011
Baboon	HC	0.9090	-0.003	0.013
	VC	0.8989	-0.008	-0.005
	DC	0.8610	-0.009	0.007
Boat	HC	0.9389	0.012	0.008
	VC	0.9425	0.007	-0.004
	DC	0.9070	-0.001	0.008
Flower	HC	0.9989	-0.002	0.01
	VC	0.9825	0.006	0.009
	DC	0.9570	0.003	-0.006

Composantes R, G, B

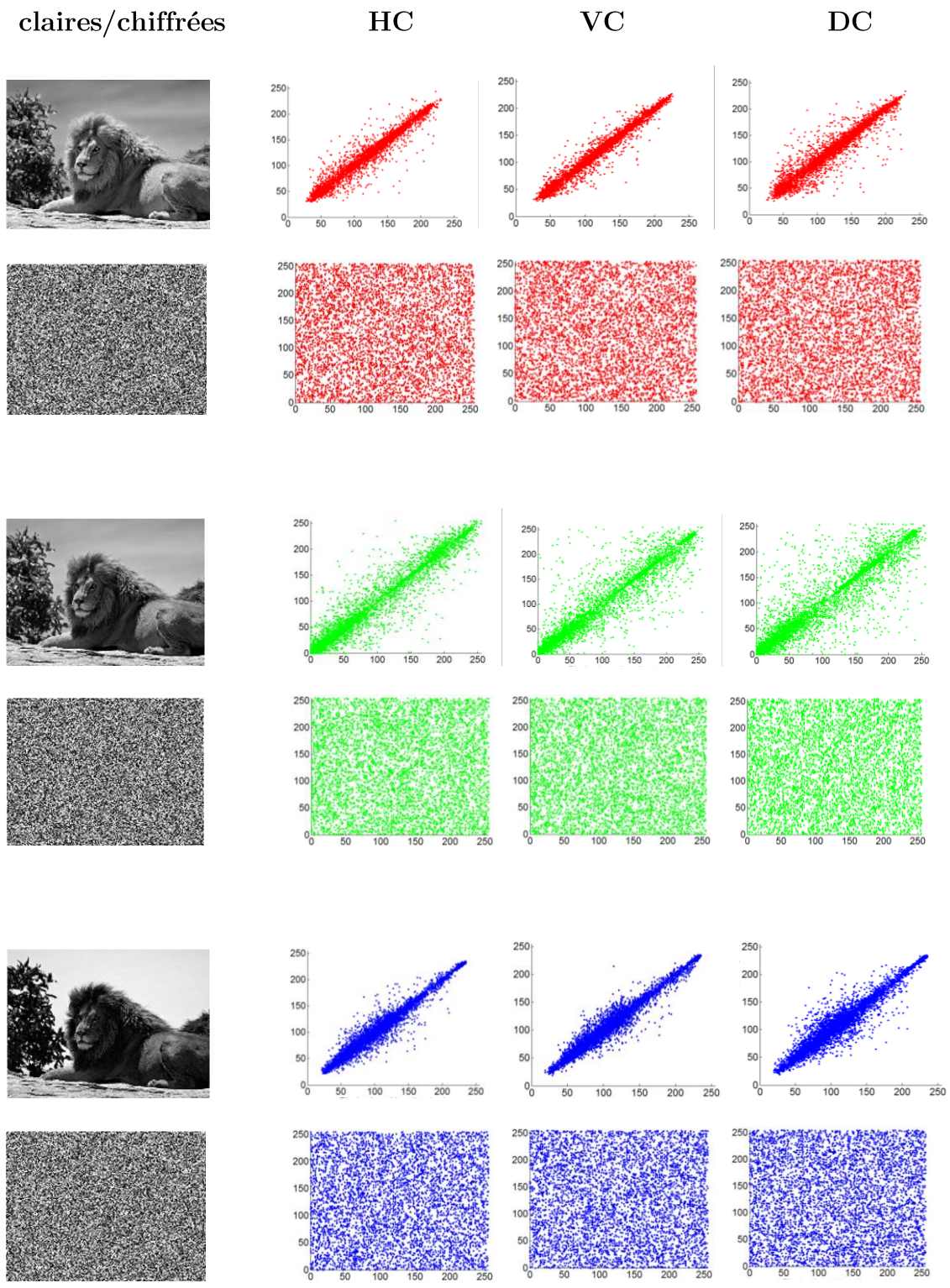


Figure 3.10 *Graphe des coefficients de corrélations horizontale (HC), verticale (VC) et diagonale (DC) des composantes respectives R, G, B de l'image lion et de leurs versions chiffrées.*

3.3.3 Entropie de l'information

Le Tableau 3.12 récapitule les résultats de l'entropie de l'information obtenus pour quelques images et leurs correspondantes chiffrées. La valeur d'entropie d'un excellent désordre dans l'information étant de 8, nos valeurs sont de l'ordre de 7,999 et sont bien meilleures que celles des algorithmes de chiffrements validés en littérature [52, 112]. L'algorithme de chiffrement que nous proposons est donc sûr face à l'attaque d'entropie.

Tableau 3.12*: Entropie de l'information de quelques images en niveau de gris et de leurs versions chiffrées.*

Image en niveau de gris	Image en clair	Image chiffrée $2^{ième}$ contribution	Image chiffrée $1^{ère}$ contribution	[52]	[112]
Cameraman	6.9719	7.9975	7.9971	7.9985	7.9971
Lena	7.5925	7.9994	7.9994	7.9963	7.9994
Baboon	7.4125	7.9994	7.9993	-	7.9992
Airport	7.0896	7.9998	7.9998	7.9960	7.9992
Barbara	7.4562	7.9993	7.9994	7.9978	7.9991
Boat	7.1701	7.9993	7.9993	7.9980	7.9994
black	0	7.9993	7.9992	-	-

3.3.4 Analyse de la clé

a) Analyse de l'espace de clé

Dans l'algorithme proposé, la clé est externe et a une longueur de 256 bits soit $\mathbf{2^{256}} \approx \mathbf{10^{80}}$ comme espace de clé, or actuellement une longueur de $\mathbf{10^{30}}$ est considéré comme sûr [35]. Notre espace de clé est donc suffisante pour résister à une attaque de force brute.

b) Analyse de la sensibilité de la clé

Pour ce test de sensibilité, nous avons chiffré les images Pepper couleur et flower en utilisant les clés secrètes quasi-identiques :

*k_1 "420A2383AE6224C2AACB04A31E74**3**3C028427D49823CB725428BCB42C3F10B**8**"*

*k_2"420A2383AE6224C2AACB04A31E74**3**3C028427D49823CB725428BCB42C3F10B**7**",*

*k_3"420A2383AE6224C2AACB04A31E74**2**3C028427D49823CB725428BCB42C3F10B**8**".*

Ensuite, nous avons évalué le pourcentage de différence entre cryptogramme à clés de chiffrements différents mais à image en clair identique. Le tableau 3.4 révèle les résultats de cette évaluation, elle fait état d'un pourcentage de différence de l'ordre de 99.62 % ; et d'un coefficient de corrélation presque nulle (0.00002) entre les cryptogrammes de clés différentes. En outre, à la figure 3.12, on observe les cryptogrammes des différentes images peppers et flowers obtenues avec la clé K_1 (figure 3.12-(b)) mais déchiffrés avec les clés K_2 (figure 3.12-(c)) et K_3 (figure (d)) sans succès. L'algorithme que nous proposons est extrêmement sensible à un changement de clé.

Tableau 3.13 *Pourcentages de différences entre images chiffrées avec des clés presque similaires.*

	Différence entre Clés			Coefficient de corrélation entre cryptogramme		
Images	K_1 vs K_2	K_2 vs K_3	K_1 vs K_3	K_1 vs K_2	K_1 vs K_2	K_1 vs K_2
Lena	99.61	99.65	99.58	0.9×10^{-5}	2.2×10^{-5}	5.0×10^{-5}
Goldhill	99.52	99.63	99.56	0.8×10^{-6}	0.1×10^{-4}	4×10^{-5}
Peppers	99.63	99.60	99.59	5×10^{-5}	0.2×10^{-5}	6.1×10^{-5}
Black	99.61	99.67	99.63	0.4×10^{-5}	0.6×10^{-6}	1.1×10^{-5}

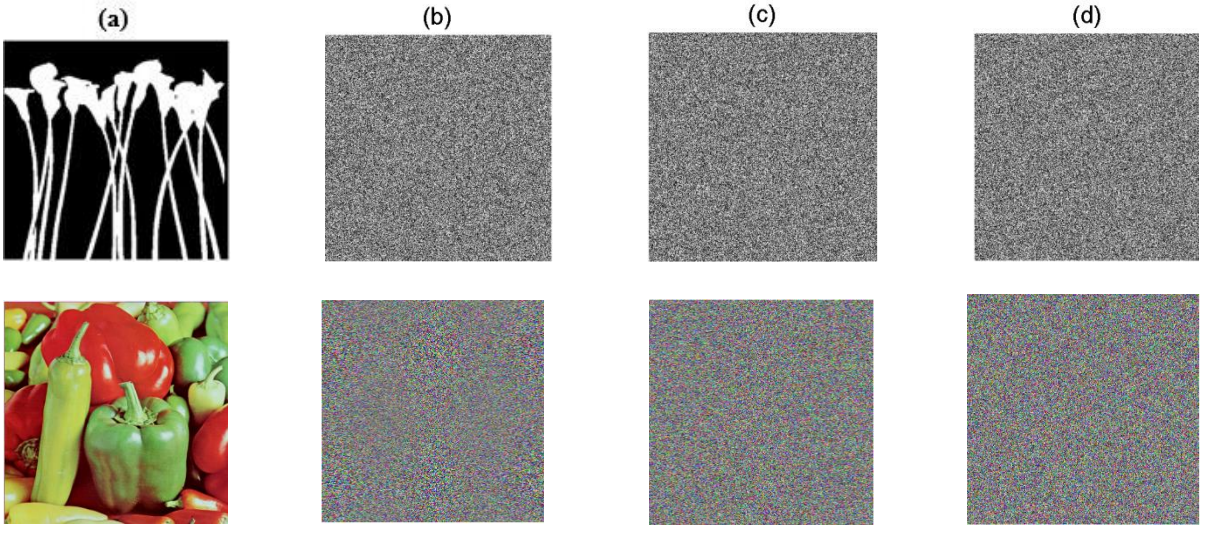


Figure 3.11 *Tentative de déchiffrement des cryptogrammes flower et peppers en (b) avec des clés légèrement différentes de K_1 : (c) déchiffrement avec K_2, (d) déchiffrement avec K_3.*

3.3.5 Analyse de l'attaque différentielle

Nous avons réitéré le procédé de la *section 3.2.5* utilisé pour tester l'influence du bit le plus insignifiant (en début, en milieu, en fin de la matrice image) sur le chiffrement. Les valeurs NPCR et UACI obtenues dans le tableau 3.14 pour les images grises, et le tableau 3.15 pour les images couleurs, sont respectivement d'une moyenne de 99.61 et 33.52. Ces valeurs sont dans l'ordre des valeurs attendues d'un bon cryptosystème et sont comparables à ceux de [120] et [121], par conséquent, une attaque différentielle sera sans effet sur notre algorithme de chiffrement.

Tableau 3.14 *Evaluation des critères NPCR et UACI de quelques images en niveau gris.*

		Modification du bit le plus insignifiant (LSB) de notre algorithme sur :			[120]	[121]
Image	Test	Le premier Pixel	le Pixel du milieu	Le dernier Pixel		
Cameraman	NCPR	99.62	99.66	99.64	99.60	99.62
	UACI	33.59	33.47	33.60	33.49	33.45
Lena	NCPR	99.60	99.63	99.62	99.58	99.59
	UACI	33.58	33.59	33.55	33.46	33.41
Barbara	NCPR	99.65	99.61	99.62	99.60	99.60
	UACI	33.43	33.52	33.57	33.47	33.41
Baboon	NCPR	99.60	99.60	99.63	-	99.61
	UACI	33.55	33.57	33.45	-	33.43
Boat	NCPR	99.61	99.60	99.63	99.62	-
	UACI	33.51	33.49	33.59	33.39	-

3.3.7 Analyse du temps de chiffrement/déchiffrement

Les temps de chiffrement/déchiffrement obtenus pour plusieurs images (Cameraman, Lena couleur et gris, Airport) sont consignés dans le tableau 3.15. Nous avons obtenu un temps moyen de 650 ms (millisecondes) avec une image Lena 512×512 couleurs et un algorithme non-optimisé. Ce temps serait bien meilleur dans un environnement logiciel compte tenu du fait que matlab est une plateforme d'évaluation consommant d'énormes ressources matérielles (processeur et RAM).

Tableau 3.15 *Durées (en seconde) de chiffrement déchiffrement de quelques images.*

Image	Type	Durée chiffrement	Durée de déchiffrement
Cameraman	gris	0.451 (s)	0.290 (s)
Lena	gris	0.687 (s)	0.543 (s)
Airport	gris	1.150 (s)	1.001 (s)
Lena	couleur	0.898 (s)	0.776 (s)

3.3.8 Bilan des évaluations et comparaison

Dans le tableau 3.16, nous avons récapitulé l'ensemble des évaluations faites avec les différentes métriques, et nous avons juxtaposé à cela ceux de bons cryptosystèmes en littérature pour une image Lena (grise 512×512). L'on constante au regard de ce tableau que les valeurs de *NCPR/UACI*, du pourcentage de différence, du coefficient de corrélation satisfont l'expectative d'un bon cryptosystème, et sont légèrement meilleurs que ceux des auteurs [52] et [116]. Bien que l'espace de clé soit inférieur à celui de Belazi et al. [52], il est suffisant large pour résister l'attaque de force brute. En outre, la valeur de l'entropie de l'information qui quantifie le désordre créé par notre algorithme, est la meilleure du tableau. Ces observations nous permettent d'affirmer que le cryptosystème proposé dans cette section est apte à être utilisé dans un environnement multimédia réel.

Tableau 3.16 *Comparaison de notre algorithme à quelques-uns en littérature.*

Tests	Image chiffrée $2^{\text{ième}}$ contribution	Image chiffrée $1^{\text{ère}}$ contribution	[52]	[116]
Espace de clé	$\mathbf{10^{80}}$	$\mathbf{10^{142}}$	10^{120}	10^{98}
Sensibilité de la clé	99.65	99.66	99.68	-
Coefficient de correlation moyen	0.006	0.004	0.007	0.514
Entropie de l'information	**7.9994**	**7.9994**	7.9963	7.9566
NCPR	**99.62**	**99.62**	99.62	99.62
UACI	33.54	33.59	33.57	32.47
Durée de chiffrement en (s)	**0.675**	**0.480**	0.497	2.401

3.4 Les attaques de cryptanalyse

Beaucoup d'algorithmes récents ont échoué le test de révélation des sous-clés par les attaques à image claire et à image chiffrée choisie [61, 62], simplement parce qu'ils ne l'ont pas effectué avant de proposer leur algorithme.

3.4.1 Cryptanalyse de : « un algorithme de chiffrement d'image basé sur la technique de substitution et le mixage de la récurrence GNPA »

a) L'attaque à image claire choisie

Nous avons mené l'attaque avec une image tout-zéro et son équivalent chiffré sur les images Lena, Peppers couleur, Airport et Baboon, selon le principe spécifié à la *section 2.4.4.1* de ce document. Les résultats de cette cryptanalyse à la figure 3.12 montrent bien qu'elles ont échoué étant donné que les cryptogrammes (Lena, Peppers couleur, Airport et Baboon) ayant subi la cryptanalyse reste chiffrée. Ces résultats ne surprennent guère si on prend en compte que chaque image chiffrée est fortement dépendante de l'empreinte du moindre bit à travers le PIST, et que cette dépendance a été renforcée par l'usage adéquate des méthodes de chiffrement combinées aux nouvelles récurrences.

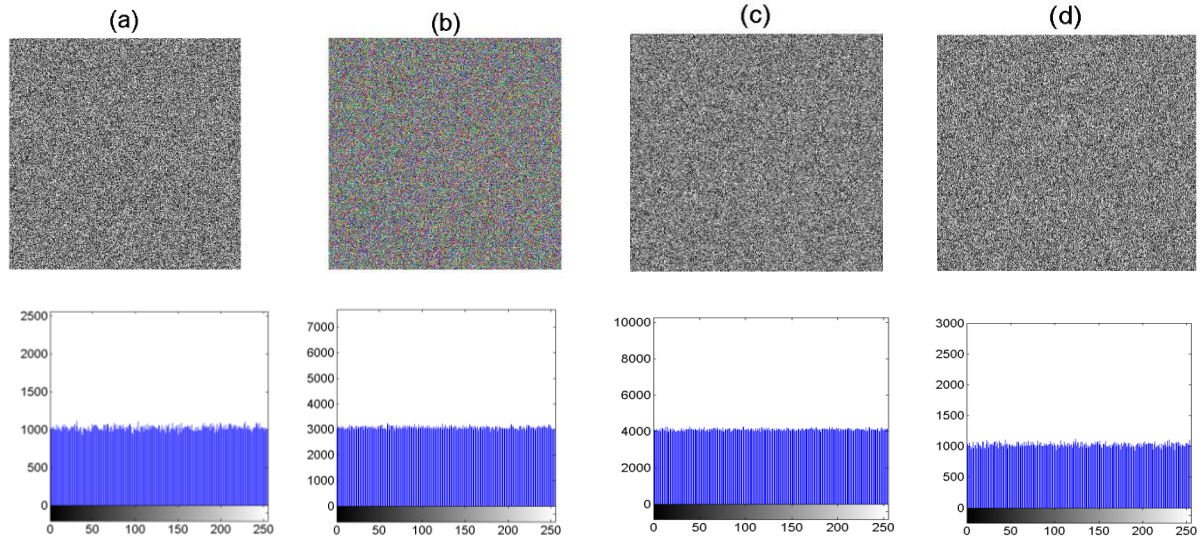


Figure 3.12 *Resultat de l'attaque à image claire choisie sur les cryptogrammes : (a) Lena, (b) Pepper couleur, (c) Airport, (b) Baboon.*

b) L'attaque à image chiffrée choisie

Cette attaque a été menée en considérant une image tout-un comme image chiffrée, et utilisant son équivalent déchiffrée. Plusieurs images (Caméraman, Lion, Lena couleur, flowers) ont été soumises aux attaques selon les équations (2.43) de la *section 2.4.4.2*, mais aucun succès apparent n'a été enregistré car les images attaquées restent indéchiffrable (figure 3.14). Cette robustesse s'explique par les raisons énumérées précédemment (*section 3.4.1.1*).

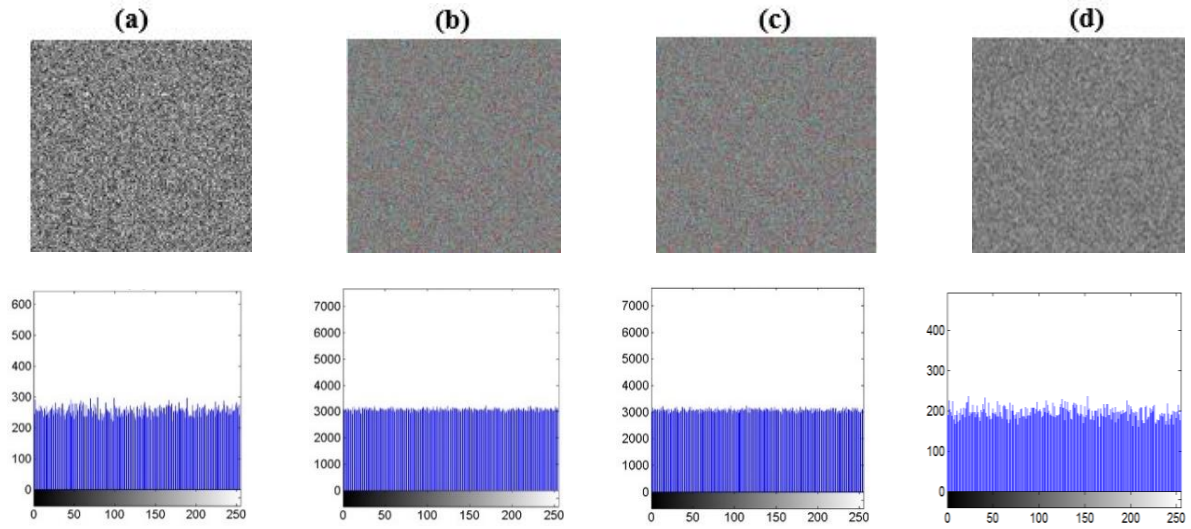


Figure 3.13 *Resultat de l'attaque à image claire choisie sur les cryptogrammes : (a) Cameraman, (b) Lion, (c) Lena couleur, (b) Baboon.*

3.4.2 Cryptanalyse de : « utilisation adéquate de la récurrence logistique dans un chiffrement à flot auto-synchronisé pour le chiffrement d'image ».

a) L'attaque à image claire choisie

Cette attaque a été menée avec une image tout-un et son équivalent chiffré sur les images à niveau de gris (Boat, Baboon couleur, Barbara, lion), selon le principe de la *section 2.4.4.1*. Les images de cryptanalyse obtenues à la figure 3.15 montrent bien que l'attaque a été mise en échec car elles sont toujours chiffrées. On rappelle que le chiffrement se fait en utilisant toutes les caractéristiques des pixels, une image ne peut donc pas être exploitée pour déchiffrer une autre. L'attaque à image claire choisie ne peut permettre de casser notre algorithme.

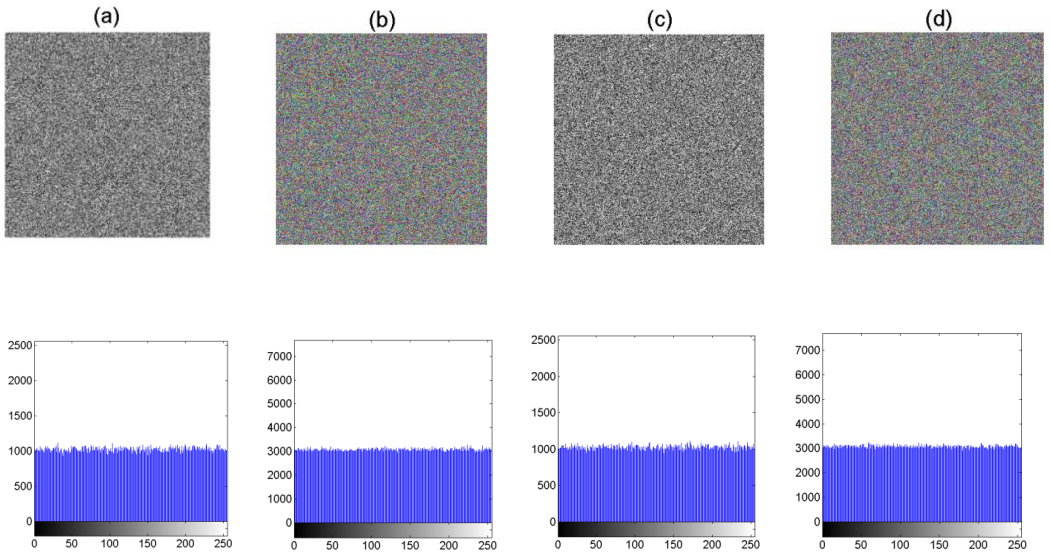


Figure 3.14 *Resultat de l'attaque à image claire choisie sur les cryptogrammes : (a) Boat, (b) Lion, (c) Baboon couleur, (b) Lion.*

b) L'attaque à image chiffrée choisie

Les attaques menées ici ont été réalisées en utilisant une image tout-un (considéré comme image chiffrée) et son équivalent déchiffré, pour révéler les images chiffrées (Goldhill, Peppers, Barbara, flowers), selon le principe de la section *2.4.4.2*. Les résultats obtenus et révélés à la figurent 3.16, montrent bien que les attaques sont sans effet parce qu'aucune image chiffrée n'a été révélée. L'attaque à image chiffrée choisie est impuissante face à notre algorithme de chiffrement.

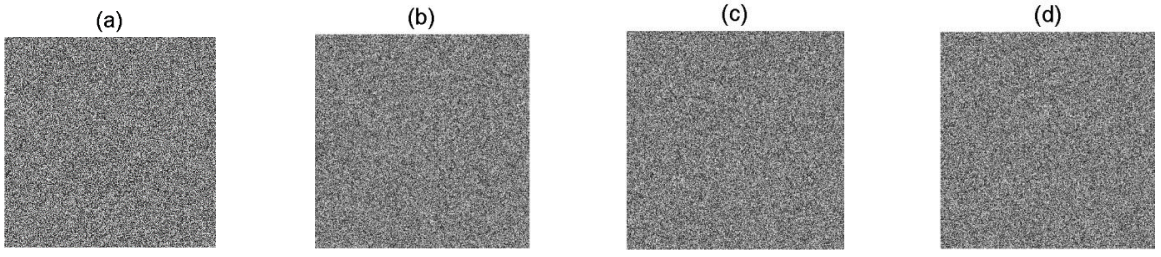


Figure 3.15 *Resultat de l'attaque à image claire choisie sur les cryptogrammes : (a) Goldhill, (b) Peppers, (c) Barbara, (b) Flowers.*

3.4.3 Discussion

Les attaques de cryptanalyses utilisées contre nos algorithmes ont été sans succès, démontrant qu'ils sont suffisamment robustes pour servir dans un environnement réel. Les raisons de ces échecs sont liées à la structure de chaque algorithme.

Dans le premier algorithme, le PIST est l'atout majeur favorisant la sensibilité du cryptogramme à un changement d'un pixel. Les opérations de diffusion et de permutation exercées ensuite amplifient cette moindre différence de départ, de sorte que chaque image produit un cryptogramme dépendant de tous les bits et du moindre bit à la fois. Les éléments caractéristiques d'un cryptogramme ne pourront donc jamais être utiles à la cryptanalyse d'un autre.

Le second algorithme dispose la permutation après la diffusion, or la permutation est la dernière opération de chiffrement et ses clés sont conçues à partir des éléments résultants de la diffusion. Ainsi la permutation porte l'empreinte de la diffusion, soit en d'autres termes, des caractéristiques au moins partielles de l'image claire. En outre, la diffusion est conçue en mode de chaînage avec le cryptogramme du pixel précédent qui influence le suivant par le biais de la clé renouvelée. Deux cryptogrammes de diffusion seront donc très différents de par la constitution (ou les caractéristiques) de leur image en claire. Dans ce contexte, il est impossible de mener une attaque à image choisie.

Conclusion

Les tests d'évaluations menés dans ce chapitre et sur des cryptogrammes produits par nos algorithmes ont été parfaitement à la hauteur des attentes. Des valeurs d'espace de clé, de sensibilité de la clé, de variance d'histogramme, de corrélation, d'entropie de Shannon, de $NCPR$ et $UACI$, et de durée de chiffrement obtenues sont comparables et même meilleures que les standards en littérature ou de celles de certains auteurs. L'échec des attaques de cryptanalyse effectuées ont bien démontré que l'on a conçu des algorithmes à base de récurrence logistique robuste.

Conclusion générale et perspectives

Dans ce document, nous nous sommes penchés sur les questions de sécurité cryptographique utilisant la récurrence logistique, et liées aux trafics des données numériques images générées par la démocratisation des systèmes de communications et d'Internet. Comme nous l'avons démontré, la cryptographie par chaos fournit une panoplie de solutions adéquates et d'un usage facile en matière de sécurité des données numériques. Mais le chiffrement des données images mis en avant dans ce travail reste faillible et/ou lent dans les méthodes indexées par cette cryptographie chaotique, malgré qu'il soit clairement établit qu'elle est la plus adaptée à leurs natures. Comme nous l'avons constaté, depuis l'émergence de cette cryptographie, sa récurrence maîtresse de chiffrement connu sous le nom de récurrence logistique, présente certains défauts inhérents à son utilisation et fragilise la plupart des cryptosystèmes d'images proposés. En effet, nous avons pu constater que les faiblesses telles que la périodicité, la distribution non uniforme des valeurs générées, la synchronisation sont des inconvénients majeurs de la récurrence logistique qui nuisent à la bonne conception de bons systèmes cryptographiques.

Pour contribuer à l'efficacité du chiffrement d'images à base de la récurrence logistique, nous avons principalement proposé deux méthodes de chiffrements distinctes l'une étant conçu en mixant des récurrences similaires et l'autre en contournant ses défauts.

Pour le choix du système chaotique de la première méthode de chiffrement, nous avons proposé l'utilisation des nouvelles récurrences à une dimension aux excellentes propriétés que sont : LOMAS, LOGOS, LOGAS, MAGOS, MAGAS et GAGOS. Ces récurrences sont engendrées par la combinaison deux à deux des récurrences défaillantes logistique, May, la récurrence de Gompertz et gaussienne. Les bonnes propriétés de nos récurrences créées ont été certifiées par les résultats des outils de base de détection du chaos à savoir , le calcul des exposants de Lyapunov dont toutes les valeurs ont été positives; et celui du tracé du diagramme de bifurcation affichant une répartition pleine et dense des variables aléatoires générées. La comparaison que nous

avons fait entre anciennes et nouvelles récurrences a démontré la supériorité des nouvelles sur les anciennes, car elles sont denses, ne présentent aucune forme de périodicité, la distribution de leurs variables est uniforme et en plus elles possèdent des valeurs d'exposants de Lyapunov supérieur à 2 (contre 0.7 pour les anciennes). La technique de chiffrement qui a suivi cette phase consiste à exercer au préalable sur l'image à chiffrer une technique de substitution sans clé (PIST) servant à augmenter sa sensibilité. Le chiffrement se poursuit ensuite avec l'utilisation de plusieurs de nos nouvelles récurrences chaotiques à une dimension aux excellentes propriétés. Les séquences des nombres aléatoires obtenues de ces récurrences ont servi à la technique de masquage-brouillage des pixels à l'aide d'une boite S-boxes ; la technique étant elle-même une altération de la valeur d'un pixel suivie de sa relocation en un processus. Le deuxième algorithme quant à lui utilise une clé externe 256 bits qui après avoir été transformée en paramètres de la récurrence logistique, est utilisée dans chiffrement à flot auto-synchronisé. Ce chiffrement prend en compte tous les états des pixels chiffrés précédemment en diffusion et est spécifiquement bâti sur les valeurs de l'ensemble des états des pixels chiffrés en permutation.

Nous avons effectué des tests d'évaluations sur nos méthodes de chiffrement afin de s'assurer qu'elles sont robustes et que leurs temps de chiffrement sont praticables en transfert de données. Ainsi les tests d'espace de clé, de sensibilité de clé, d'attaques statistiques (histogramme, coefficient de corrélation, entropie de l'information), d'attaques différentielles et de temps de chiffrement nous ont satisfait tant les valeurs des résultats étaient meilleures que les standards, et ceux de bien d'algorithmes proposés en littérature. En outre, les cryptanalyses (attaque à texte clair et texte chiffré choisi) effectuées sur nos deux algorithmes de chiffrement n'ont eu aucun succès, nous résumant à l'idée qu'elles sont sans failles de sécurité.

L'ensemble des résultats produits tant sur les nouvelles récurrences, que sur les méthodes de chiffrements, surmontent à priori assez bien, et eu égard les résultats de tous les tests, les manquements ou les difficultés liées à la cryptographie d'images utilisant la récurrence logistique.

Dans l'optique d'étendre, d'améliorer ou de valoriser la qualité des résultats obtenue dans ce document, nous envisageons pour la suite des travaux la possibilité :

- D'étendre ces algorithmes de chiffrement d'images aux vidéos. Etant donné que les données vidéos numériques sont assimilables à un empilement de taille bien

déterminé d'images numériques (trame), on se demande quel serait le champ d'application des algorithmes proposés pour ce types de données;

- D'étudier le comportement de nos algorithmes de chiffrements dans un environnement de canal de transmission réel, enfin d'envisager une exploitation sous forme logicielle visant l'un des multiples appels d'offres ;
- De convertir les structures algorithmiques de chiffrements proposés en structures électroniques de chiffrement grâce au FPGA, ou simplement de les adapter à des structures électroniques comme les microcontrôleurs et les DSP.
- De considérer les résultats obtenus par rapport à la cryptographie homomorphique qui a pour principal but de supplanter les atouts des ordinateurs quantiques.

Annexe

Algorithme de brouillage d'image sous matlab 2012b (section 2.3.2.2)

% Acquisition de l'image à traiter ; % exemple Im=imread(lena.tiff)

```
Im=imread(filename.extension); Ie=Im; Ie=uint8(Ie); [N M]=size(Ie); N=size(Ie,1); M=size(Ie,2); Ie=uint8(Ie);
```

% Initialisation des vecteurs des séquences aléatoires et des matrices

```
Ik=zeros(N,M); x=zeros(1,N+100); y=zeros(1, M+100); ligne_1=zeros(1,N); colonne_1=zeros(1,M);
```

%Génération des séquences speudo-aléatoire pour les lignes et les colonnes % la séquence ou le vecteur x pour la 1ière ligne et y pour la 1ière colonne

```
for i=1:N+100
    x(i+1)=r*x(i)*(1-x(i));
end
for i=1:N
  ligne(i)= x(i+100);
end
for j=1:M+100
 y(j+1)=r*y(j)*(1-y(j));
end
for j=1:M
  colonne(j)=y(j+100);
end
```

% Réorganiser les valeurs des séquences aléatoire ligne/colonne par ordre croissant et créer les séquences ligne_1 et colonne_1 par extraction des valeurs de position précédente de chaque nombre aléatoire.

```
for i=1:N
   X=sort(ligne); val=ligne(i);
Ligne_1(i)=find(X==val);
end
for j=1:M
   Y=sort(colonne); val=colonne(j);
   colonne_1(j)=find(Y==val);
end
```

% Opération de chiffrement par brouillage, Ik représente la matrice de l'image chiffrée

```
for i=1:N
   for j=1:M
      n=ligne_1(i) ; m=colonne_1(j);
       Ik(n,m)=Ie(i,j);
   end
end
Ik=uint8(Ik);
```

Liste des publications

- **Y. P. K. Nkandeu, A. Tiedeu**, An image encryption algorithm based on substitution technique and chaos mixing, Multimedia Tools and Application 78 (8) (2019) 10013-10034.

- **Nkandeu, Y.P.K., Mboupda Pone, J.R. & Tiedeu, A.** Image Encryption Algorithm Based on Synchronized Parallel Diffusion and New Combinations of 1D Discrete Maps. Sensing and Imaging 21, 55 (2020).

Références bibliographiques

[1] **Auguste Kerchkoffs**, la cryptographie militaire, Journal des sciences militaire, vol IX (1883) 5-38.

[2] **C. Shannon**, Communication theory of secrecy systems. Bell System Technical Journal 28, (1949) 656-715.

[3] **R. Matthews**, On the derivation of a chaotic encryption algorithm, Cryptologia XIII, London, 1 (1989) 29-42.

[4] **J. Fridrich**, Symmetric ciphers based on two-dimensional chaotic maps, Int. J. Bifurc. Chaos 8 (6) (1998) 1259-1284.

[5] **G. Chen, Y. Mao, C. K. Chui**, A symmetic image encryption Scheme based on 3D Chaotic cat maps. Chaos Solitons Fractals, 21 (2004) 749-761.

[6] **G. Jakimoski, L. Koracev**, Chaos and cryptography: Block encryption ciphers based on chaotic Maps. IEEE Transactions on circuits and systems. Fund. Theo. Appl. 48 (2) (2001) 45-51.

[7] **R. L. Devaney**, An Introduction to Chaotic Dynamical Systems (Addison-Wesley, Redwood City, California, USA).1989.

[8] **G. Alvarez, S. Li**, Some Basic Cryptographic Requirements for Chaos-Based Cryptosystems, International Journal of Bifurcation and Chaos, 16 (8) (2006) 2129-2151,.

[9] **T. Gao, Z. Chen**, A new image encryption algorithm based on hyper-chaos. Phys. Lett. A 372 (2008) 394–400.

[10] **N. K. Pareek, V. Patidar, K. K. Sud**, Image encryption using chaotic logistic map, Image Vis. Comput. 24 (2006) 926-934.

[11] **Y. lui, H. Fan, E. Y. Xie, G. Chen, C. li**, International Journal of Bifurcation and Chaos, 25(13) (2015) 1550 1558.

[12] **T. Gao, Z. Chen**, image encryption based on a new total shuffling algorithm, Chaos Solitons fractals 38 (2008) 213-220.

[13] **X. Wang, C. Duan, N. Gu**, A new chaotic cryptography based on ergodicity, International Journal of Modern Physics B 22 (7) (2008) 901–908.

[14] **D. Arroyo, G. Alvarez, S. Li, C. Li, V. Fernandez**, Cryptanalysis of a new chaotic cryptosystem based on ergodicity, International Journal of Modern Physics B, arXiv:0806.3183v1 [nlin.CD.19 juin 2008].

[15] **M. Sharma,** Image encryption based on a new 2D logistic adjusted logistic map, Multimedia Tools and Applications 79 (2020) 355–374.

[16] **V. Patidar, N. pareek, K.K Sud**, a new substitution-diffusion based image cipher using chaotic standard and logistic maps, commun. Nonlinear Sci. Numer. Simul. 14 (7) (2009) 3056-3075.

[17] **V. Patidar, N. Pareek, G. Purohit, K. Sud**, Modified substitution–diffusion image cipher using chaotic standard and logistic maps, Commun. Nonlinear Sci. Numer. Simul. 15 (2010) 2755–2765.

[18] **N. K Pareek, V. Patidar, K. K Sud**, Diffusion–substitution based gray image encryption scheme. Digital Signal Processing, 23 (3) (2013) 894–901.

[19] **C. Li, S. Li, A. Muhammad**, On the security defects of an image encryption Scheme, Image Vis. Comput. 27 (9) (2009) 1371-1381.

[20] **R. Rhouma, E. Solak, S. Belghith**, Cryptanalysis of a new substitution–diffusion based image cipher, Commun. Nonlinear Sci. Numer. Simul. 15 (7) (2010) 1887–1892.

[21] **C. Li, S. Li, K.-T. Lo**, Breaking a modified substitution–diffusion image cipher based on chaotic standard and logistic maps, Commun. Nonlinear Sci. Numer. Simul. 16 (2011) 837–843.

[22] **G. Ye**, Image scrambling encryption algorithm of pixel bit based on chaos map, Pattern Recognition Letters 31 (5) (2010) 347–354.

[23] **C. Li, M. Z. Q. Chen, K-T. Lo**, Breaking an image encryption algorithm based on chaos, Int. J. Bifurc. Chaos 21 (7) (2011) 2067-2075.

[24] **C. Li, K-T Lo**, Optimal quantitative cryptanalysis of permutation-only multimedia ciphers against plaintext attacks, Signal processing, 91 (2011) 949-954.

[25] **K. Rao, C. Gangadhar**, "Modified chaotic key-based algorithm for image encryption and its VLSI realization," Proceedings of the 2007 15th International Conference on Digital Signal Processing, (2007) 439–442.

[26] **H. Liu, Y. Liu,** Cryptanalyzing an image encryption scheme based on hybrid chaotic system and cyclic elliptic curve, Optics & Laser Technology 56 (2014) 15–19.

[27] **A. El-Latif, X. M. Niu**, A hybrid chaotic system and cyclic elliptic curve for image encryption. Int. J. Electron. Commun. (AEÜ) 67 (2013) 136–143.

[28] **D. Arroyo, G. Alvarez, V. Fermandez**, On the inadequacy of the logistic map for cryptographic applications, arXiv:0805.4355v1[nlin.CD] 28 May 2008.

[29] **D. Arroyo, G. Alvarez, V. Fermandez**, A basic framework for the cryptanalysis of digital chaos-based cryptography, arXiv0811.1859v1[cs.CR] 12 Nov 2008.

[30] **H.T. Panduranga, N. Kumar, S.K. Kiran**, Image encryption based on permutation-substitution using chaotic map and Latin square image cipher. The European Physical Journal-Special Topics. 223 (8) (2014) 1663–1677.

[31] **M. Ahmad, F. Ahmad**, Cryptanalysis of Image Encryption Based on Permutation-Substitution Using Chaotic Map and Latin Square Image Cipher, Proc. of the 3rd Int. Conf. on Front. of Intell. Comput. Theory and Applications (FICTA), (2014) 481-488.

[32] **X. Wang, K. Guo**, A new image alternate encryption algorithm based on chaotic map. Nonlinear Dyn.76 (4) (2014)1943–1950.

[33] **W.-S. Yap, C.-W. R. Phan,W.-C. Yau, S.-H. Heng**, Cryptanalysis of a new image alternate encryption algorithm based on chaotic map, Nonlinear Dyn, 80 (3) (2015) 1483–1491.

[34] **X.-Y. Wang, Y.-Q. Zhang, L.-T. Liu**, An enhanced sub-image encryption method, Optics and Lasers in Engineering 86 (2016) 248–254.

[35] **O. Mirzaei, M. Yaghoobi, H. Irani**, A new image encryption method: parallel subimage encryption with hyper chaos. Nonlinear Dyn 67 (2012) 557–566.

[36] **Dragan Lambi**, Cryptanalyzing a novel pseudorandom number generator based on pseudorandomly enhanced logistic map, Nonlinear Dyn DOI 10.1007/s11071-017-3583-1

[37] **M.A.Murillo-Escobar, C. Cruz-Hernandez, L. CardozaAvendano, R. Mendez-Ramirez,**: A novel pseudorandom number generator based on pseudorandomly enhanced logistic map. Nonlinear Dyn. 87 (2017) 407–425.

[38] **M.A. Murillo-Escobar, C. Cryz-Hernandez, F. Abundiz-Pérez, R. M. Lopez-Gutiérrez, O. R. A. Del Campo**, A RBG image encryption algorithm based on total plain image characteristics and chaos, Signal Process. 109 (2015) 119-131.

[39] **H. Fan M. Li D. Liu K. An**, Cryptanalysis of a plaintext-related chaotic RGB image encryption scheme using total plain image characteristics, Multimed Tools Appl. https://doi.org/10.1007/s11042-017-5437-8.

[40] **M. Khan T. Shah**, A Literature Review on Image Encryption Techniques, 3D Res 5 (2014)29-33.

[41] **B. Schneier**, Applied cryptography-protocols, algorithms, and source code in C, 2nd edn Wiley, Hobok (1996)

[43] **Maqableh, Mahmoud, Mohammad**. Analysis and design security primitives based on chaotic systems for eCommerce, Durham theses, Durham university, (2012). Available at Durham E-Theses online: http://etheses.dur.ac.uk/738/

[44] **Andrea Röck**, Quantifying Studies of (Pseudo) Random Number Generation for Cryptography. Thèse de Doctorat/PhD. École Polytechnique, INRIA Paris-Rocquencourt (2009)

[45] **Ghada Zaïbi**. Sécurisation par dynamiques chaotiques des réseaux locaux sans l au niveau de la couche MAC. Other. Ecole nationale d'Ingénieurs de Sfax (Tunisie), (2012).

[46] **Renaud Dumont**. Cryptographie et Sécurité informatique, Notes de cours Provisoires, Université de Liège, Faculté des Sciences Appliquées (2009 – 2010)

[47] **A. Jnaidi, A. Tarah**, "AES immunity Enhancement against algebraic attacks by using dynamic S-Boxes", Information and Communication Technologies: From Theory to Applications, ICTTA 2008. 3rd International Conference, 6 (2008) 7-11.

[48] **X. Wang, L. Teng, X. Qin**, A novel colour image encryption algorithm based on chaos, Signal Processing 92 (2012) 1101–1108.

[49] **L. Sui, M. Xin, A. Tian, H. Jin**, Single-channel color image encryption using phase retrieve algorithm in fractional Fourier domain, Optics and Lasers in Engineering 51 (2013) 1297–1309.

[50] **M.A. Murillo-Escobar, C. Cryz-Hernandez, F. Abundiz-Pérez, R. M. Lopez-Gutiérrez, O. R. A. Del Campo**, A RBG image encryption algorithm based on total plain image characteristics and chaos, Signal Process. 109 (2015) 119-131.

[51] **S. M. Seyedzadeh, B. Norouzi, M. R. Mosavi, S. Mirzakuchaki,** A novel color image encryption algorithm based on spatial permutation and quantum chaotic map, Nonlinear Dynamic 81(2) (2015) 511–529.

[52] **A. Belazi, A.A.A. El-Latif, S. belghiht**, A novel image encryption scheme based on substitution-permutation network and chaos, Signal Process. 128 (2016) 155-170.

[53] **M. J. Rostami, A. Shahba, S. Saryazdi, H. Nezamadabi-Pour**, A novel parallel image encryption with chaotic windows Based on logistic map, Computers and Electrical Engineering 000 (2017) 1–17

[54] **Volodymyr Lynnyk**, Chaos-based communication systems, Czech Technical University in Prague, Doctoral Thesis, Faculty of Electrical Engineering, Department of Control Engineering (2010) 15-16.

[55] **W. H. Morris, S. Stephen, R. L. Devaney**, Differential Equations, Dynamical Systems, and an Introduction to Chaos. Elsevier Academic Press. 2004.

[56] **H. S. Steven**, Nonlinear Dynamics and Chaos. Perseus Books. 1994.

[57] **C. H. Robert**, Chaos and Nonlinear Dynamics. Oxford University Press. 2000.

[58] **Ott. Edward**, Chaos in Dynamical Systems. Cambridge University Press. 2002.

[59] **M. Cencini, F. Cecconi, A. Vulpiani**. Chaos: From Simple Models to Complex Systems. World Scientic. 2010.

[60] **P.F. Verhulst**, Recherches mathématiques sur la loi d'accroissement de la population. Nouv. Mém. de l'Académie Royale des Sci. et Belles-Lettres de Bruxelles 18 (1845)1-41.

[61] **P.F. Verhulst**, Deuxième mémoire sur la loi d'accroissement de la population. Mém. de l'Académie Royale des Sci., des Lettres et des Beaux-Arts de Belgique 20 (1847)1-32.

[62] **W.J. Cunningham**, A non-linear differential-difference equation of the growth. Proc. Natl. Acad. Sci. USA 40 (1954) 708-13.

[63] **J. Meynard-Smith**, Mathematical Ideas in Biology. Cambridge University Press. (1968) (p.23)

[64] **R. May**, Biological populations with nonoverlapping generations: stable points, stable cycles, and chaos. Science 186 (1974) 645-647.

[65] **R. May**, Biological populations obeying difference equations: stable points, stable cycles, and chaos. J Theor Biol. 51 (1975) 511-624.

[66] **R. May**, Simple mathematical models with very complicated dynamics, Nature 261 (1976) 459-467.

[67] **W. Briden, S. Zhang**, Stability of solutions of generalized logistic difference equations, Periodica Mathematica Hungarica 9 (1994) 81-87.

[68] **J. Arino, L. Wang, G.S. Wolkowicz**, An alternative formulation for a delayed logistic equation. J Theor Biol 241(2006)109-19.

[69] **B. Hernandez-Bermejo, L. Brenig**, Some global results on quasipolynomial discrete systems, Nonlin Anal-RealWorld Applic 7 (2006) 486-496

[70] **H. Gao, Y. Zhang, S. Liang, D.Li**, A new chaotic algorithm for image encryption. Chaos, Solitons & Fractals, 29(2) (2006) 393–9.

[71] **S. Mazloom, A. M. Eftekhari-Moghadam**, Color image encryption based on Coupled Nonlinear Chaotic Map, Chaos, Solitons and Fractals, 42 (2009) 1745–1754.

[72] **C.-Y. Song, Y.-L. Qia, X.-Z Zhang**, An image encryption scheme based on new spatiotemporal chaos, Optik, 124 (2013) 3329-3334.

[73] **Y. Zhou, L. Bao, C. L. P. Chen**, A new 1D chaotic system for image encryption, Signal Process. 97 (2014) 172-182.

[74] **C. Pak, L. Huang**, A new color image encryption using combination of the 1D chaotic map, Signal Processing (2017), doi: 10.1016/j.sigpro.2017.03.011

[75] **G. Alvarez, J. M. Amigo, D. Arroyo, S. Li**, Lessons learnt from the cryptanalysis of chaos-based ciphers, Studies in Comput. Intel. 354 (2011) 257-295

[76] **H. Liu, X. Wang**, Triple image encryption scheme based on one-time key stream generated by chaos and plain images, The Journal of Systems and Software 86 (2013) 826-834.

[77]:http://www.bart-konieczny.com/fr/blog/securite-des-applications web/cryptagesymetrique-et-asymetrique. < visité le : 07/03/2019>

[78] **W. Diffie and M.E. Hellman**, New directions in cryptography. IEEE Transactions on Information Theory, IT-22 (6) (1976.) 644-654.

[79] **A. El-Latif, XM. Niu**, A hybrid chaotic system and cyclic elliptic curve for image encryption. Int. J. Electron. Commun. (AEÜ) 67 (2013) 136–143.

[80] **X. Wang, L. Liu, Y. Zhang**, A Novel Chaotic block image encryption algorithm based on dynamic random growth technique, Optics and Lasers in Engineering, 66 (2015) 10-18.

[81] **C. Fu, W.H. Meng, Y.F. Zhan**, An efficient and secure medical image protection scheme based on chaotic maps, Comput. Biol. Med. 43 (8) (2013) 1000–1010.

[82] **J.-X. Chen, Z.-L. Zhu, C. Fu, H. Yu, Y. Zhang**, Reusing the permutation matrix dynamically for efficient image cryptographic algorithm, Signal Process. 111 (2015) 294-307.

[83] **L. Xu, Z. Li, J. Li, W. Hua**, A novel bit-level image encryption algorithm based on chaotic maps, Optics and Lasers in Engineering, 78 (2016) 17–25.

[84] **G. Ye, H. Zhao, H. Cha**, Chaotic image encryption algorithm using wave-line permutation and block diffusion, Nonlinear Dyn. 83 (2016) 2067–2077.

[85] **A. Jain, N. Rajpal**, A robust image encryption algorithm resistant to attacks using DNA and chaotic logistic maps, Multimed. Tools. Appl.75 (10) (2015) 5455-5472.

[86] **J. S. A. Eyebe, J. Y. Effa, M. Alie**, Highly secured chaotic block cipher for fast image encryption, Applied Soft comput. 25 (2014) 435-444.

[87] **J. S. A. Eyebe, J. Y. Effa,S. L. Sabat, M. Ali**, A fast chaotic block cipher for image encryption, Commun Nonlinear. Sci. Numer. Simulat. 19 (3) (2014) 578-588.

[88] **Y. P. K. Nkandeu, A. Tiedeu**, An image encryption algorithm based on substitution technique and chaos mixing, Multimed. Tools. Appl. (2018) 1-22.

[89] **R. M. May**, Chaos and the dynamics of biological populations, Proc. Royal Soc. Lond. A, 413 (1987) 27–44.

[90] **C. H. Skiadas, C. Skiadas**, Chaotic Modelling and Simulation; Analysis of Chaotic Models, Attractors and Forms, Chapman & Hall/CRC Taylor & Francis Group, New York 2009.

[91] **B. Lü, L. Pan**, Propagation of vector Gaussian-Schell-model beams through a paraxial optical ABCD system, Opt. Comm. 205 (2002) 7-16.

[92] **R. Rhouma, S. Belghith**, Cryptanalysis of a new image encryption algorithm based on hyper-chaos, Phys Lett. A 372 (2008) 5973–5978.

[93] **D.Arroyo, C. Li, S. Li, G. Alvarez, W.A. Halang**, Cryptanalysis of an image encryption scheme based on a new total shuffling algorithm, Chaos Solitons Fractals 41 (5) (2009) 2613-2616.

[94] **C. Li, S. Li, A. Muhammad**, On the security defects of an image encryption Scheme, Image Vis. Comput. 27 (9) (2009) 1371-1381.

[95] **C. Li, S. Li, K.-T. Lo**, Breaking a modified substitution–diffusion image cipher based on chaotic standard and logistic maps, Commun. Nonlinear Sci. Numer. Simul. 16 (2011) 837–843

[96] **R. Rhouma, S. Belghith**, Cryptanalysis of a spatiotemporal chaotic image/video cryptosystem, Phys. Lett. A 372 (2008) 5790–5794

[97] **H. Hermassi, A. Belazi, R. Rhouma, S. Belghith**, Security analysis of an image encryption algorithm based on a DNA addition combining With chaotic maps, Multimed. Tools. Appl, 72 (3) (2014) 2211-2224.

[98] **A. Belazi, H. Hermassi, R. Rhouma, S. Belghith**, Algebraic analysis of a RGB image encryption algorithm based on DNA encoding and chaotic map, J. NonLinear Dyn.76 (4) (2014) 1989-2009.

[99] **X.-Y. Wang, Y.-Q. Zhang, L.-T. Liu**, An enhanced sub-image encryption method, Optics and Lasers in Engineering 86 (2016) 248–254.

[100] **X. Wang, D. Luan, X. Bao**, Cryptanalysis of an image encryption algorithm using Chebyshev generator, Digital Signal Process. 25 (2014) 244–247.

[101] **H. Liu, Y. Liu**, Cryptanalyzing an image encryption scheme based on hybrid chaotic system and cyclic elliptic curve, Optics & Laser Technology 56 (2014) 15–19.

[102] **R. Bechikh, H. Hermassi, A. A. A. El-Latif, R. Rhouma, S. Belghith**, Breaking an image encryption scheme based on a spatiotemporal chaotic system, Signal Processing : Image Commun. 39 (2015) 151-158.

[103] **B. Norouzi, S. Mirzakuchaki**, Breaking an Image Encryption Algorithm based on the New Substitution Stage with Chaotic Functions, Optik - Int. J. Light Electron. 127 (14) (2016) 5695-5701.

[104] **M. Ahmad, F. Ahmad**, Cryptanalysis of Image Encryption Based on Permutation-Substitution Using Chaotic Map and Latin Square Image Cipher, Adv. Intell. Syst. Comput. 327 (2014) 481-488.

[105] **L.B. Zhang, Z.L. Zhu, B.Q. Yang, W.-Y Liu, H.-F. Zhug, M. Zou**, Cryptanalysis and improvement of an efficient and secure medical image protection scheme, Math. Probl. Eng. 2015 (2015) 11.

[106] **L. Chen, S. Wang**, Differential cryptanalysis of a medical image cryptosystem with multiple rounds, Comput. Biol. Med. 65 (2015) 69–75.

[107] **Y. Liu, J. Tang, T. Xie**, Cryptanalyzing a RGB image encryption algorithm based on DNA encoding and chaos map, Optics & Laser Technology 60 (2014) 111–115.

[108] Z. **Ying-Qian, W. Xing-Yuan**, A symmetric image encryption algorithm based on mixed linear–nonlinear coupled map lattice, Information Sciences 273 (2014) 329–351.

[109] **X. Wu, H. Kan, J. Kurhts**, A new color image encryption scheme based on DNA sequences and multiple improved 1D chaotic maps, Appl. Soft Comput. J. (2015), http://dx.doi.org/10.1016/j.asoc.2015.08.008

[110] **R. Enayatifar, A. H. Abdullah, I. F. Isnin**, Image encryption using a synchronous permutation-diffusion technique[J]. Optics & Lasers in Engineering, 90 (2017) 146-154.

[111] **A. Y. Niyat, M. H. Moattar, M. N. Torshiz**, Color image encryption based on hybrid hyper-chaotic system and cellular automata[J]. Optics & Lasers in Engineering, 90 (2017) 225-237.

[112] **J. Wu, X. Liao, B. Yang**, Image encryption using 2D Hénon-Sine map and DNA approach, Signal proc. 153 (2018) 11-13.

[113] **Z. Parvin, H. Seyedarabi, M. Shamsi**, A new secure and sensitive image encryption scheme based on new substitution with chaotic function, Multimed. Tools Appl. (2014) 1-18.

[114] **R. Guesmi,M. A. B. Farah, Kachouri, M. Sametwang**, A novel chaos-based image encryption using DNA sequence operation and Secure Hash Algorithm SHA-2, Nonlinear. Dyn. 83 (3) (2016) 1123–1136.

[115] **L. Teng, X. Wang, J. Meng**, A chaotic color image encryption using integrated bit-level permutation, Multimed. Tools. Appl.77 (6) (2018) 6883-6896.

[116] **Ü. Çavusoglu, S. Kaçar, I. Pehlivan, A. Zengin**, Secure image encryption algorithm design using a novel chaos based S-Box, Chaos, Solitons and Fractals, 95 (2017) 92–101.

[117] **S. J. Sheela, K. V. Suresh1, D. Tandur**, Image encryption based on modified Henon map using hybrid chaotic shift transform, Multimed Tools Appl (2018) https://doi.org/10.1007/s11042-018-5782-2

[118] **Z. Gan, X. Chai, M. Zhang, Y. Lu**, A double color image encryption scheme based on three-dimensional brownian motion, Multimed Tools Appl (2018) https://doi.org/10.1007/s11042-018-5974-9

[119] **Z.L. Zhu, W. Zhang, W., K.W. Wong, H. Yu**, A chaos-based symmetric image encryption scheme using a bit-level permutation, Info. Sci. 181 (2011) 1171-1186.

[120] **A. Belazi, A. A. A. El-Latif, A.-V. Diaconu, R. Rhouma, S. Belghith**, Chaos-based partial image encryption scheme based on linear fractional and lifting wavelet transforms, Optics and Lasers in Engineering 88 (2017) 37–50.

[121] **X. Chai, Y. Chen, L. Broyde**, A novel chaos-based image encryption algorithm using DNA sequence operations, Optics and Lasers in Engineering, 88 (2017) 197–213

[122] **Y. Abanda, A. Tiedeu**, Image encryption by chaos mixing, IET Image Processing, 10 (10) (2016) 742 – 750.

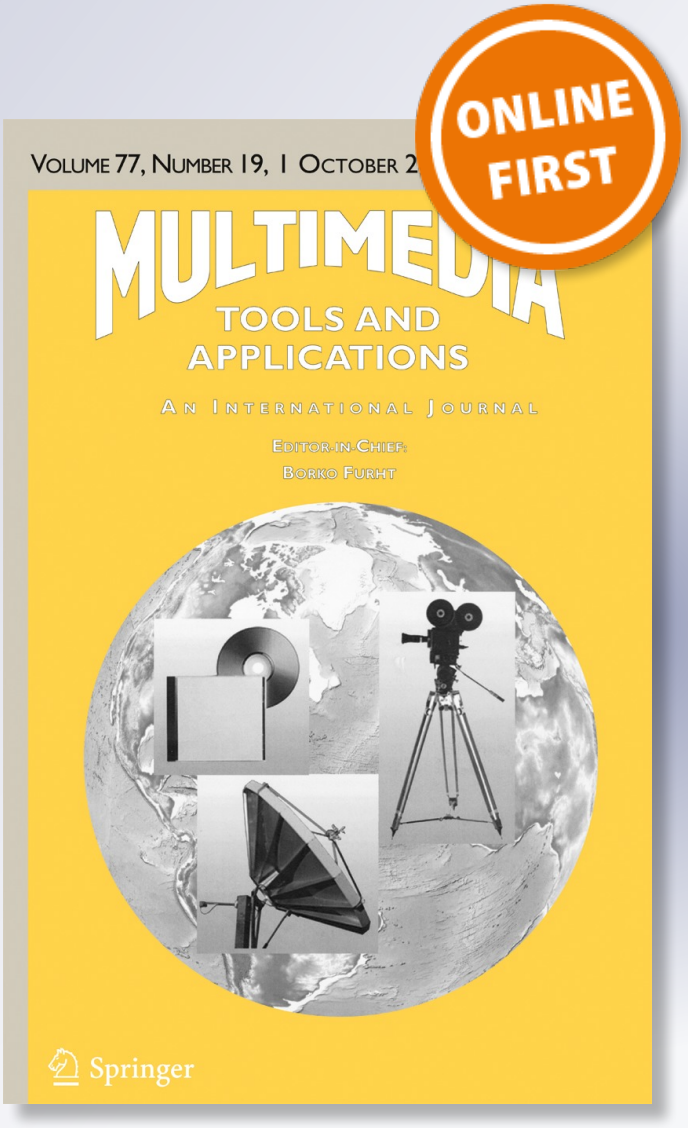
An image encryption algorithm based on substitution technique and chaos mixing

Yannick Pascal Kamdeu Nkandeu & Alain Tiedeu

Multimedia Tools and Applications
An International Journal

ISSN 1380-7501

Multimed Tools Appl
DOI 10.1007/s11042-018-6612-2



Springer

Multimedia Tools and Applications
https://doi.org/10.1007/s11042-018-6612-2

An image encryption algorithm based on substitution technique and chaos mixing

Yannick Pascal Kamdeu Nkandeu [1] · Alain Tiedeu [1]

Received: 7 February 2018 / Revised: 8 August 2018 / Accepted: 27 August 2018
Published online: 01 September 2018

Abstract
These recent years, countless chaos-based image encryption algorithms have been proposed to meet security needs in real time multimedia communication. However, many of these have exhibited flaws due to the chaotic map inadequacy. In this paper, we proposed a fast and secure image encryption algorithm by using new 1D chaotic systems, with better chaotic properties in the range of their control parameters. These new chaotic systems were obtained from well-known 1D chaotic maps (Logistic, May, Gaussian, Gompertz) with flaws in their chaotic properties. From the chaotic systems designed, we extracted a pseudo random number sequence (PRNS) and generated S-boxes. Then a novel technique of plain image substitution was used to enhance the sensitivity of the original image pixels, followed by a scrambling-masking technique using the generated S-box. Security tests and evaluation metrics confirmed that the proposed cryptosystem was efficient, practicable, and reliable, with high potential to be adopted for network security and secure communications because of its high encryption speed.

Keywords Chaos · Image encryption · Scrambling-masking

1 Introduction

In everyday application, cryptography serves at various levels and scopes of human activities in relation to secured data transfer with the guarantee of privacy [27, 30, 38]. With the rapid expansion of multimedia and internet, the urgent need for appropriate encryption algorithm for image and video online communication have favoured the up-rise of cryptography using chaotic maps.

Chaotic maps were found to be good candidates for cryptography because of the close relationship between chaos properties like ergodicity, sensitivity to initial conditions and control parameters, random-like behaviour, unpredictability, and properties of a good cipher such as sensitivity to key and plaintext, randomness in confusion and diffusion processes [22,

✉ Alain Tiedeu
alain.tiedeu@polytechnique.cm

[1] National Advanced School of Engineering, LAGEMES, PO. Box 8390, Yaoundé, Cameroon

32]. Furthermore chaos-based cryptography was also found suitable for image and video encryption as traditional cryptography (DES, IDEA, AES...) failed [10, 16, 38]. Many techniques and architectures involving different chaotic systems have therefore been published [1–26, 28, 29, 31–37, 39, 41, 43–57].

Permutation–diffusion is the most common architecture used in chaos-based cryptography. It consists of many rounds of crafty association between pixel values relocation (shuffling) and pixel values alteration (diffusion) using chaotic maps [1, 5–7]. Chen et al. [12] proposed a scheme of permutation-diffusion in which the diffusion key stream was extracted from the permutation matrix generated with Baker's map, and used for shuffling. In [7] Belazi et al. used four chaos-based cryptographic phases to design a substitution-permutation network. The author in [50] designed a scheme using a key hash function to generate a hash value from both plain image and a secret hash key, then he used logistic and standard maps and the hash value to perform permutation-diffusion and authentication of the encrypted/decrypted image, to prevent chosen plaintext and middle attack. Paper [35] proposed a secure and lightweight image encryption scheme based on 2D Baker's map, which scheme uses two sets of secret keys, one for permutation and another for diffusion.

In 2011, Zhu et al. [57] proposed a new scheme using a bit-level permutation in which separation of pixels into groups of bits depends on the percentage of pixel information. Afterward they used Cat and Logistic maps to permute and alter the abovementioned bits in pixel values. The scheme was cryptanalyzed and improved in [53]. Zhang et al. [54] proposed a new approach in which he considered an image of M×N size with 2^8 grayscale values as a 3D bit matrix M×N×8, and designed a new bit-level permutation architecture. Another bit-level permutation technique associated with pixel-level substitution and discrete cat map proposed in [17], was quickly cryptanalyzed and improved in [55]. Surprisingly, the improved scheme was broken and proved to have equivalent permutation Keys by Chen et al. in [11].

With the introduction of DNA computing, some researchers proposed image encryption algorithm based on DNA. Pixels in an image are DNA encoded and each nucleotide in the DNA encoded image is transformed to its base pair for DNA addition, complementary rules or replication with the help of chaotic maps and diffusion-confusion technique. However, the DNA image encryption algorithm using the Logistic map proposed in [51] was found non-invertible and prone to known-plaintext attack by authors in [19]. An enhanced version in [31] was cryptanalyzed in [8, 28]. Finally, more suitable versions for colour images using multiple improved 1D chaotic maps [23] and 2D logistic chaotic maps were proposed [21], but with high time consumption.

In 2015, Liu et al. [25] proposed a colour image encryption scheme based on three S-boxes generated in one-time by the complex Chen system. Each S-box randomly took turns to encrypt one of the colour components in each pixel adhering to the switching sequence. The S-box technique is an inheritance of traditional cryptography. The principle is to generate a random number of perfectly distributed 2D or 3D matrices and proceed to substitution, which is a nonlinear transformation, replacing each pixel value with another. In [44], Wang et al. proposed an image encryption based on dynamic S-boxes constructed by chaotic systems but with high time consumption. The encryption algorithm proposed by Belazi et al. [6] applied a lifting wavelet transform (LWT) to the original image in order to encrypt the latter by block permutation based on a chaotic Tent map. Then, a new S-box method based on chaotic system and linear fractional transform (LFT) was used to substitute the permuted image. Wang et al. [46] proposed an encryption algorithm using a discrete wavelet transform (DWT) to split up a digital image into different frequency coefficients before scrambling. Afterwards, the image sequences were encrypted with a multiple chaos encryption matrix.

Earlier in 2014, Eyebe et al. [14] proposed a scrambling-masking technique using a piece wise linear chaotic map (PWLCM) and the Leophantine equation (LDE) for generation of a large pseudo-random key stream. The algorithm achieved fast encryption since the pixel position and value were modified in a single process, but the encryption process was independent of the plain image characteristics. Another scrambling-masking scheme proposed by Huang et al. [20] using a 2D chaotic Chebychev function to scramble and mask pixel images was later proven to have security flaws [43].

Some schemes further analysed, were declared vulnerable to attacks as they were less sensitive to plaintext [5, 13, 15, 26, 29, 33, 34, 41, 47, 52]. The related algorithm suffered from inefficient chaining mode which prevents from different attacks, by creating an "avalanche effect" when a single pixel in the plain image is modified. Wang et al. [47] demonstrated that the sub-image encryption method based on hyperchaos presented by Mirzaei et al. [33] had some security weaknesses to chosen plaintext attack and improved on the scheme. Liu et al. [26] found some security defects in the scheme proposed by El-Latif et al. [13], designing an image cryptosystem based on a hybrid logistic map and a cyclic elliptic curve. Song et al. [41] presented a new spatiotemporal chaos and combined it to Nonlinear Chaotic Algorithm (NCA) to permute and diffuse image pixels. Bechikh et al. [5] analysed the scheme and concluded that the substitution key stream was the same for every cipher image/plain image pair. Murillo-Escobar [34] designed a colour image encryption algorithm using a 1D logistic map, and to avoid chosen/known plain image attack, the scheme relies on the plain image characteristics. Recently these encryption algorithms were successfully cryptanalyzed by Fan et al. [15]. The recent image encryption algorithm based on hyper-chaotic system and dynamic S-box proposed by Liu et al. [29] was proven to be insecure and not suitable for image secure communication by [52].

Other algorithms were prone to attack because their schemes contained chaotic maps (Logistic, Tent…) which had weaknesses like non-uniform data output, small key space, periodic data output, poor ergodic properties for some ranges of control parameter [3, 4, 23, 24]. To overcome these setbacks, some researchers suggested that they should not be used alone [2, 36], others proposed modified or new chaotic systems with better properties [1, 37, 39, 45, 49, 56]. In [56] for example, the author used two existing 1D chaotic maps to generate a number of new chaos with good chaotic properties, and designed an encryption algorithm capable of generating a completely different encrypted image each time it is applied to the same original image. The weakness of this cipher is its high decryption error. Sheela et al. [39] modified the Henon map in order to increase the chaotic region - which in turn improved the range of system parameters - and generated sequences for column and row shift transformation, then carried out diffusion using XOR operator. Yang et al. [49] generalized the chaotic Logistic map to the finite field, and designed a coloured image encryption scheme. Abanda et al. [1] combined outputs of Duffing and Colpitts chaotic systems to encrypt grey and colour images. In [45], Logistic and Kent chaotic mappings were used to produce two sub-matrices of pseudo random number, then a combined matrix was generated from both to perform XOR operation with the original data for encryption.

As can be seen, the common major weaknesses are the use of chaotic map with poor randomness properties outcome, lack of sensitivity to the plaintext in the method, and high computational load. With the purpose to overcome these difficulties, this paper introduces a fast image encryption algorithm built with new chaotic maps (obtained by mixing known 1D seed maps) and using a new encryption technique depending on the plain image. The new chaotic maps constructed proved to have better properties and were used to generate S-boxes

by their PRNS. The encryption technique first applies a substitution of the plain image by moving and "XORing" pixels in between themselves, such that the sensitivity to plain image is enhanced. The confusion-diffusion is obtained in one time, exploiting the S-box for the substituted-pixel relocation and masking in a scrambling-masking process. Security tests carried out and evaluation metrics applied to assess the cryptosystem confirm that the aforementioned setbacks were solved.

The rest of this paper is organized as follows. An overview of seed chaotic maps is given in Section 2 while in Section 3, the new chaotic maps are designed and proven to be chaotic. The encryption algorithm proposed, is detailed in Section 4. Section 5 focuses on common security tests like key space, key sensitivity, differential attack, while Section 6 concludes the paper.

2 Presentation of 1D seed chaotic maps

2.1 Logistic map

The Logistic map is one of the most studied chaotic systems and is mathematically translated by the equation:

$$x_{n+1} = rx_n(1-x_n) \tag{1}$$

Where $x_n \in [0, 1]$ is the discrete state of the output chaotic sequence, r is the control parameter with values in the range [0, 4]. The chaotic behaviour of the Logistic map is observed in the range [3.5, 4]. However, its chaotic properties are not so good, as shown in Fig. 1a.

2.2 May map

Published by Robert May [40], the May map has behaviour and properties similar to that of the Logistic map and is expressed by the following equation:

$$x_{n+1} = x_n \exp(a(1-x_n)) \tag{2}$$

Where $x_n \in [0, 10.9]$ and the control parameter a belongs to the range [0, 5].

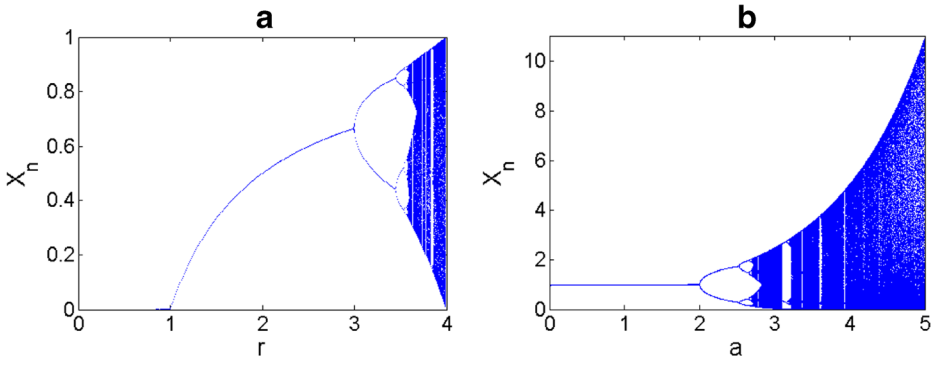


Fig. 1 The bifurcation diagrams of the **a** Logistic map and the **b** May map

Figure 1b illustrates the bifurcation diagram of May map in which, we can observe a non-uniform data output distribution and periodicity (expressed by blank space) in the range of [2.6, 5].

2.3 Gompertz map

First proposed by Gompertz [40], the Gompertz map has a very low level of chaotic behaviour and properties. Its equation is:

$$x_{n+1} = -bx_n \ln x_n \tag{3}$$

Where the control parameter $b \in [0,e]$, e=2.71829… and is the exponential function.

In Fig. 2a, one can see how low the chaocity the Gompertz map exhibits through its bifurcation diagram.

2.4 Gaussian map

The Gaussian map's equation is:

$$x_{n+1} = \exp\left(-\alpha x_n^2\right) + c \tag{4}$$

$\alpha \in [4.7, 17]$, $c \in [-1, 1]$.

Also known as mouse map, this map is a consequence of some mathematical assumptions and approximations over the Gaussian noise function [40]. The bifurcation diagram of the Gaussian map in Fig. 2b shows how their chaotic behaviour and properties are different from the ones of the Logistic, May and Gompertz, and appears in various small intervals of their control parameter c.

3 The proposed chaotic map

The chaotic properties of the above seed maps are not suitable to build a secure cryptosystem [3, 4]. In this section we design new maps with better levels of chaocity and that can therefore be integrated in a good cipher.

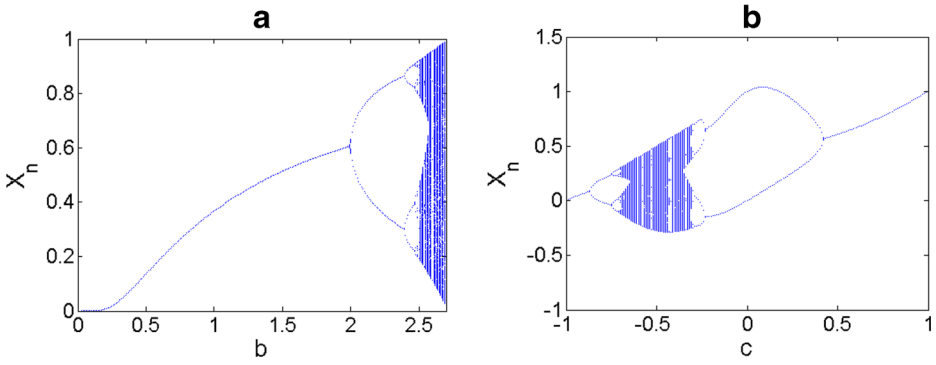


Fig. 2 The bifurcation diagrams of the **a** Gompertz map and the **b** Gaussian map

3.1 Scheme of the designed map

The method proposed by Zhou et al. [56] was adopted to combine the different seed maps. Depicted in Fig. 3, the scheme shows how a new map is obtained from a nonlinear combination of two different 1D chaotic maps.

3.2 The new chaotic maps

From the four 1D chaotic maps (Logistic, May, Gompertz and Gaussian) used as seed, six new chaos will be constructed and analysed, each using two different seeds with unified control parameter r. The criterion for the level of chaocity here will be the maximum Lyapunov exponent.

3.2.1 The Logistic-May system

The first system is made of the Logistic and May maps and is called the Logistic-May System (LOMAS). Its equation is written:

$$x_{n+1} = (x_n \exp((r+9)(1-x_n)) - (r+5)x_n(1-x_n)) \bmod 1 \tag{5}$$

Where $x_n \in [0, 1]$ and $r \in [0, 5]$.

The bifurcation diagram and Lyapunov exponent are shown in Fig. 4 a and d. We then can therefore see that chaotic properties are excellent within [0, 5], with a maximum Lyapunov exponent equal to 8.3.

3.2.2 The Logistic-Gompertz system

Logistic and Gompertz maps are the seeds of the second system called the Logistic-Gompertz system (LOGOS). It is mathematically given by Eq. (6).

$$x_{n+1} = (-(r-31)x_n(1-x_n) - (r+35)x_n \log x_n) \bmod 1 \tag{6}$$

Where $x_n \in [0, 1]$ and $r \in [0, 5]$.

Even though the Gompertz map has poor chaotic properties (Fig. 4), the bifurcation diagram of its combination with Logistic exhibits a rather good level of chaocity (Fig. 4b). The Lyapunov exponent of LOGOS has a mean value of 2.5 (Fig. 4e).

3.2.3 The Logistic-Gaussian system

Constructed with the Logistic and Gaussian maps, the third system is called the Logistic-Gaussian system (LOGAS) and is defined by:

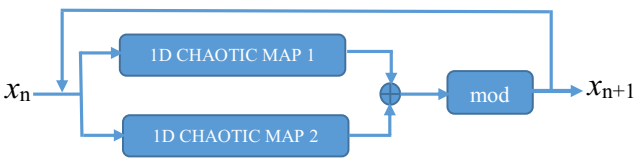


Fig. 3 The new chaotic map scheme

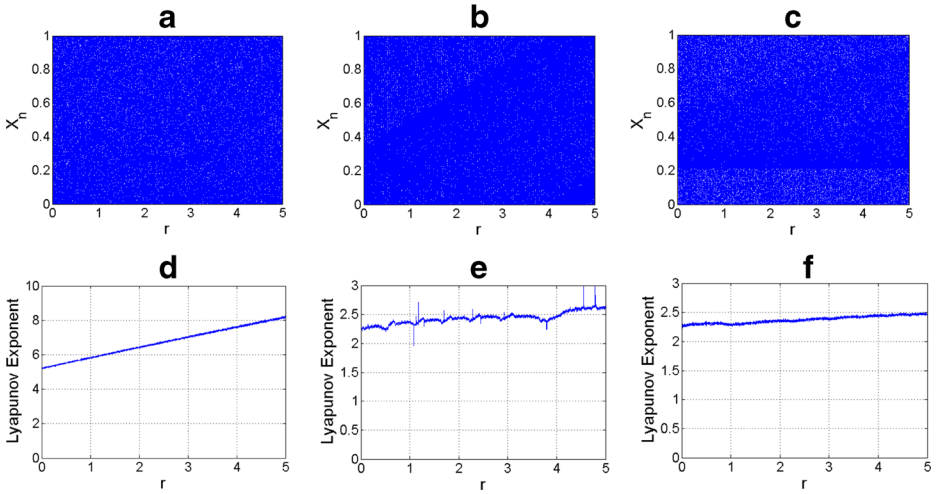


Fig. 4 The bifurcation diagrams and the Lyapunov exponent graphics of the new chaotic maps, **a-d** LOMAS, **b-e** LOGOS, **c-f** LOGAS

$$x_{n+1} = \left(-(r-33)x_n(1-x_n) + \frac{(r+37)}{4} + \exp\left(-\alpha x_n^2\right)\right) \bmod 1 \tag{7}$$

Where $x_n \in [0, 1]$, $r \in [0, 5]$, $\alpha \in [4.7, 17]$.

LOGAS's bifurcation diagram depicted by Fig. 4c proves its good chaotic behaviour in the range [0, 5]. Its maximum Lyapunov exponent is equal to 2.5 (Fig. 4f).

3.2.4 The May-Gompertz system

The fourth system derives from the May and Gompertz maps and is named the May-Gompertz system (MAGOS). Its equation is:

$$x_{n+1} = (x_n \exp((r+10)(1-x_n)) - (r+10)x_n \log x_n) \bmod 1 \tag{8}$$

Where $x_n \in [0, 1]$ and $r \in [0, 5]$.

Figure 5a and c show how its properties in terms of the bifurcation diagram and Lyapunov exponents (with a maximum value equivalent to 8.7) are excellent in the range of [0, 5].

3.2.5 The May-Gaussian system

The May combined to the Gaussian map form the fifth system called the May-Gaussian system (MAGAS) which is built up by the following equation:

$$x_{n+1} = \left(x_n \exp((r+10)(1-x_n)) + \frac{(r+5)}{4} + \exp\left(-\alpha x_n^2\right)\right) \bmod 1 \tag{9}$$

Where $x_n \in [0, 1]$, $r \in [0, 5]$, $\alpha \in [4.7, 17]$.

Through the bifurcation diagram of MAGAS in Fig. 5b, one can see an output sequence uniformly distributed within [0,1]. Figure 5e shows its positive Lyapunov exponents and belonging to the range [2.5, 5.6].

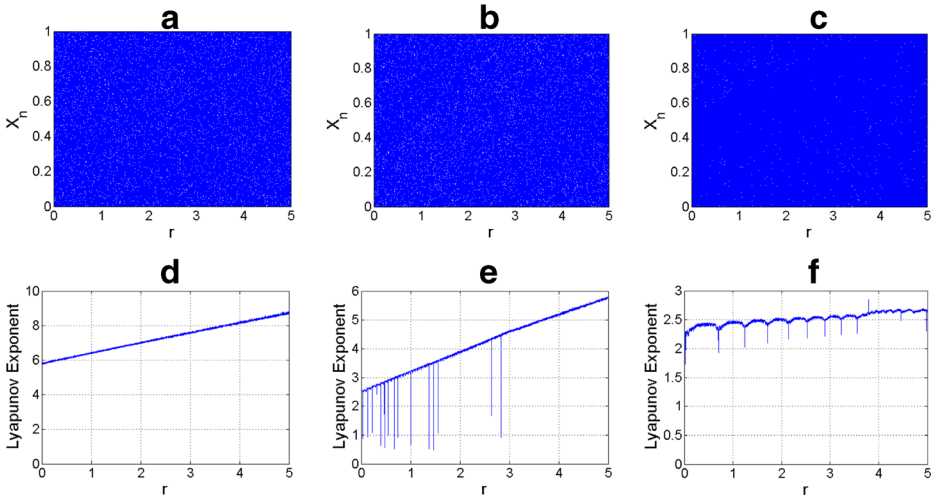


Fig. 5 The bifurcation diagrams and the Lyapunov exponent graphics of the new chaotic maps, **a**-**d** MAGOS, **b**-**e** MAGAS, **c**-**f** GAGOS

3.2.6 The Gaussian-Gompertz system

The last system designed is the Gaussian-Gompertz system (GAGOS). It uses the Gaussian and Gompertz maps and is expressed by the following formula:

$$x_{n+1} = \left(\frac{(r/5+26)}{4} + \exp\left(-\alpha x_n^2\right) - (r/5+26)x_n \log x_n\right) \bmod 1 \tag{10}$$

Where $x_n \in [0, 1]$, $r \in [0, 5]$, $\alpha \in [4.7, 17]$.

The GAGOS bifurcation diagram (Fig. 5c) shows a uniform distribution of sequences like the previous new chaos designed. It also has a mean Lyapunov exponent value around 2.5 (Fig. 5f).

3.3 Advantages of the new maps

All the combined chaotic systems designed above exhibit better chaotic behaviour than those obtained from a single seed. Although the mathematical theory behind chaocity improvement by combining seed maps is not yet established, one easily notices that the bifurcation diagrams of the combined maps have a wider chaotic range and a more uniform distribution of their density functions (Figs. 4a, b and c, 5a, b and c) than their seeds (Figs. 1 and 2.). Furthermore, the maximum Lyapunov exponent values of the Logistic, May, Gompertz and Gaussian maps are respectively equal to 0.6, 0.4, 0.5 and 0.7 [40]. The ones obtained with the LOMAS, LOGOS, LOGAS, MAGOS, MAGAS, GAGOS maps have values of 8.1, 2.6, 2.5, 8.7, 5.6 and 2.5 respectively (Figs. 4d, e and f and 5d, e and f). A high Lyapunov exponent means less iterations and less transient effects to have two totally different PRNS from two very close initial conditions with the same control parameter. It is therefore obvious that the new chaotic maps are better pseudo random number generators (PRNG) than their seeds. We can conclude that these will be more suitable for secure and high speed encryption provided the encryption algorithm is built around a good algebraic structure.

4 Proposed image encryption algorithm

This section presents the new chaotic encryption algorithm based on two main procedures which are a novel plain image substitution technique and a scrambling-masking technique using S-boxes.

4.1 Plain image substitution technique

The plain image substitution technique (PIST) is applied to the plain image for the enhancement of sensitivity such that any change in any pixel in the plain image will cause a substantial change in the corresponding cipher image. It does not depend on a key, and can be applied to any type of images. The steps to apply the PIST to an image are:

- From bottom to top, in each column in an image I of size $M\,X\,N$, replace the value of the pixel in process with the one obtained by bit-wise XOR operation between that value and the value of the previous pixel. The process starts on the second to the last pixel (Eq. (11)).

$$\begin{cases} I(M-i,j) = I(M-i,j) \oplus I(M+1-i,j) \\ i = 1,\ldots M-1; j = 1,\ldots N \end{cases} \tag{11}$$

Where (i, j) are indices of an image I of size $M\times N$, and the symbol $\oplus$is the bit-wise XOR operation.

- Repeat the same process in each row (Eq. (12)).

$$\begin{cases} I(i,N-j) = I(i,N-j) \oplus I(i,N+1-j) \\ i = 1,\ldots M; j = 1,\ldots N-1 \end{cases} \tag{12}$$

As a consequence of applying the PIST on a plain image, any tiny change in a pixel will spread and affect many pixels in the vertical and horizontal directions and finally the pixels in the first row and the first column (Fig. 6). In the Chaining Block Cipher (CBC) mode or Propagating Chaining Block Cipher (PCBC) mode, a modification of the first pixels in the plain image easily affects the rest when encryption occurs [38]. This technique can be used as a response to the insensitivity to plain images of many cryptosystems as shown in [5, 13, 15, 17, 26, 29, 31, 33, 34, 38, 41, 47, 52, 57]. Figure 6 shows how a grey image Lena is confused when it undergoes the PIST.

4.2 Confusion technique using S-boxes

This section describes the encryption process using as key the initial condition $w_0 = 0.4, x_0 = 0.3, y_0 = 0.2$, $z_0 = 0.1$ and control parameter $r_1 = 1$, $r_2 = 2$, $r_3 = 3$, $r_4 = 4$, $\alpha = 6$, of the new chaos maps (LOMAS, LOGOS, LOGAS, MAGOS). The Plain image I of size $M\times N$ which has undergone the PIST will yield the encrypted image C after a scrambling-masking process using S-boxes, and finally an encrypted image C' after the shuffling process. Below are the steps of the confusion technique.

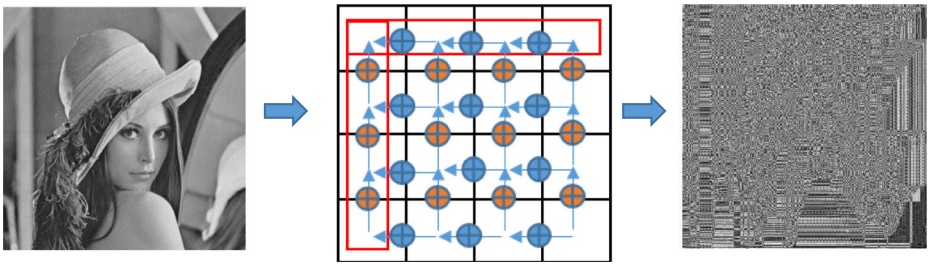


Fig. 6 Effect of PIST process on grey image Lena

Step 1: After 500 iterations to avoid transient effects, iterated LOMAS, LOGOS, LOGAS and MAGOS $M\times N$ times. Build four 1D array vectors W, X, Y, and Z of size $M\times N$, and two 2D vectors S_x (S-box obtained from 1D array X) and S_y (S-box obtained from 1D array Y) of sizes $M\times N$ respectively with the PRNS obtained from the chaotic systems above.

Step 2: Encrypt the first row and the first column of the image using the PRNS of the 1D vectors X and Y as expressed by Eq. (13).

$$\begin{cases} j = 2,3,\ldots N \text{ and } i = 1,2,\ldots M \\ C(1,j) = I(1,j) \oplus \left\lfloor \left(X(j+100)\times 10^{15}\right) \bmod 256 \right\rfloor \oplus \left\lfloor \left(Y(j+100)\times 10^{15}\right) \bmod 256 \right\rfloor \\ C(i,1) = I(i,1) \oplus \left\lfloor \left(X(i+200)\times 10^{15}\right) \bmod 256 \right\rfloor \oplus \left\lfloor \left(Y(i+200)\times 10^{15}\right) \bmod 256 \right\rfloor \end{cases} \quad (13)$$

Where i and j are the indices of an image I of size $M\times N$, the symbol $\lfloor t \rfloor$ is to round up the element of t to the nearest integer less than or equal to t, mod is the modulus operator and the symbol $\oplus$ denotes bit-wise XOR operation. (Fig. 7)

Step 3: For each encrypted value $C(i,1)$ and $C(1, j)$ of the first column and the first row, calculate the number $l(i)$ and $k(j)$ (Eq. (14)), and use each of them as an index to definitively extract a number respectively in the sets of values $\{2, 3\ldots M\}$ for rows, and $\{2, 3\ldots N\}$ for columns.

$$\begin{cases} i = 2,3\ldots N \text{ and } j = 2,3\ldots M \\ l(i) = 1 + \left(C(i,1) \oplus \left\lfloor \left(Z(i+200)\times 10^{15}\right) \bmod 256 \right\rfloor \times \left\lfloor \left(W(i+200)\times 10^{15}\right) \right\rfloor\right) \bmod (M+1-i) \\ k(j) = 1 + \left(C(1,j) \oplus \left\lfloor \left(Z(j+100)\times 10^{15}\right) \bmod 256 \right\rfloor \times \left\lfloor \left(W(j+100)\times 10^{15}\right) \right\rfloor\right) \bmod (N+1-j) \end{cases} \quad (14)$$

Where l, k are 1D arrays respectively of sizes M and N.

Step 4: Substitute the pixels of indices $i = \{2, 3\ldots M\}$ and $j = \{2, 3\ldots N\}$ in a scrambling-masking process with the indices a and b extracted respectively from $l(i)$ and $k(j)$ following Eq. (15).

$$\begin{cases} i = 2,3\ldots M \text{ and } j = 2,3\ldots N \\ C(a,b) = I(i,j) \oplus S_x(i,b) \oplus S_y(a,j) \end{cases} \quad (15)$$

Where each element of the S-boxes (S_x and S_y) are grey values obtained calculating (x (n) $\times 10^{15}$) mod 256 and (y (n) $\times 10^{15}$) mod 256 respectively, with $n = \{1, 2\ldots M\times N\}$.

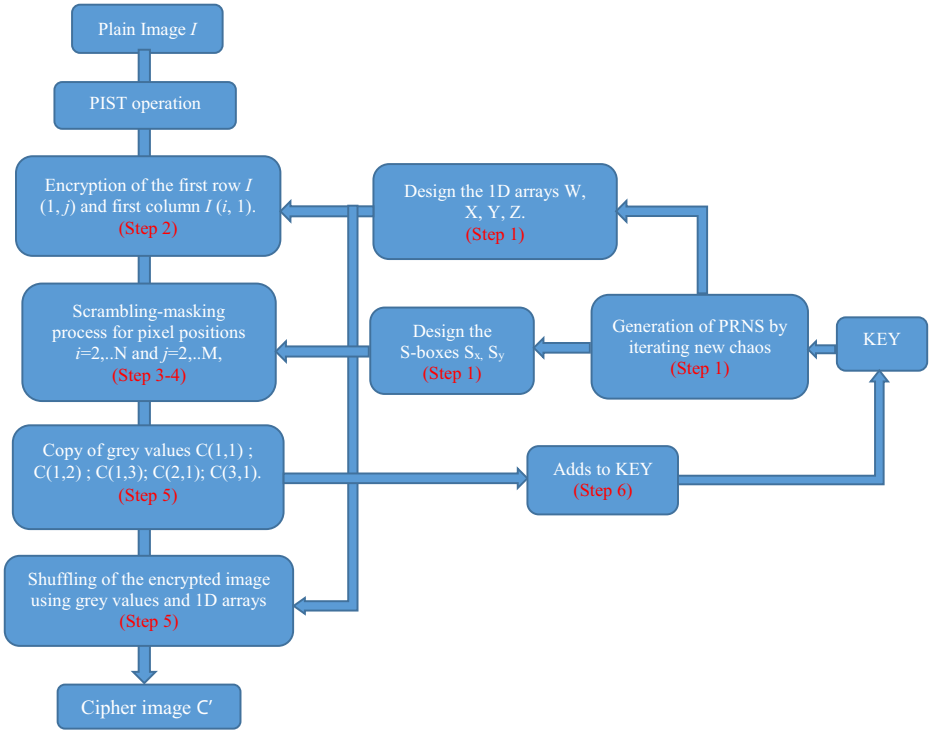


Fig. 7 Flowchart of encryption algoritm

Step 5: Determine 1D arrays *u(i)* and *v(j)* using the four systems and the five encrypted values of pixels $C(1,1)$, $C(1,2)$, $C(1,3)$, $C(2,1)$, $C(3,1)$ as expressed by Eq. (16). And substitute the indices *(i, j)* of each encrypted pixel with the indices *(c, d)*, where c and d are the values extracted from the sets $\{1, 2\ldots M\}$ and $\{1, 2\ldots N\}$ using *u(i)* and *v(j)* as indices respectively.

$$\begin{cases} i = 1,2,\ldots M \quad \text{and} \quad j = 1,2,\ldots N \\ u(i) = 1 + \left(C(1,1) \oplus \left\lfloor \left(W(i + C(1,2)) \times 10^{15}\right) \bmod 256 \right\rfloor \times \left\lfloor \left(X(i + C(1,3)) \times 10^{15}\right) \right\rfloor\right) \bmod (M + 1 - j) \\ v(j) = 1 + \left(C(1,1) \oplus \left\lfloor \left(Y(j + C(2,1)) \times 10^{15}\right) \bmod 256 \right\rfloor \times \left\lfloor \left(Z(j + C(2,3) \times 10^{15}\right) \right\rfloor\right) \bmod (N + 1 - i) \end{cases} \quad (16)$$

Where u, v are 1D arrays respectively of sizes M and N.

Step 1: Send a copy of $C(1,1)$, $C(1,2)$, $C(1,3)$, $C(2,1)$, $C(2,3)$ values as a part of the key.

4.3 Colour image encryption

The encryption process remains unchanged for coloured images containing R G B components. However, with the purpose of attaining high sensitivity, they will be joined together to form a unique matrix image before encryption, and restored as R G B components at the end.

4.4 Plain image recovering process

Recovering the plain image is done in two steps. Firstly, undo the substitution of indices of all pixels of the encrypted image by using the initial condition and control parameter of the four systems, and also the $C(1,1)$, $C(1,2)$, $C(1,3)$, $C(2,1)$, $C(2,3)$ values (step 6 of Section 4.2). Afterwards, recover confused pixels of indices $i = \{2,3,...M\}$ and $j = \{2,3,...N\}$ using the PRNS of the four systems, the first row and the first column of the confused image. Then decrypt the first row and the first column and apply the PIST to recover the original plain image.

5 Security analysis

The security tests in this section are conducted with a Core(TM) i5-2430 M processor, on a Matlab 2012b platform. The visual results of the encrypted images (Cameraman 256 ×256 image size, Colour Lena 512 ×512, Airport 1024 ×1024) of Fig. 8 are further analysed in terms of statistical attack, brute force attack, differential attack, Chosen plain and cipher image attack, and Speed.

5.1 Statistical analyses

Histogram, correlation analysis, and information entropy of the cipher image are the three main statistical tests (metrics) needed to assess robustness against statistical attack.

5.1.1 Histogram and variance of histogram

The histogram of a noise-like-image must be uniform. In Fig. 8, the histogram of the encrypted images (cameraman, Lena, airport) seem to be uniform. However, the best evaluation is done by calculating the variance of the histogram given by Eq. (18).

$$Var(z) = \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} \frac{1}{2} \left(z_i - z_j\right)^2 \tag{18}$$

Where Z is the vector of the histogram values and $Z = \{z_1, z_2, ..., z_{256}\}$, z_i and z_j are the numbers of pixels for which the grey values are equal to i and j respectively.

In Table 1, the values of the variance of the histogram of the proposed encryption algorithm are shown with the ones of some recent cryptosystems. It appears that the mean value of the histogram of the proposed cryptosystem is around 5465 which is very close to that of the good cryptosystem proposed in [48], and not far away from the ideal value of 5000 [36]. Histogram analysis proves that the proposed cryptosystem is safe as far as statistical attacks are concerned.

5.1.2 Correlation analysis

In a good encrypted image, there must be no or a very low correlation between neighbouring pixels in every direction. The usual method to assess this is to compute the correlation coefficient Cr of 5000 pairs of randomly chosen pixels in the horizontal (HC), vertical (VC), and diagonal (DC) directions using Eq. (19).

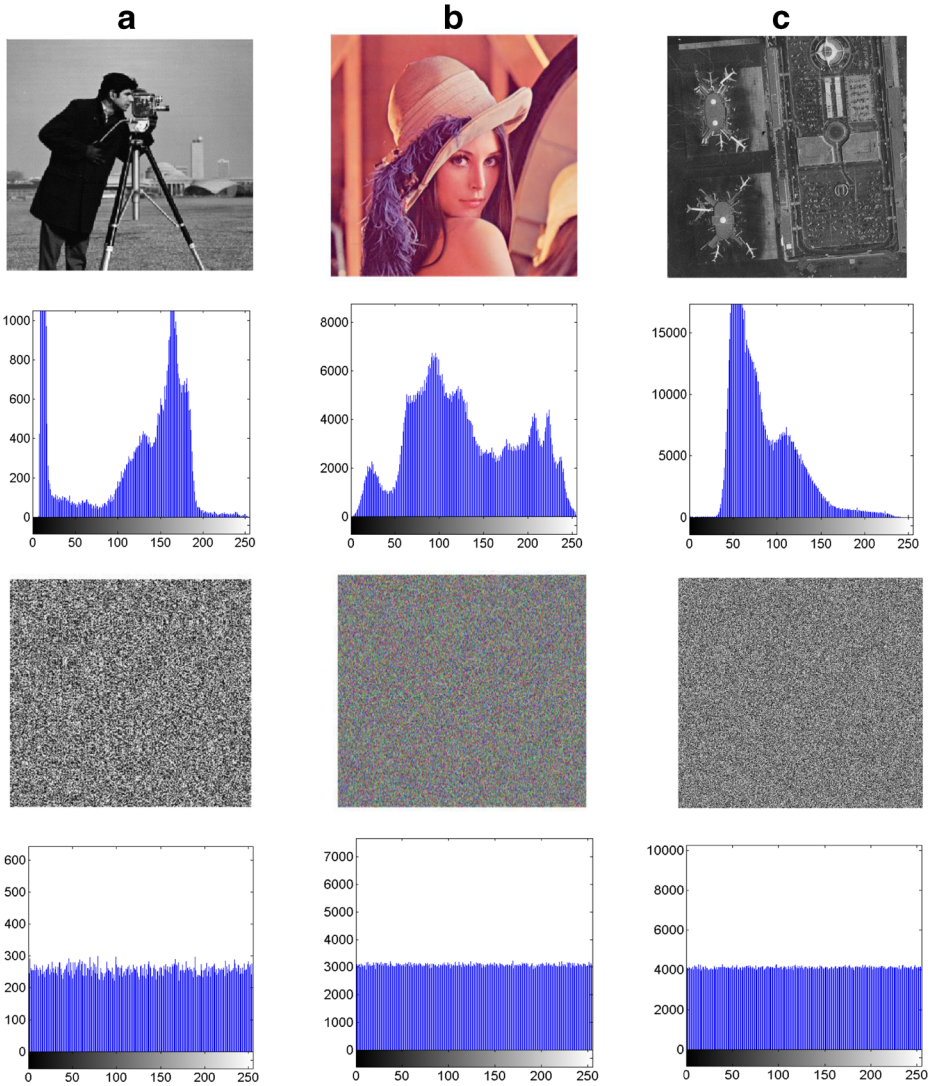


Fig. 8 The grey and colour images encrypted and their histograms. (a) Cameraman, (b) Colour Lena, (c) airport

$$Cr = \frac{K \times \sum_{i=1}^{K} X_i Y_i - \sum_{i=1}^{K} X_i^2 \times \sum_{i=1}^{K} Y_i^2}{\sqrt{\left(K \times \sum_{i=1}^{K} (X_i)^2 - \left(\sum_{i=1}^{K} X_i\right)^2\right) \times \left(N \times \sum_{i=1}^{K} (Y_i)^2 - \left(\sum_{i=1}^{K} Y_i\right)^2\right)}} \tag{19}$$

Where X and Y are grey scale values of two adjacent pixels in the image, K is the number of pairs of pixels and Cr is the value of correlation belonging to the [−1,1] range.

Cr tends to be 1 or − 1 for high correlations and tends to be 0 for very low correlations. Table 2 shows the calculated correlation coefficient of a 512×512 Lena image in every

Table 1 Variance of histograms of some cipher images

Grey image	Proposed algorithm	Ref. [48]	Ref. [36]
Cameraman (256 ×256)	5482.61	–	–
Lena (512 ×512)	5450.87	5468.38	5335.83
Airport (1024 ×1024)	5471.65	–	–

direction. A mean value of the proposed encryption algorithm is about 0.007, which tends towards zero. Moreover, Fig. 9 shows how grey values of the cameraman correlated in the horizontal direction in Fig. 9a, are spread in Fig. 9b. From these results, one can conclude that a statistical attack through correlation analysis between adjacent pixels cannot help to break the proposed encryption algorithm.

5.1.3 Information entropy analysis

The information entropy gives an account of the quantum of randomness present in a message (m) as follows.

$$H(m) = \sum_{i=0}^{2^K-1} p(m_i)\log_2(1/p(m_i)) \tag{20}$$

Where $p(m_i)$ represents the probability of symbolm_i, K is the number of bits of the message and 2^K all possible values. For a 256 grayscale image, the pixel data has 2^8 possible values and the ideal entropy of a true random image must be 8.

Table 3 shows entropy values of some images of the proposed encryption algorithm very close to 8 as expected, and slightly better than common ones in literature [48, 38].

5.2 Key analysis

5.2.1 Key space

The key space for an encryption algorithm must be large enough to avoid brute force attack. According to ref. [17], a key size of $\mathbf{10^{30}}$ is sufficient. The secret key of the proposed algorithm consists of 4 initial conditions (w_0, x_0, y_0, z_0), 5 control parameters (r_1, r_2, r_3, r_4, α), and five 8-bit values ($C(1,1)$, $C(1,2)$, $C(1,3)$, $C(2,1)$, $C(3,1)$), giving a total key space of $\mathbf{(10^{15})^4}\times\ \mathbf{(10^{15})^5}\times$

Table 2 Correlation coefficient of two adjacent pixels

Image	Size	Test	Plain image	Encrypted image
Cameraman	(256 ×256)	HC	0.9377	−0.009
		VC	0.9535	0.010
		DC	0.9043	−0.006
Lena	(512 ×512)	HC	0.9679	0.001
		VC	0.9845	−0.014
		DC	0.9580	−0.006
5.3.02	(1024 ×1024)	HC	0.9090	0.002
		VC	0.8989	−0.014
		DC	0.8610	0.018

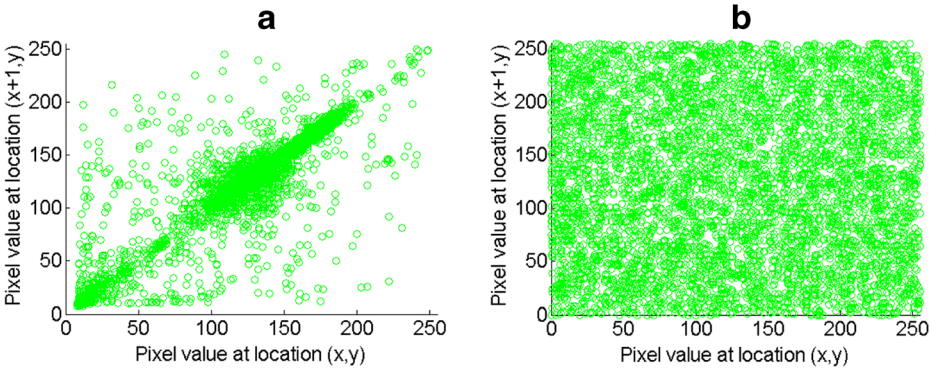


Fig. 9 The horizontal direction correlation graphics. (a) Original cameraman image, (b) encrypted cameraman image

$(2^8)^5 = 10^{142} \approx 2^{475}$, if the decimal precision is set at 15. This key space is large enough to make brute force attack inefficient.

5.2.2 Key sensitivity

An encryption algorithm must be sensitive to the less significant decimal value of its key to resist chosen plain image or chosen cipher image attack due to an insensitive, weak or equivalent key. Therefore, an original key $K_1 = r_1, r_2, r_3, r_4, \alpha, w_0, x_0, y_0, z_0$, and the modified versions K_2 ($r_2 = r_2 + 10^{-15}$ for K_2, the rest unchanged) and K_3 ($z_0 = z_0 + 10^{-15}$ for K_3, the rest unchanged) on the 15th decimal are used to encrypt the same image. Then, the percentage of difference between pixels of encrypted images is calculated. Table 4 reports that the encrypted images obtained with the K_1, K_2 and K_3 keys differ from one another by at least 99.62%. Such results are not surprising, considering the fact that the Lyapunov exponent of the new chaos is very high (Section 3). Furthermore, the airport image (Fig. 8c) encrypted with K_1 is decrypted with K_2 and K_3 and shown in Fig. 10. The decrypted image with K_2, K_3, has a noise-like appearance.

5.3 Differential attack analysis

A cryptosystem must be sensitive with respect to plain text or plain images, if not, it can undergo a successful differential attack. The sensitivity of a cryptosystem is evaluated through NCPR (Number of Pixel Change Rate) (Eq. (21)) and UACI (Unified Average Change

Table 3 Information entropy of some plain images and their cipher image

Grey image	Proposed algorithm	Ref. [9]	Ref. [34]
Cameraman (256 ×256)	7.9971	7.956	7.9953
Lena (512 ×512)	7.9994	–	7.9975
Baboon (512 ×512)	7.9993	–	–
Airport (1024 ×1024)	7.9998	–	7.9978

Table 4 Proof of key sensitivity

Key	Proposed algorithm	Ref. [42]	Ref. [44]
Key1 Vs Key2	99.61	99.58	99.65
Key2 Vs Key3	99.62	99.59	99.60
Key1Vs Key3	99.65	99.57	99.59

Intensity) (Eq. (22)) metrics [53], which consist of testing the influence of one pixel change on a plain image on the resulting cipher image.

$$NPCR = \frac{\sum_{i,j} D(i,j)}{W \times H} \times 100\% \tag{21}$$

$$UACI = \frac{1}{W \times H}\left[\sum_{i,j} \frac{|C_1(i,j) - C_2(i,j)|}{255}\right] \times 100\% \tag{22}$$

Where C_1 and C_2 are two images with same size $W \times H$. If $C_1(i,j) \neq C_2(i,j)$ then $D(i,j) = 1$, otherwise, $D(i,j) = 0$.

Table 5 gives the measurement of NCPR and UACI between two cipher images of the cameraman, Lena and the airport when a less significant bit (LSB) changed on the grey value in the first, middle, or last pixel's position. The values obtained are around the average of 99.62 for NCPR and 33.51 for UACI. These values are a little better than the ones proposed in literature [18, 34, 44]. Such good values result from the PIST, because the latter accumulates all pixel information in the first row and the first column of the image, which are then used in the confusion step (Section 4). Table 6 shows the effect of one pixel change on component RGB of Lena coloured image.

5.4 Cryptanalysis

Some recent encryption algorithms failed the chosen plain image or/and chosen cipher image attack with all-zero or all-one images [5, 13, 15, 26, 29, 33, 34, 41, 47, 52]. Both attacks are applied to the proposed cryptosystem.

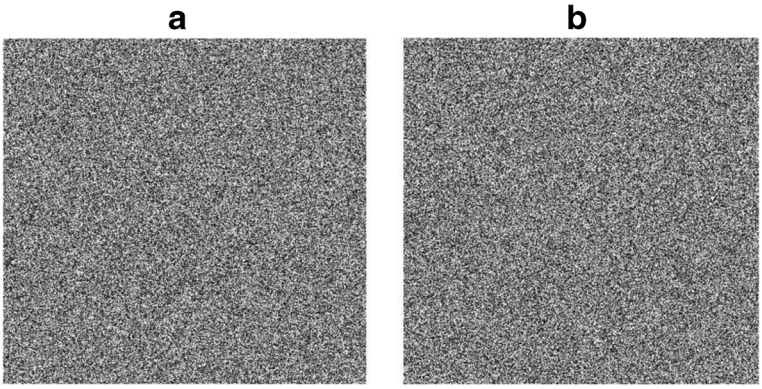


Fig. 10 Decrypted airport image with slightly different keys, (a) K_2, (b) K_3

Table 5 NCPR and UACI measure after a LSB change

		LSB change on the		
Image	Test	First pixel	Middle pixel	Last pixel
Cameraman (256×256)	NCPR	99.63	99.64	99.62
	UACI	33.55	33.55	33.52
Lena (512×512)	NCPR	99.62	99.62	99.61
	UACI	33.46	33.47	33.46
Airport (1024×1024)	NCPR	99.62	99.60	99.62
	UACI	33.47	33.50	33.47

5.4.1 Chosen plain image attack

The opener has an encrypted image C but does not know the key. However, he possesses a plain image P_0 of all-zero (or all-one), and its encrypted version C_0 obtained with the same unknown key. He extracts the sub-key used for pixel encryption as follows:

$$Sk_0{}^{i,j} = C_0{}^{i,j} \oplus P_0{}^{i,j} \tag{23}$$

Where $P_0{}^{i,j}$ =0, 0, 0... is a null-image in terms of grey values, and $C_0{}^{i,j}$, its corresponding cipher image, and (i,j) denotes the 2D-position of the pixels. The operation $(C_0{}^{i,j} \oplus P_0{}^{i,j})$extracts the key stream$Sk_0{}^{i,j}$.

Then, the sub-key extracted is used to recover the plain image P of the encrypted one C with Eq. (24).

$$P^{i,j} = C^{i,j} \oplus Sk_0{}^{i,j} \tag{24}$$

Where $P^{i,j}$is an image with the same size as $C_0{}^{i,j}$ and $C^{i,j}$ is its encrypted version.

In Fig. 11a, the chosen plain image attack on the airport encrypted image using a null-image has failed because, the scrambling-masking process and the shuffling process rely on the PIST which is highly sensitive to insignificant changes of a grey value. Therefore each encrypted image is specific to its plain image pixel characteristics.

5.4.2 Chosen cipher image attack

This time, the opener possesses an encrypted image C_0 made of all-zero (or all-one) and its corresponding decrypted version P_0. He still wants to determine the key-stream (according to Eq. (23)) necessary to recover the plain image P (colour Lena) from its encrypted image C

Table 6 NCPR and UACI measures On Lena RGB image

Image component	Test	Proposed algorithm	Ref. [18]	Ref. [34]
R	NCPR	99.63	99.59	99.63
	UACI	33.52	33.33	33.31
G	NCPR	99.61	99.62	99.60
	UACI	33.55	33.35	33.34
B	NCPR	99.64	99.63	99.61
	UACI	33.45	33.12	33.43

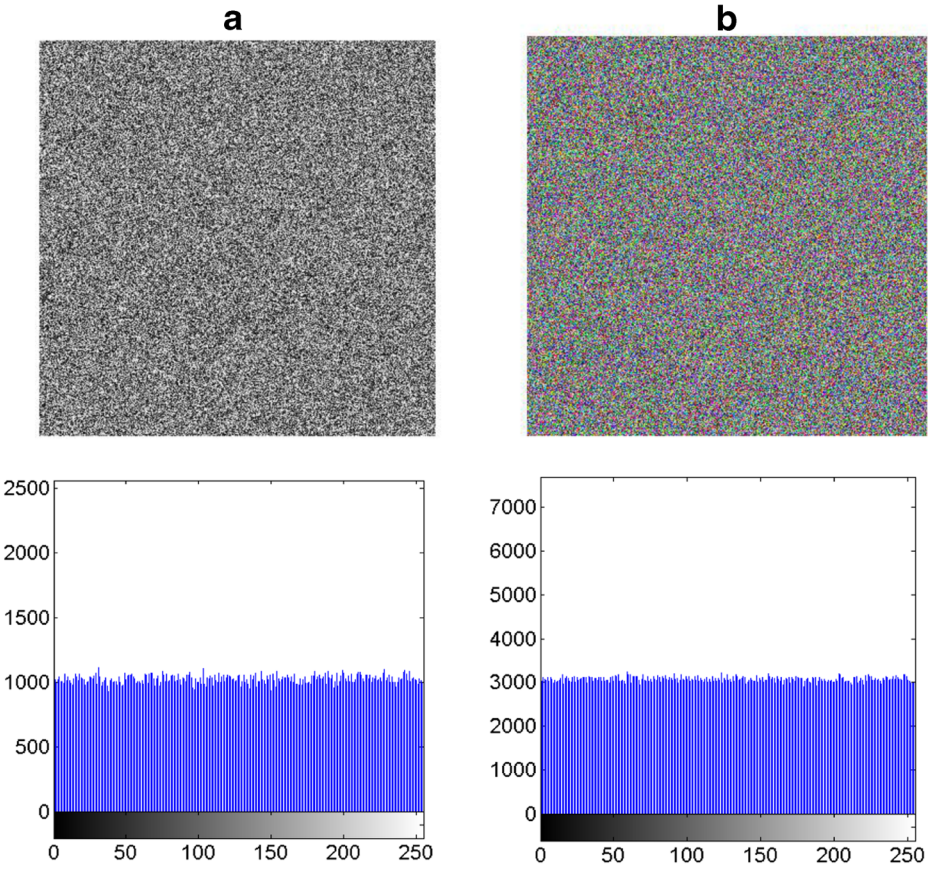


Fig. 11 Cryptanalysis. **a** Chosen plain image attack on grey Airport and its histogram, **b** chosen cipher image attack on colour Lena and its histograms

using Eq. (24). For the same reasons mentioned above (Section 5.4.1), and as shown by Fig. 11b, this type of attack does not work.

5.5 Encryption/decryption time analysis

Table 7 reports a comparison of encryption time by the proposed algorithm with some in literature for different images. The algorithm written under Matlab platform was not optimize.

The computer time consumption is smaller than those of [6, 9], while the proposed algorithm is faster than those in literature.

Table 7 Encryption time in seconds under Matlab 2012b

Image	Size	Type	Proposed algorithm	Ref. [9]	Ref. [6]	Ref. [56]
Cameraman	(256×256)	Grey	0.195	1.673	0.223	0.178
Lena	(512×512)	Grey	0.650	–	–	0.663
Airport	(1024×1024)	Grey	2.897	–	–	3.142
Lena	(512×512)	Colour	2.100	–	–	–

Table 8 Comparison of the proposed algorithm with others

Tests	Proposed cryptosystem	Ref. [42]	Ref. [2]	Ref. [57]
Key space	10^{142}	10^{96}	10^{143}	10^{42}
Key Sensitivity	99.66	–	–	99,61
Average Correlation	0.004	−0.005	0.003	0.004
Entropy	7.9994	7.999	7.9994	7.9993
NCPR	99.62	99.58	99.60	99.59
UACI	33.53	33.25	33.50	33.47
Encryption time in (s)	0.099	0.174 (4 round)	0.105 (4 round)	0.101

5.6 Overall comparison with other encryption algorithms

The performances of the proposed algorithm is here compared (Table 8) to those of some recent and good standing papers of the literature. Test are done using the colour Lena of size 512×512, and the time encryption is evaluated under visual C++ 2010 platform in accordance with real time multimedia application.

The metrics of the proposed cryptosystem reported in Table 8 suggests that, the key space, the NCPR, the encryption time, and the entropy wise are the best values. As far as key sensitivity, correlation and UACI are concerned, the values are in the order of the best values in literature.

6 Conclusion

In this paper, the proposed image encryption algorithm is based on many new 1D chaotic maps, and a substitution technique based plain image and S-boxes. The new chaotic maps are a combination of Logistic, May, Gompertz, and Gaussian maps, and have better maximum Lyapunov exponents, and therefore better chaotic properties than the originals. The encryption uses firstly, the PIST for image sensitiveness enhancement, secondly, S-boxes constructed with PRNS of the new chaos (LOMA, LOGOS, LOGAS, MAGOS, MAGAS, GAGOS, MAGOS); and thirdly, a scrambling-masking technique which permutes and diffuses image pixels in a single process with the help of S-boxes. The evaluation metrics of the proposed cryptosystem NCPR, UACI, correlation coefficient, entropy, key space and key sensitivity are amongst the best values in literature. More interestingly, a LSB change in any pixel value results in a totally different encrypted image, and chosen plaintext attack or chosen cipher image conducted is inefficient, proving the robustness of the cryptosystem. The encryption speed obtained with the non-optimized algorithm is fast enough in his current version for online multimedia communication. This proposed encryption algorithm can surely guarantee security and speed of all types of digital data transfer in a digital network.

Acknowledgement The authors wish to thank Professor Barbara ATOGHO-TIEDEU for proof reading the manuscript.

References

1. Abanda Y, Tiedeu A (2016) Image encryption by chaos mixing. IET Image Process 10(10):742–750
2. Ahmed A, El-Latif A, Li L, Niu X (2014) A new image encryption scheme based on cyclic elliptic curve and chaotic system. Multimed Tools Appl 70(3):1559–1584
3. Arroyo D, Alvarez G, Fermandez V (2008) On the inadequacy of the logistic map for cryptographic applications. arXiv:0805.4355v1[nlin.CD]
4. Arroyo D, Alvarez G, Fermandez V (2008) A basic framework for the cryptanalysis of digital chaos-based cryptography. arXiv:0811.1859v1[cs.CR]
5. Bechikh R, Hermassi H, El-Latif AAA, Rhouma R, Belghith S (2015) Breaking an image encryption scheme based on a spatiotemporal chaotic system. Signal Process Image Commun 39:151–158
6. Belazi A, Abd El-Latif AA, Diaconu A-V, Rhouma R, Belghith S (2017) Chaos-based partial image encryption scheme based on linear fractional and lifting wavelet transforms. Opt Lasers Eng 88:37–50
7. Belazi A, El-Latif AAA, Belghiht S (2016) A novel image encryption scheme based on substitution-permutation network and chaos. Signal Process 128:155–170
8. Belazi A, Hermassi H, Rhouma R, Belghith S (2014) Algebraic analysis of a RGB image encryption algorithm based on DNA encoding and chaotic map. J NonLinear Dyn 76(4):1989–2009
9. Çavusoglu Ü, Kaçar S, Pehlivan I, Zengin A (2017) A Secure image encryption algorithm design using a novel chaos based S-Box. Chaos, Solitons Fractals 95:92–101
10. Chen G, Mao Y, Chui CK (2004) A symmetric image encryption Scheme based on 3D Chaotic cat maps. Chaos, Solitons Fractals 21:749–761
11. Chen L, Wang S (2015) Differential cryptanalysis of a medical image cryptosystem with multiple rounds. Comput Biol Med 65:69–75
12. Chen J-X, Zhu Z-L, Fu C, Yu H, Zhang Y (2015) Reusing the permutation matrix dynamically for efficient image cryptographic algorithm. Signal Process 111:294–307
13. El-Latif A, Niu XM (2013) A hybrid chaotic system and cyclic elliptic curve for image encryption. Int J Electron Commun 67:136–143
14. Eyebe JSA, Effa JY, Alie M (2014) Highly secured chaotic block cipher for fast image encryption. Appl Soft Comput 25:435–444
15. Fan H, Li M, Liu D, AN K (2017) Cryptanalysis of a plaintext-related chaotic RGB image encryption scheme using total plain image characteristics. Multimed Tools Appl 1–25
16. Fridrich J (1998) Symmetric ciphers based on two-dimensional chaotic maps. Int J Bifurc Chaos 8(6):1259–1284
17. Fu C, Meng WH, Zhan YF (2013) An efficient and secure medical image protection scheme based on chaotic maps. Comput Biol Med 43(8):1000–1010
18. Guesmi R, Farah MAB, Kachouri A, Sametwang M (2016) A novel chaos-based image encryption using DNA sequence operation and Secure Hash Algorithm SHA-2. Nonlinear Dyn 83(3):1123–1136
19. Hermassi H, Belazi A, Rhouma R, Belghith S (2014) Security analysis of an image encryption algorithm based on a DNA addition combining With chaotic maps. Multimed Tools Appl 72(3):2211–2224
20. Huang XL (2012) Image encryption algorithm using chaotic Chebyshev generator. Nonlinear Dyn 67(4): 2411–2417
21. Jain A, Rajpal N (2015) A robust image encryption algorithm resistant to attacks using DNA and chaotic logistic maps. Multimed Tools Appl 75(10):5455–5472
22. Jakimoski G, Koracev L (2001) Chaos and cryptography: Block encryption ciphers based on chaotic Maps. IEEE Transactions on Circuits and Systems Fund Theo Appl 48(2):163–169
23. Li C, Li S, Lo K-T (2011) Breaking a modified substitution–diffusion image cipher based on chaotic standard and logistic maps. Commun Nonlinear Sci Numer Simul 16:837–843
24. Li C, Li S, Muhammad A (2009) On the security defects of an image encryption Scheme. Image Vis Comput 27(9):1371–1381
25. Liu H, Kadir A, Gong P (2015) A fast color image encryption scheme using one-time S-Boxes based on complex chaotic system and random noise. Optics Comm 338:340–347
26. Liu H, Liu Y (2014) Cryptanalyzing an image encryption scheme based on hybrid chaotic system and cyclic elliptic curve. Opt Laser Technol 56:15–19
27. Liu Y, Nie L, Han L, Zhang L, Rosenblum DS (2015) Action2Activity: Recognizing Complex Activities from Sensor Data In: Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence IJCAI, 2015. aaai.org, pp 1617–1623
28. Liu Y, Tang J, Xie T (2014) Cryptanalyzing a RGB image encryption algorithm based on DNA encoding and chaos map. Opt Laser Technol 60:111–115
29. Liu Y, Tong X, Ma J (2015) Image encryption algorithm based on hyper-chaotic system and dynamic S-box. Multimedia Tools Appl 1–21

30. Liu Y, Zhang L, Nie L, Yan Y, Rosenblum DS (2016) Fortune Teller: Predicting Your Career Path. In: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence AAAI, 2016. aaai.org, pp 201–207
31. Liu L, Zhang Q, Wei X (2012) A RGB image encryption algorithm based on DNA encoding and chaotic map. J Comput Electric Eng 28(5):1240–1248
32. Matthews R (1989) On the derivation of a chaotic encryption algorithm. Cryptologia XIII 1:29–42
33. Mirzaei O, Yaghoobi M, Irani H (2012) A new image encryption method: parallel subimage encryption with hyperchaos. Nonlinear Dyn 67:557–566
34. Murillo-Escobar MA, Cryz-Hernandez C, Abundiz-Pérez F, Lopez-Gutiérrez RM, Del Campo ORA (2015) A RBG image encryption algorithm based on total plain image characteristics and chaos. Signal Process 109:119–131
35. Noura, M, Noura, H, Chehab A, Mansour M M, Sleem M, Couturier R (2018) A dynamic approach for a lightweight and secure cipher for medical images. Multimed Tools Appl 1–19
36. Pandurang HT, Kumar N, Kiran SK (2014) Image encryption based on permutation-substitution using chaotic map and Latin square image cipher. The European Physical J-Spec Topics 223(8):1663–1677
37. Parvin Z, Seyedarabi H, Shamsi M (2014) A new secure and sensitive image encryption scheme based on new substitution with chaotic function, Multimed Tools Appl 1–18
38. Schneier B (1996) Applied cryptography-protocols, algorithms, and source code in C, 2nd edn. Wiley, Hoboken
39. Sheela S J, Suresh K V, Tandur D (2018) Image encryption based on modified Henon map using hybrid chaotic shift transform. Multimed Tools Appl 1–29
40. Skiadas CH, Skiadas C (2009) Chaotic Modelling and Simulation; Analysis of Chaotic Models, Attractors and Forms. Chapman & Hall/CRC Taylor & Francis Group, New York
41. Song C-Y, Qia Y-L, Zhang X-Z (2013) An image encryption scheme based on new spatiotemporal chaos. Optik 124:3329–3334
42. Wang X, Liu L, Zhang Y (2015) A Novel Chaotic block image encryption algorithm based on dynamic random growth technique. Opt Lasers Eng 66:10–18
43. Wang X, Luan D, Bao X (2014) Cryptanalysis of an image encryption algorithm using Chebyshev generator. Digital Signal Process 25:244–247
44. Wang X, Qiang W (2014) A Novel image encryption algoritm based on dynamic S-boxes constructed by chaos. Nonlinear Dyn 75:567–576
45. Wang W, Si M, Pang Y, Ran P, Wang H, Jiang X, Liu Y, Wub J, Wu W, Chilamkurti N, Jeon G (2018) An encryption algorithm based on combined chaos in body area networks. Comput Electr Eng 65:282–291
46. Wang W, Tan H, Sun P, Yu P, Ren B (2015) A novel digital image encryption algorithm based on wavelet transform and multi-chaos. In: Proceeding of the International Conference Wireless Communications and Sensor Network, WCSN 2015, pp 711–71946.
47. Wang X-Y, Zhang Y-Q, Liu L-T (2016) An enhanced sub-image encryption method. Opt Lasers Eng 86: 248–254
48. Wua X, Kan H, Kurths J (2015) A new color image encryption scheme based on DNA sequences and multiple improved 1D chaotic maps. Appl Soft Comput 37:24–39
49. Yang B, Liao X (2018) A new color image encryption scheme based on logistic map over the finite field ZN. Multimed Tools Appl 1–19
50. Yang H, Wong K-W, Liao X, Zhang W, Wei P (2010) A fast image encryption and authentification scheme based on chaotic maps. Commun Nonlinear Sci Numer Simul 15:3507–3517
51. Zhang Q, Guo L, Wei X (2010) Image encryption using DNA addition combining with chaotic maps. J Math Comput Modeling 52:2028–2035
52. Zhang X, Nie W, Ma Y et al (2017) Cryptanalysis and improvement of an image encryption algorithm based on hyper-chaotic system and dynamic S-box. Multimed Tools Appl 76(14):15641–15659
53. Zhang Y-Q, Wang X-Y (2014) Analysis and improvement of a chaotic-based symmetric image encryption scheme using a bit-level permutation. Nonlinear Dyn 77(4):687–698
54. Zhang W, Yu H, Zhao Y-I, Zhu Z-L (2016) Image encryption based on three-dimensional bit matrix permutation. Signal Process 118:36–50
55. Zhang LB, Zhu ZL, Yang BQ, Liu W-Y, Zhug H-F, Zou M (2015) Cryptanalysis and improvement of an efficient and secure medical image protection scheme. Math Probl Eng 2015:1–11
56. Zhou Y, Bao L, Chen CLP (2014) A new 1D chaotic system for image encryption. Signal Process 97:172–182
57. Zhu ZL, Zhang W, Wong KW, Yu H (2011) A chaos-based symmetric image encryption scheme using a bit-level permutation. Inf Sci 181:1171–1186

Yannick Pascal Kamdeu Nkandeu is a holder of a Master degree in Physics from the University of Yaoundé I since 2014. He is currently on a Ph.D. program at the LAGEMES, Ecole Nationale Supérieure Polytechnique, University of Yaoundé I. He is specializing in image encryption.

Alain Tiedeu received his doctorate degree from the University of Yaoundé I, Cameroon, in 1995. He has been teaching electronics, digital signal processing, artificial neural networks, digital image processing, and related subjects at the National Advanced School of Engineering for many years. Professor Tiedeu has also served as reviewer, program committee member, and on editorial advisory board of a number of international conferences and journals (IEEE SITIS conference series, IEEE SETIT conference series, WSEAS conference series, RPBME, etc.). A former regular associate member of the Abdus Salam International Centre for Theoretical Physics, his research interests include biomedical instrumentation and modelling, medical signal and image processing and analysis and image encryption.

Sensing and Imaging (2020) 21:55
https://doi.org/10.1007/s11220-020-00318-y

ORIGINAL PAPER

Image Encryption Algorithm Based on Synchronized Parallel Diffusion and New Combinations of 1D Discrete Maps

Yannick Pascal Kamdeu Nkandeu[1] · Justin Roger Mboupda Pone[2] · Alain Tiedeu[1,3]

Received: 23 January 2020 / Revised: 15 September 2020 / Accepted: 10 October 2020

Abstract

The need for secure communications has triggered research in cryptography in general and image encryption in particular. Papers have been published and many approaches used. One of them is chaos-based image encryption which uses a chaotic map as an essential part of the cryptosystem. For the cryptosystem to be efficient, the chaotic map must exhibit a good chaocity. Hence the need to propose a good chaos generator. In this paper, we set forth a general approach to design a chaotic generator with good properties, using the existing ones. This method was applied and a chaotic map generator obtained. Then an encryption algorithm into which the above chaotic map was combined to the image characteristics, to generate both the encryption keys and the random numbers needed for the encryption process. The encryption procedure consisted in a diffusion process in cipher-block-chaining mode of block image synchronized for parallel computing, followed by a confusion process implemented by means of pixel permutation. The security and robustness tests carried out on the algorithm yielded a high sensitivity to any pixel change or key change and robustness in face of statistical, differential, Chosen known plain /cipher test attacks combined to a fast encryption speed allowing real-time operations.

Keywords Chaotic maps · Encryption · Permutation-diffusion · Parallel computing

✉ Alain Tiedeu
berlin.tiedeu@uy1.uninet.cm

[1] Signal, Image and Systems Laboratory, Department of Medical and Biomedical Engineering, HTTTC (Higher Technical Teachers' Training College), University of Yaoundé I, P.O. BOX 886, Ebolowa, Cameroon

[2] Electrical Engineering Department of IUT-FV, Research Unit of Automation and Applied Computer RU-AIA, University of Dschang, P.O. Box: 134, Bandjoun, Cameroon

[3] Centre for Research, Experimentation and Production (CREP), HTTTC of Ebolowa, PO Box 886, Ebolowa, Cameroon

Published online: 30 October 2020

1 Introduction

Multimedia numerical data exchange has experienced a rapid expansion this last years due to the increasing number of users. This expansion is mainly characterized by a growing variety of data transfer environment and applications: virtual platforms, web sites or android applications. The most significant part of data exchanged are images and videos. They span to domains as diverse as medicine, diplomacy and military, just to name a few. In these domains, the need of secrecy and confidentiality cannot be overemphasized. This encouraged the development of encryption algorithms. Chaotic maps were introduced in cryptography as their intrinsic mathematical characteristics as pseudo-random numbers generators (PRNG) suited the need of reaching a good level of randomness desirable for cryptosystems. Many chaotic map-based cipher systems were designed and published [1–20].

Chaotic image encryption algorithms generally consist of a confusion process followed by (or mixed with) a diffusion process [8]. The confusion process randomly modifies the position (location) of the pixels while the diffusion process alters their grey values. Both processes are achieved with the help of a chaotic equation. Many schemes designed for chaos-based image cryptography simply apply different techniques of one or many rounds of sequential confusion and diffusion process, initialized by a secret key and driven by a chaotic map [9–32, 34].

Ye et al. [9] designed a diffusion function for image encryption using logistic map and an hyperchaotic system, adding an error concept in the initial condition in every round as for self-adaptive modelling. The authors of Ref [10]. proposed another diffusion-permutation architecture in which the Piece Wise Linear Chaotic Map (PWLCM) parameter was pertubed by the output of the Chebyshev map, and the initial conditions were obtained from the message digest 5 (MD5) algorithm of the input image. Abanda et al. [11] mixed the outputs of Colpitt and Duffing chaotic systems for better PRNS, then applied an image encryption based on diffusion-permutation. Bit level permutation applying encryption decision taking on the percentage of contribution of a bit in a grey value, and using 2D (two dimensional) logistic map was proposed by Zhu et al. [12]. Zhang et al. [13] proposed a new approach of that architecture that considers an image of $M \times N$ size with 2^8 greyscale values, as a 3D (three dimensional) bit matrix $M \times N \times 8$ for decisional level of encryption, involving Chen system and 3D cat map. Liu et *al*. [14] lightened the method and proposed a spatial bit-level permutation algorithm for image encryption implemented by scrambling the binary matrix obtained from the input image, and using PWLCM. In 2018, Wu et al. [15] proposed another image encryption architecture based on DNA and a novel 2D Hénon-sine map. He applies random DNA addition, complementary rules, and replication to DNA encoded pairs of bits of the image pixels. Jain et al. [16], on their side, used DNA encoded bits of pixels and PRN (pseudorandom number) generated by logistic map, mixed both, and then applied DNA addition, complementary rules and permutation to obtain cipher image.

A simple but efficient DNA architecture was proposed in Ref. [17]. It randomly applies complementary rules on DNA encoded image pixel by the means of Chebyshev map.

Often, when a system exhibits a suitable enough chaotic behaviour, researchers use it to design S-boxes or P-boxes for image encryption. In this vein, Nkandeu et al. [18] generated many new 1D chaotic structures that were used to encrypt an input image with a combination of S-box and scrambling-masking architecture. Belazi et al. [19] proposed an S-box image encryption with a dynamic key constructed using chebyshev, logistic, tent map and lifting-wavelet transform. Recently, Wang et al. [20]. implemented a fast and robust encryption algorithm taking advantage of parallel diffusion method (instead of streaming diffusion method) and an efficient permutation method, both based on CML (coupled map logistic lattice) system. They also proposed another innovative method in Ref [21]. in which they used a PRNS of a 2D logistic-adjusted-sine-map and a matrix semi-tensor product technique to design an algorithm capable to generate from original ones, an encrypted image or an encrypted Boolean network coded as a Boolean matrix. Very recently, they published an encryption algorithm based on the matrix semi-tensor product where the secret key was generated by a Boolean network [22]. Many other chaotic maps or systems like spatiotemporal chaos [24], improper fractional-order chaotic system [25], chaotic nonlinear adaptive filter [26], Van der Pol-Duffing [27], quantum chaotic map [28], have been used to propose algorithm in image cryptography not only for spatial domain, but also for architecture like neural-network [31], cellular-automata [33] and watermarking [34].

Despite this cloud of cryptosystems designed with care, many have however displayed a low encryption speed [12–17], while others were found with security defects due either to the techniques that were used [35–42], or to setbacks inherent to inadequacy of chaotic maps involved (like logistic map and some other 1D maps) [40–43]. To solve the abovementioned inadequacy, researchers have proposed suitable methods to improve on the chaocity of different chaotic maps for image cryptography. A few of these methods are: generation of new chaotic sequences by mixing output chaotic sequences of two chaotic maps [11]; coupling two chaotic maps sequentially or analytically [15, 20, 21, 33, 44]; modulation combination of two or more chaotic maps for generation of new ones [18, 45]; extension or improvement of an existing chaotic map [23, 46].

From the shortcomings above, in this paper, a new chaos-based cryptosystem has been proposed. Here are some advantages of our scheme:

1. **Fast encryption**: 1D chaotic maps are generally used because they have a simple structure, are easy to implement, their computation take less time and they generate very quick PRNS [47, 48]. In this paper, we proposed three groups of new 1D chaotic maps based on new combination principles, constructed using Logistic, Gompertz, and sine maps as 1D seeds. Metrics like Lyapunov exponents and bifurcation diagrams proved that the combined maps performed better than the seed maps.
2. **Reinforced security:** Generally, the encryption key is used to secure the cryptosystem. In order to improve on the security of our system, image parameters were each time included in the encryption key.

3. **Development of a new combination theorem for seed chaotic maps:** Some researchers have proposed methods to combine chaotic maps such as coupling or cascading two chaotic maps,.switching between multiple chaotic maps, and perturbation of chaotic maps by means of a pseudorandom process. In this work, we developed a new theorem to combine in order to improve on seed maps.

The rest of this paper is organized as follows. Section 2 briefly reports on the chaotic behaviour of some discrete maps. In Sect. 3, new chaotic maps are designed and their chaotic properties are analysed. The image encryption/decryption scheme is described in detail in Sect. 4. Then, in Sect. 5, thorough security analysis of the cryptosystem is carried out. Finally, concluding remarks are presented in Sect. 6.

2 Presentation of 1D Chaotic Maps Used as Seeds

This section reviews the 1D chaotic maps with setbacks since they are seeds or parents for generating new maps.

2.1 Logistic Map

The logistic map is one of the most used chaotic systems, it is analytically written as follows.

$$x_{n+1} = L(\lambda, x_n) = \lambda E_L(x_n) = \lambda x_n\left(1 - x_n\right) \tag{1}$$

where $x_n \in [0, 1]$ is the discrete state of the output chaotic sequence, λ is the control parameter with values in the range [0, 4], and $E_L(x_n) = x_n\left(1 - x_n\right)$, the variable part of the logistic when λ is fixed. The chaotic behaviour of the logistic starts for $\lambda = 3.5$, but output data sequences tend to be really chaotic when λ is near the value of 4 [18].

2.2 Sine Map

The sine map is given by Eq. (2).

$$x_{n+1} = S(\mu, x_n) = \mu E_S(x_n) = \mu \sin(\pi x_n) \tag{2}$$

where $x_n \in [0, 1]$ is the interval of the discrete state of the output chaotic sequence, μ is the control parameter with values in the range [0, 1], and $E_S(x_n) = \sin(\pi x_n)$ is the equation of sine when μ is known. Sine chaotic behaviour stands for $\mu = 0.75$, and fulfilled chaotic properties in the range [0.8, 1] [47 45].

2.3 Gompertz Map

The Gompertz map is another 1D chaotic with great potential for cryptography, but with a low level of chaotic behaviour and properties. Its equation is as follows.

$$x_{n+1} = G(\beta, x_n) = \beta E_G(x_n) = -\beta x_n \ln x_n \tag{3}$$

where $x_n \in [0, 1]$ is the range of discrete state of the output chaotic sequence, the control parameter $\beta \in [0, e]$, where $e = 2.71829$, and $E_G(x_n) = -x_n \ln x_n$ is the Gompertz equation. Chaotic properties appears when $\beta = 2.56$ and are optimum in the range of $[2.67, e]$ [18].

3 New Chaotic Maps Combination Theorem and Examples

In this section, a new combination theorem for 1D chaotic maps is proposed with the objective to establish an explicit mathematical basis of combination principle. Some examples of combined chaotic maps based on this theorem are constructed and analysed.

3.1 Unimodal Chaotic Maps Combination Theorem

Let's consider here only one dimensional chaotic maps with structure similar to $x_{n+1} = C(\alpha, x_n) = \alpha E_C(x_n)$, where α is the control parameter and $E_C(x_n)$ the part of the chaotic map depending only of x_n. The chaotic maps described in Sect. 2 are examples of that structure.

Axiom 3.1. Let $C(\alpha, x_n)$ be a unimodal chaotic map defined in an interval I, with its control parameter α belonging to the interval P, there exists a sub-interval $Q \subset P$ for which this map presents good chaotic dynamical behaviour (Table 1)

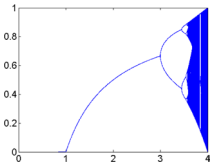
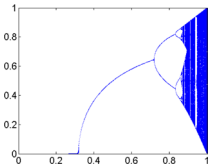
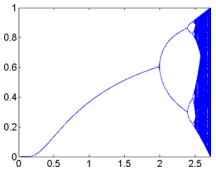
Theorem 3.2. *Let's $C_1(\alpha_1, x_n)$ and $C_2(\alpha_2, x_n)$ be two unimodal chaotic maps, each sending the same interval I to itself. And J,K intervals or sub-intervals containing the control parameters α_1 and α_2 respectively, for which C_1 and C_2 present a good chaotic dynamical behaviour. There exists real numbers a, b, d, e, such that:*

If $\left(a + b \cdot C_2(\alpha_2, x_n)\right) \in J$, Then $\left(a + b \cdot C_2(\alpha_2, x_n)\right) \cdot C_1(\alpha_1, x_n) \cdot \frac{1}{\alpha_1}$ is a unimodal map presenting a good dynamical behaviour in the wide range of α_2, its unique control parameter;

If $\left(e + d \cdot C_1(\alpha_1, x_n)\right) \in K$, Then $\left(e + d \cdot C_1(\alpha_1, x_n)\right) \cdot C_2(\alpha_2, x_n) \cdot \frac{1}{\alpha_2}$ is a unimodal map presenting a good dynamical behaviour in the wide range of α_1, its unique control parameter.

Proof 3.3. The map $\left(a + b \cdot C_2(\alpha_2, x_n)\right) \cdot C_1(\alpha_1, x_n) \cdot \frac{1}{\alpha_1}$ can be written as $\left(a + b \cdot \alpha_2 \cdot E_2(x_n)\right) \cdot \alpha_1 \cdot E_1(x_n) \cdot \frac{1}{\alpha_1}$, which is finally expressed as $x_{n+1} = \left(a + b \cdot \alpha_2 \cdot E_2(x_n)\right) \cdot E_1(x_n)$, with a and b constant values, and α_2 the unique control parameter. For a and b well-chosen, $\left(a + b \cdot C_2(\alpha_2, x_n)\right) \in J$ the sub-interval of values of α_1 for which the map $\alpha_1 \cdot E_1(x_n) = C_1(\alpha_1, x_n)$ has a good chaotic

Table 1 Minimum and maximum Lyapunov exponent (LE) values, and bifurcation diagram of logistic, sine and Gompertz maps.

Chaotic map	Min and Max LE of the combined map	Bifurcation diagram Uniformity *(outcome iteration versus control parameter)*
Logistic	−44.456 and +0,6724	
Sine	−34.623 and +0.6610	
Gompertz	−55.453 and +0.6346	

properties. Consequently,$x_{n+1} = \left(a + b \cdot \alpha_2 \cdot E_2(x_n)\right) \cdot E_1(x_n)$ will tend to have good (Fig. 1)chaotic properties for all values of α_2 in its defined interval.

3.2 Example of Designed System Structures

Given that many chaotic maps can be generated from the combination theorem according to the well-chosen values of a and b (proof 3.3), we will define three types of combination where the illustration of possible maps could result in: (i) Simple chaotic map (SC map) where a, b are constant values; (ii) iteration depending chaotic map (IDC map), where a is constant, b is proportional to $(-1)^n$ and n is the number of iteration. (iii) Computer depending chaotic map (CDC map) with a a constant value and b proportional to a computer random value. Tables 2, 3 and 4 depict a number of new combined map equations. The evaluation of the chaotic behaviour of the new chaos is determined by the Lyapunov exponents (LE) and the bifurcation diagrams indicator.

The Lyapunov exponent (LE) is defined by $LE = \lim_{x\to\infty} \frac{1}{n} \sum_{i=1}^{n} \ln \left| \frac{df(x)}{dx} \right|$: a positive LE of a dynamical system means that the two system trajectories exponentially diverge in each unit time for a small difference of initialization, identifying chaotic behaviour [49].

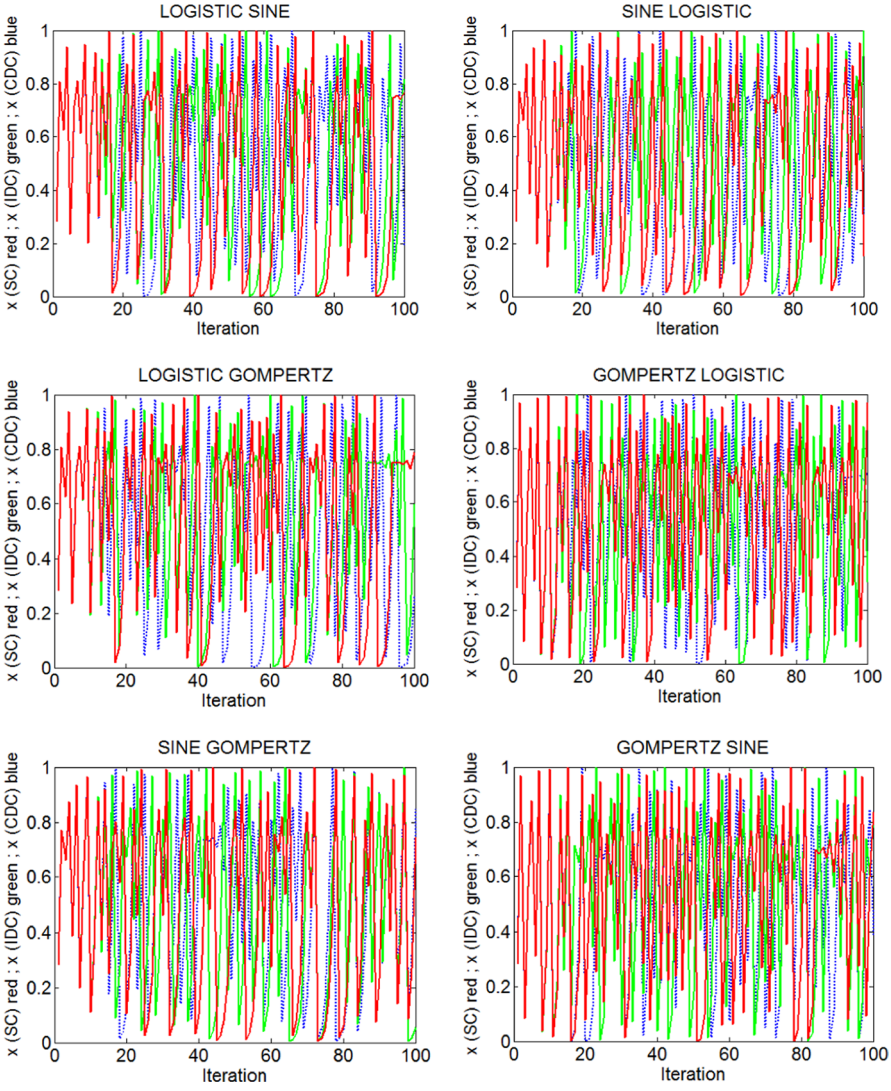


Fig. 1 Lyapunov Exponents graph of the new chaotic maps (For the interpretation of the references to colour in this figure, the reader is referred to the web version of this article)

3.3 Comparison Between The New Maps and Their Parents

The LE of each of the combined maps shown in Tables 2, 3 , 4 and (Fig. 2) has a steady positive values around 0.63 which is nearly constant in the wide range of their parameters [0, 4]. They have better mean values than their seed maps: logistic, sine, and Gompertz (Table 1; Fig. 2) for which LE means are only positive in the lower part of the control parameter set ([3.56, 4], [0.8, 1] and [2.67, e]). The

Table 2 Minimum and maximum lyapunov exponent values, and bifurcation diagrams of combined maps of logistic, sine and Gompertz in SC mode.

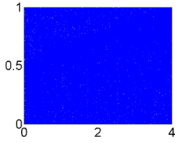
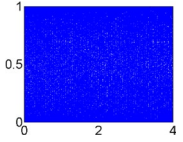
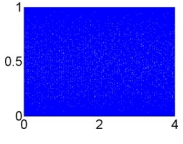
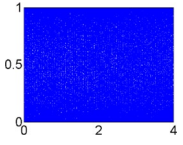
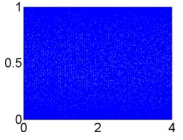
Mode	Combination name	Combination equation	Min and Max LE of the combined map	Bifurcation diagram Uniformity *(outcome iteration versus control parameter)*
SC	Logistic-sine	$a = 3.99; b = 1 \times 10^{-5} : x_{n+1} = (3.99 + 10^{-5}\mu \sin \pi x_n)x_n(1 - x_n)$	+0.3748 and +0.6631	
	Sine-logistic	$a = 0.98; b = 1 \times 10^{-5} : x_{n+1} = (0.98 + 10^{-5}\lambda x_n(1 - x_n)) \sin \pi x_n$	+0.6368 and +0.6526	
	Logistic-gompertz	$a = 3.999; b = 1 \times 10^{-3} : x_{n+1} = (3.999 - 10^{-3}\beta x_n \ln x_n)x_n(1 - x_n)$	+0.6498 and +0.6609	
	Gompertz-logistic	$a = 2.69; b = 1 \times 10^{-5} : x_{n+1} = (2.69 + 10^{-5}\lambda x_n(1 - x_n))x_n \ln x_n$	+0.6103 and +0.6230	
	Sine-gompertz	$a = 0.97; b = 1 \times 10^{-5} : x_{n+1} = (0.97 - 10^{-5}\beta x_n \ln x_n) \sin \pi x_n$	+0.6346 and +0.6526	

Table 2 (continued)

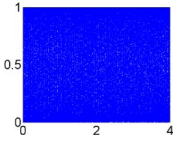
Mode	Combination name	Combination equation	Min and Max LE of the combined map	Bifurcation diagram Uniformity *(outcome iteration versus control parameter)*
	Gompertz-sine	$a = 2.698; b = 1 \times 10^{-4} : x_{n+1} = (2.698 + 10^{-5}\mu \sin \pi x_n) x_n \ln x_n$	+0.6100 and +0.6209	

Table 3 Minimum and maximum lyapunov exponent values, and bifurcation diagrams of combined maps of logistic, sine and Gompertz in IDC mode.

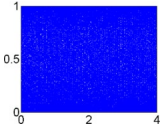
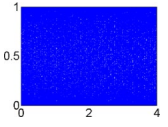
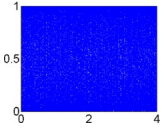
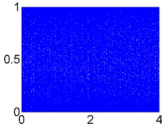
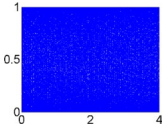
Mode	Combination name	Combination equation	Min-Max LE of the combined map	Bifurcation diagram Uniformity *(outcome iteration versus control parameter)*
mIDC	Logistic-sine	$a = 3.999; b = (-1)^n \times 10^{-4}$: $x_{n+1} = (3.999 + 10^{-4}(-1)^n \mu \sin \pi x_n) x_n (1 - x_n)$	+0.6521 and +0.6613	
	Sine-logistic	$a = 0.999; b = (-1)^n \times 10^{-5}$: $x_{n+1} = (0.99 + 10^{-5}(-1)^n \lambda x_n (1 - x_n)) \sin \pi x_n$	+0.6358 and +0.6518	
	Logistic-gompertz	$a = 3.99; b = (-1)^n \times 10^{-4}$: $x_{n+1} = (3.99 - 10^{-4}(-1)^n \beta x_n \ln x_n) x_n (1 - x_n)$	+0.6521 and +0.6613	
	Gompertz-logistic	$a = 2.69; b = (-1)^n \times 10^{-5}$: $x_{n+1} = (2.69 + 10^{-5}(-1)^n \lambda x_n (1 - x_n)) x_n \ln x_n$	+0.6103 and +0.6203	
	Sine-gompertz	$a = 0.99; b = (-1)^n \times 10^{-5}$: $x_{n+1} = (0.99 - 10^{-5}(-1)^n \beta x_n \ln x_n) \sin \pi x_n$	+0.6366 and +0.6525	

Table 3 (continued)

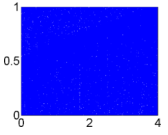
Mode	Combination name	Combination equation	Min-Max LE of the combined map	Bifurcation diagram Uniformity *(outcome iteration versus control parameter)*
	Gompertz-sine	$a = 2.68; b = (-1)^n \times 10^{-5}$: $x_{n+1} = (2.68 + 10^{-5}(-1)^n \mu \sin \pi x_n) x_n \ln x_n$	+0.6077 and +0.6203	

Table 4 Minimum and maximum lyapunov exponent values, and bifurcation diagrams of combined maps of logistic, sine and Gompertz in CDC mode.

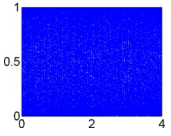
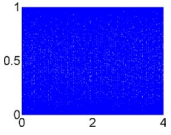
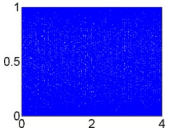
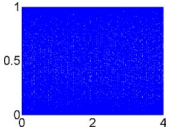
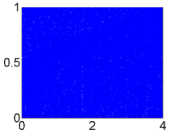
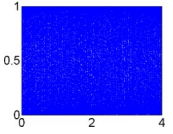
Mode	Combination name	Combination equation	Min and Max LE of the combined map	Bifurcation diagramUniformity *(outcome iteration versus control parameter)*
CDC	Logistic-sine	$a = 3.99; b = rand(1) \times 10^{-5} : x_{n+1} = (3.99 + b\mu \sin \pi x_n)x_n(1 - x_n)$	+0.6517 and +0.6614	
	Sine-logistic	$a = 0.99; b = rand(1) \times 10^{-5} : x_{n+1} = (0.99 + b\lambda x_n(1 - x_n)) \sin \pi x_n$	+0.6361 and +0.6535	
	Logistic-gompertz	$a = 3.99; b = rand(1) \times 10^{-5} : x_{n+1} = (3.99 - b\beta x_n \ln x_n)x_n(1 - x_n)$	+0.6500 and +0.6613	
	Gompertz-logistic	$a = 2.68; b = rand(1) \times 10^{-5} : x_{n+1} = (2.68 + b\lambda x_n(1 - x_n))x_n \ln x_n$	+0.6099 and +0.6210	
	Sine-gompertz	$a = 0.98; b = rand(1) \times 10^{-5} : x_{n+1} = (0.98 - b\beta x_n \ln x_n) \sin \pi x_n$	+0.6361and +0.6524	

Table 4 (continued)

Mode	Combination name	Combination equation	Min and Max LE of the combined map	Bifurcation diagramUniformity *(outcome iteration versus control parameter)*
	Gompertz-sine	$a = 2.69; b = rand(1) \times 10^{-5} : x_{n+1} = (2.69 + 10^{-5} b\mu \sin \pi x_n) x_n \ln x_n$	+0.6105 and +0.6213	

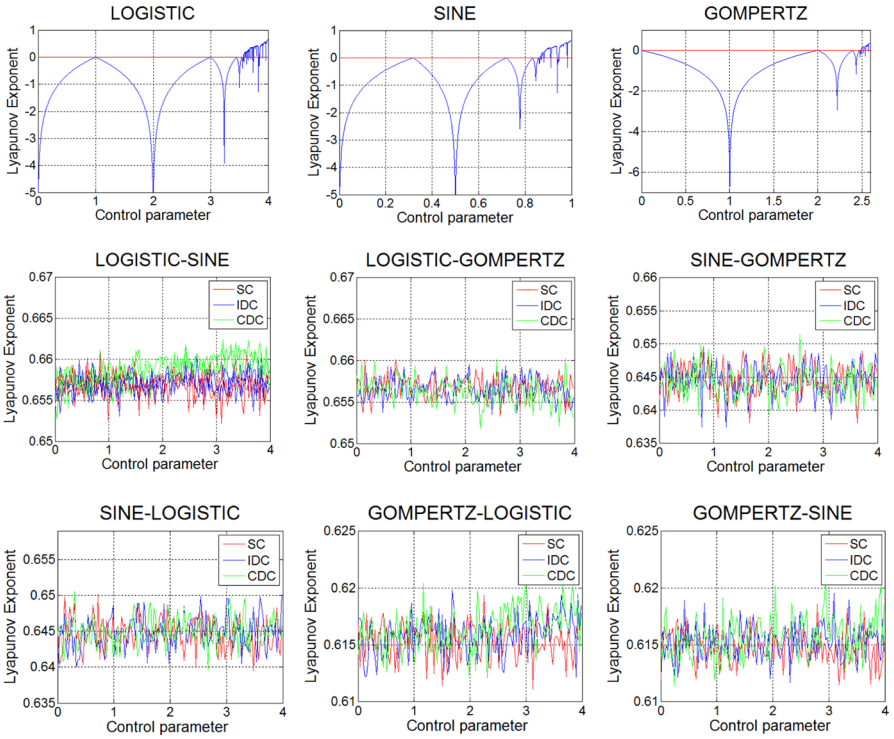


Fig. 2 Plotted of dependency test between different PRNS trajectories of combined chaotic maps for SC(red), IC(green) and CDC(blue) modes, obtained under the same initial condition $x_0 = 0.28$ and control parameter $\lambda = \beta = \mu = 3.5$. (For the interpretation of the references to colour in this figure, the reader is referred to the web version of this article)

positive and nearly constant Lyapunov exponent values of the proposed maps, predicts no periodic windows or drawbacks of any type in their generated PRNS.

Bifurcation diagram draws the asymptotic temporal evolution (or orbit) for a certain set of initial condition, when the control parameter of the discrete-time dynamical system is varied. If the considered dynamical system is a chaotic map, then the derived orbits obtained under any initial condition cover the whole phase space. All the bifurcation diagrams plotted and inserted in Tables 2, 3, 4 for different combined maps are in accordance with the criterion stated above since their diagrams are flat in the wide range of their control parameters [0, 4]. At the opposite, Table 1 presents those of their parents partially filled of blank spaces which materialize thereby weaknesses in their chaotic properties. It can be clearly observed that Logistic, Sine and Gompertz exhibit a weak chaotic state in the partial range of [3.56, 4], [0.8, 1], [2.67, e] respectively. They have a very small range of chaotic properties while the new chaotic maps have good chaotic behaviour for all control parameter values in theirs defined intervals.

Moreover, (Fig. 1) displays many graphics plotting dependency test between SC, IDC, CDC modes; they depict PRNS trajectories superposition of the

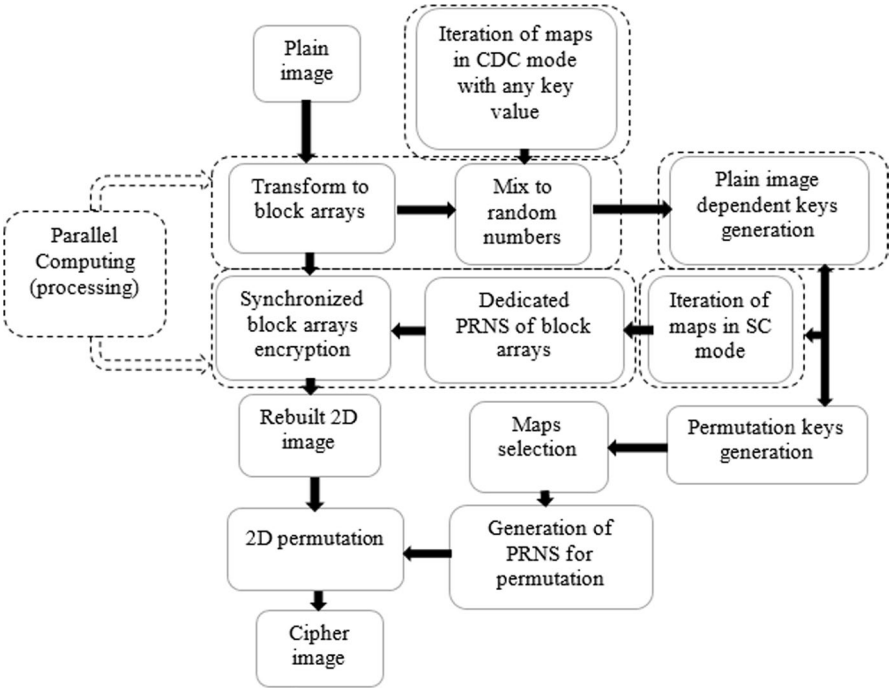


Fig. 3 Block diagram of the encryption scheme

combined maps, generated under the same control parameter and initial condition for different modes. It is clearly observed that beyond the transient effect (ending at approximately 30 iterations), the trajectories of the different PRNS are not correlated proving by then their independency in spite of the light (Fig. 3)difference between their structures.

3.4 Advantages of The New Maps

There exists a great number of couple *(a, b)* for which new combinations can be made giving the possibility of unlimited PRNG (Proof 3.3).

The combined maps have their structures relatively simple and easy to implement as their seed maps, but they have the advantage of a wide range of control parameters ([0, 4] and only positive LE values (a mean of 0.63) in their chaotic properties. They can't therefore exhibit periodic windows or drawbacks like their parents. Furthermore they are appropriate for cryptography application as PRNG because they all succeeded to the (Fig. 4)standard NIST random test (SP 800-22 rev1a test) [44] as reported in Tables 5, 6, 7.

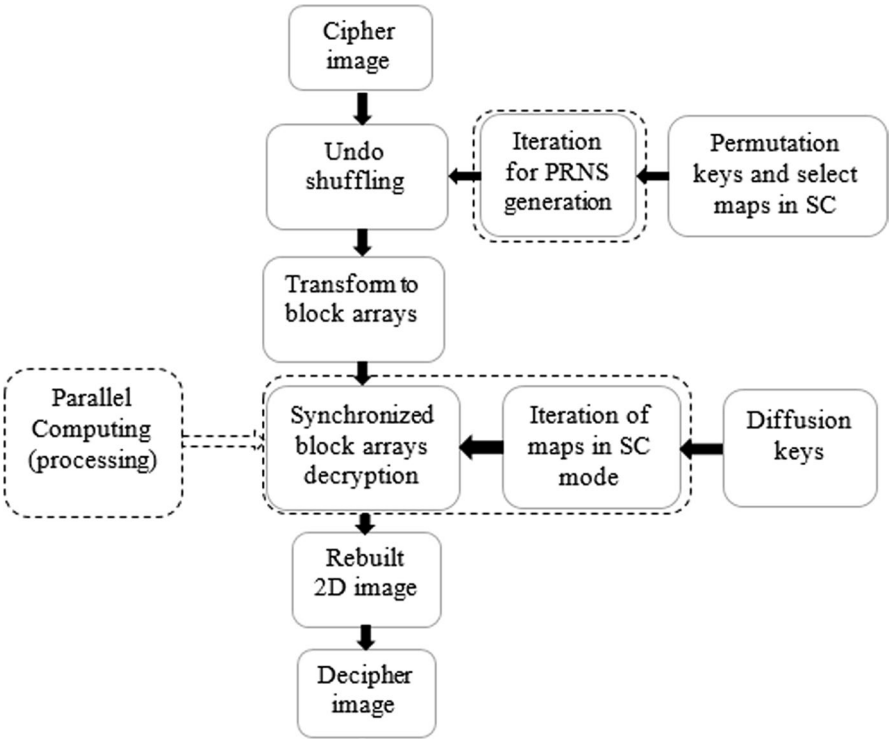


Fig. 4 Block diagram of the decryption scheme

4 The New Encryption/Decryption Procedure

This section presents the new encryption technique built using synchronized parallel-diffusion technique and six combined maps (Logistic-sine, Logistic-gompertz, Sine-gompertz, Sine-logistic, Gompertz-logistic, Gompertz-sine). The synchronized parallel-diffusion is equivalent to parallel processing or computing of operations sequentially independent in the algorithm. During implementation, all PRNS used are considered over the transient effect (over 500 iterations).

4.1 Encryption

Step 1: Transform 2D plain image P of size $W \times H$ to 1D array $p_1 p_2, ..., p_i$ of length $W \cdot H$ with $i = \{1, 2, 3, ..., W \cdot H\}$.

Step 2:

(a) Split the 1D array $p_1 p_2, ..., p_i$ into six arrays of equal size, if $\text{mod}(W \cdot H, 6) = 0$ or five arrays of equal size and one added of $\text{mod}(W \cdot H, 6)$, if $\text{mod}(W \cdot H, 6) \neq 0$. Each block-array obtained will be affected to one of the six combined maps in SC and CDC mode (Table 2 and 4).

Table 5 NIST SP800-22 test results of the proposed chaotic maps in IDC mode

Mode	SC											
Chaotic map	LSM		LGM		SLM		SGM		GLM		GSM	
Test index	*P* value and Decision		*P* value and Decision		*P* value and Decision		*P* value and Decision		*P* value and Decision		*P* value and Decision	
Approximate entropy	0.2800	√	0.0909	√	0.6382	√	0.7349	√	0.1051	√	0.8619	√
Block frequency	0.2301	√	0.9971	√	0.6638	√	0.3234	√	0.5127	√	0.1540	√
Cumulative sums	0.5033	√	0.5268	√	0.2145	√	0.6286	√	0.2526	√	0.6008	√
Fast Fourier transform	0.1476	√	0.1560	√	0.2487	√	0.9014	√	0.4390	√	0.9313	√
Frequency	0.9616	√	0.0407	√	0.7081	√	0.5032	√	0.2870	√	0.4799	√
Random excursions	0.5979	√	0.9897	√	0.4431	√	0.5882	√	0.8356	√	0.2734	√
Random excursions variable	0.8170	√	0.5507	√	0.6212	√	0.0892	√	0.7193	√	0.7116	√
Longest runs of ones	0.7139	√	0.1648	√	0.5335	√	0.5443	√	0.0616	√	0.4365	√
Rank	0.2325	√	0.0431	√	0.1710	√	0.3608	√	0.4577	√	0.2627	√
Runs	0.9380	√	0.1727	√	0.2718	√	0.4561	√	0.3485	√	0.1276	√
Serial	0.6245	√	0.5716	√	0.8363	√	0.1785	√	0.9393	√	0.6358	√
Universal statistical	0.8064	√	0.9795	√	0.5032	√	0.3939	√	0.3478	√	0.3564	√

√, means success; *LSM* Logistic-Sine Map; LGM Logistic-Gompertz Map; *SLM* Sine-Logistic Map; *SGM* Sine-Gompertz Map; *GLM* Gompertz-Logistic Map; *GSM* Gompertz-Sine-Map

Table 6 NIST SP800-22 test results of the proposed chaotic maps in CDC mode.

Mode	CDC											
Chaotic map	LSM		LGM		SLM		SGM		GLM		GSM	
Test index	P-value and Decision		P-value and Decision		P-value and Decision		P-value and Decision		P-value and Decision		P-value and Decision	
Approximate entropy	0.4562	√	0.0818	√	0.5471	√	0.6250	√	0.2178	√	0.7700	√
Block frequency	0.2711	√	0.7170	√	0.7549	√	0.2343	√	0.1138	√	0.2639	√
Cumulative sums	0.2511	√	0.5159	√	0.3256	√	0.7176	√	0.1437	√	0.7119	√
Fast Fourier transform	0.6374	√	0.2659	√	0.1376	√	0.8105	√	0.5289	√	0.8204	√
Frequency	0.2762	√	0.9318	√	0.7061	√	0.6143	√	0.1989	√	0.3680	√
Random excursions	0.7173	√	0.0908	√	0.5542	√	0.4771	√	0.7467	√	0.1625	√
Random excursions variable	0.6712	√	0.6616	√	0.7321	√	0.1981	√	0.8204	√	0.8046	√
Longest runs of ones	0.5135	√	0.0537	√	0.4246	√	0.6354	√	0.1025	√	0.5074	√
Rank	0.4222	√	0.9340	√	0.2629	√	0.2719	√	0.3668	√	0.3738	√
Runs	0.6332	√	0.2636	√	0.3829	√	0.3652	√	0.2594	√	0.2387	√
Serial	0.4456	√	0.4907	√	0.7252	√	0.2696	√	0.8204	√	0.7269	√
Universal statistical	0.3162	√	0.8778	√	0.6121	√	0.4040	√	0.2387	√	0.5979	√

√, means success *LSM* Logistic-Sine Map; *LGM* Logistic-Gompertz Map; *SLM* Sine-Logistic Map; *SGM* Sine-Gompertz Map; *GLM* Gompertz-Logistic Map; *GSM* Gompertz-Sine-Map

Table 7 NIST SP800-22 test results of the proposed chaotic maps in CDC mode.

Mode	SC											
Chaotic map	LSM		LGM		SLM		SGM		GLM		GSM	
Test index	*P* value and Decision		*P* value and Decision		*P* value and Decision		*P* value and Decision		*P* value and Decision		*P* value and Decision	
Approximate entropy	0.1203	√	0.1018	√	0.4253	√	0.7168	√	0.1069	√	0.6611	√
Block frequency	0.3601	√	0.8080	√	0.2059	√	0.1435	√	0.2047	√	0.4512	√
Cumulative sums	0.3433	√	0.6159	√	0.5322	√	0.6065	√	0.5437	√	0.2156	√
Fast Fourier transform	0.7263	√	0.1548	√	0.2467	√	0.9216	√	0.7115	√	0.4175	√
Frequency	0.1665	√	0.4413	√	0.6940	√	0.5054	√	0.1001	√	0.1247	√
Random excursions	0.9195	√	0.2436	√	0.4221	√	0.5860	√	0.3557	√	0.1417	√
Random excursions variable	0.7801	√	0.5515	√	0.5210	√	0.2872	√	0.9600	√	0.4611	√
Longest runs of ones	0.6126	√	0.1326	√	0.3123	√	0.7463	√	0.2523	√	0.4512	√
Rank	0.3311	√	0.1321	√	0.6270	√	0.3628	√	0.6118	√	0.7851	√
Runs	0.7541	√	0.2532	√	0.8228	√	0.2743	√	0.5246	√	0.3627	√
Serial	0.5467	√	0.3609	√	0.1564	√	0.1585	√	0.3601	√	0.1265	√
Universal statistical	0.2471	√	0.7127	√	0.1290	√	0.5151	√	0.1381	√	0.4970	√

√, means success; *LSM* Logistic-Sine Map; *LGM* Logistic-Gompertz Map; *SLM* Sine-Logistic Map; *SGM* Sine-Gompertz Map; *GLM* Gompertz-Logistic Map; *GSM* Gompertz-Sine-Map

(b) From each of the combined CDC maps of Table 4 (and any chosen initial condition and control parameter) generate PRNS with a size equivalent to its affected block-array, then calculate the initial condition (IC_0) and control parameter (CP) values for each block encryption by computing the following equations.

$$k_n = \sum_{i=1}^{W\cdot H/6} \left(RN_i^n + p_i^n/2^8\right) - \left\lfloor \sum_{i=1}^{W\cdot N/6} \left(RN_i^n + p_i^n/2^8\right) \right\rfloor \tag{4}$$

$$IC_0^n = \left(\sum_{n=1}^{6} k_n - k_n \right) \bmod 1 \tag{5}$$

$$CP^n = IC_0^n \times \left(CP_{\max} + 1\right) - IC_0^n \tag{6}$$

where RN_i stands for a generated floating-point random number of rank i in an array of size $W \cdot H/6$ from a CDC map; n is the rank of one of the six blocks, then $n = \{1, 2, 3, 4, 5, 6\}$; k_n is a real value in the interval [0, 1] for a block array of rank n; IC_0^n is the initial condition, and CP^n the control parameter both used to trigger iteration of the new 1D map in SC mode, and affected to the block-array n;$CP_{\max}$ is worth 4 for all the combined maps; the symbol $\lfloor x \rfloor$ is to round the element of x to the nearest integer less than or equal to x; mod is the modulus operator.

Step 3: From the six combined maps in SC mode (Table 2) and with the couples (IC_0^n,CP^n) having the same value of n, generate six arrays of PRNS for each of the six block-arrays above, and synchronize their encryption by simultaneously encrypt them in a single process using the following equation.

$$C_i^n = \left(\left(P_i^n + \left(\left\lfloor RN_i^n \times 10^{15} \right\rfloor\right) \bmod 256\right) \bmod 256\right) \oplus C_{i-1}^n \tag{7}$$

where C_i^n,P_i^n, RN_i^n, are the encrypted data, the plain data and the random number of rank i respectively, they all belong to the block-array n; the symbol $\lfloor x \rfloor$ is to round the element of x to the nearest integer less than or equal to x; mod is the modulus operator and the symbol $\oplus$ denotes bitwise exclusive or operation..

Step 4: After having set the 2D image matrix by combining the six block-arrays, use the equations:

$$IC_H = \left(\sum_{n=1}^{3} k_n \right) \bmod 1 \tag{8}$$

$$IC_V = \left(\sum_{n=4}^{6} k_n \right) \bmod 1 \tag{9}$$

$$CP_H = IC_V \times \left(CP_{\max} + 1\right) - IC_V \tag{10}$$

$$CP_V = IC_H \times \left(CP_{\max} + 1\right) - IC_H \tag{11}$$

$$RCM(j) = \left(\left\lfloor \left(CP_H \oplus CP_V\right) \times 10^{15} \right\rfloor\right) \bmod (7 - j); j = \{1, 2\} \tag{12}$$

And determine the initial condition and control parameter in the horizontal $\left(IC_H, CP_H\right)$ and vertical $\left(IC_V, CP_V\right)$ direction associated to their corresponding randomly chosen maps $(RCM(1), RCM(2))$ among the six.

Step 5: Generate PRNS in each direction and sort them in ascending order, then for each couple row-column of pixel position, find the previous position of the corresponding row-column of the PRNS sorted value and transpose them.

4.2 Decryption Procedure

Step 1: Use the values of IC_H,CP_H, IC_V,CP_V, and the identifier $RCM(1)$ and $RCM(2)$ of the selected maps to invert the shuffling process.

Step 2: Divide the matrix in block arrays as in the encryption process, and use the couples (IC_0^n,CP^n) and their corresponding combined maps in SC mode, then invert the decryption in a single synchronized operation according to the following equation.

$$P_i^n = \left(C_i^n \oplus C_{i-1}^n - \left(\left\lfloor RN_i^n \times 10^{15} \right\rfloor\right) \bmod 256\right) \bmod 256 \tag{13}$$

Step 3: Recombine the block-arrays to form 2D decrypted image.

4.3 Colour Image Encryption/Decryption

Colour image component R, G, B are combined to form a single grey image which is encrypted or/and decrypted according to the steps above, afterward it is split and ordered to obtain the ciphered or deciphered image.

5 Cryptosystem Performance Analysis

An ideal cryptosystem should be able to stand inviolable facing all known attacks such as: statistical attacks of histogram analysis, correlation of adjacent pixels, and information entropy; brute force attack on key space; differential attack;(Fig. 5) cryptanalysis of chosen plain-image and chosen cipher-image attacks.

5.1 Histograms and Variance of Histograms

Histograms of selected original images: x_ray_chest.jpg (253 × 199); Lena.tiff (512 × 512); fingerprint.jpg (220 × 229); x_ray_Skull.jpg (231 × 218), and their ciphered images are presented in Fig. 6 Colour images are also tested and shown in Fig. 5 From both figures, it is observed that the histogram of ciphered image is fairly distributed, suggesting that attacks based on histogram analysis seem impractical.

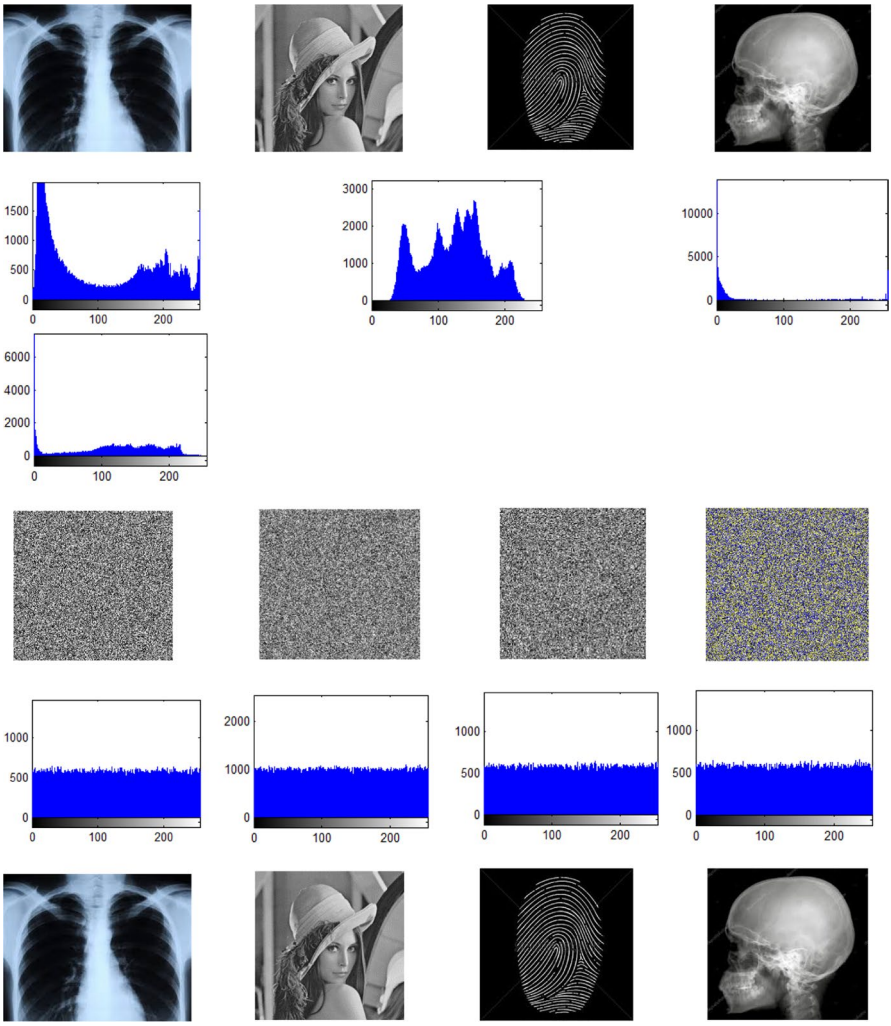


Fig. 5 Encrypted and decrypted grey images and their histograms from the first to the fourth column are: x_ray_chest.jpg (253 × 199); Lena.tiff (512 × 512); fingerprint.jpg (220 × 229); x_ray_Skull.jpg (231 × 218)

However, the variance of the histogram given by Eq. (14)., is a better criterion to evaluate the uniformity of frequencies plotted by a histogram [28].

$$Var(z) = \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} \frac{1}{2}(z_i - z_j)^2 \tag{14}$$

where Z is the vector of the histogram values and $Z = \{z_1, z_2, ..., z_{256}\}$, z_i and z_j are the numbers of pixels which grey values are equal to i and j respectively.

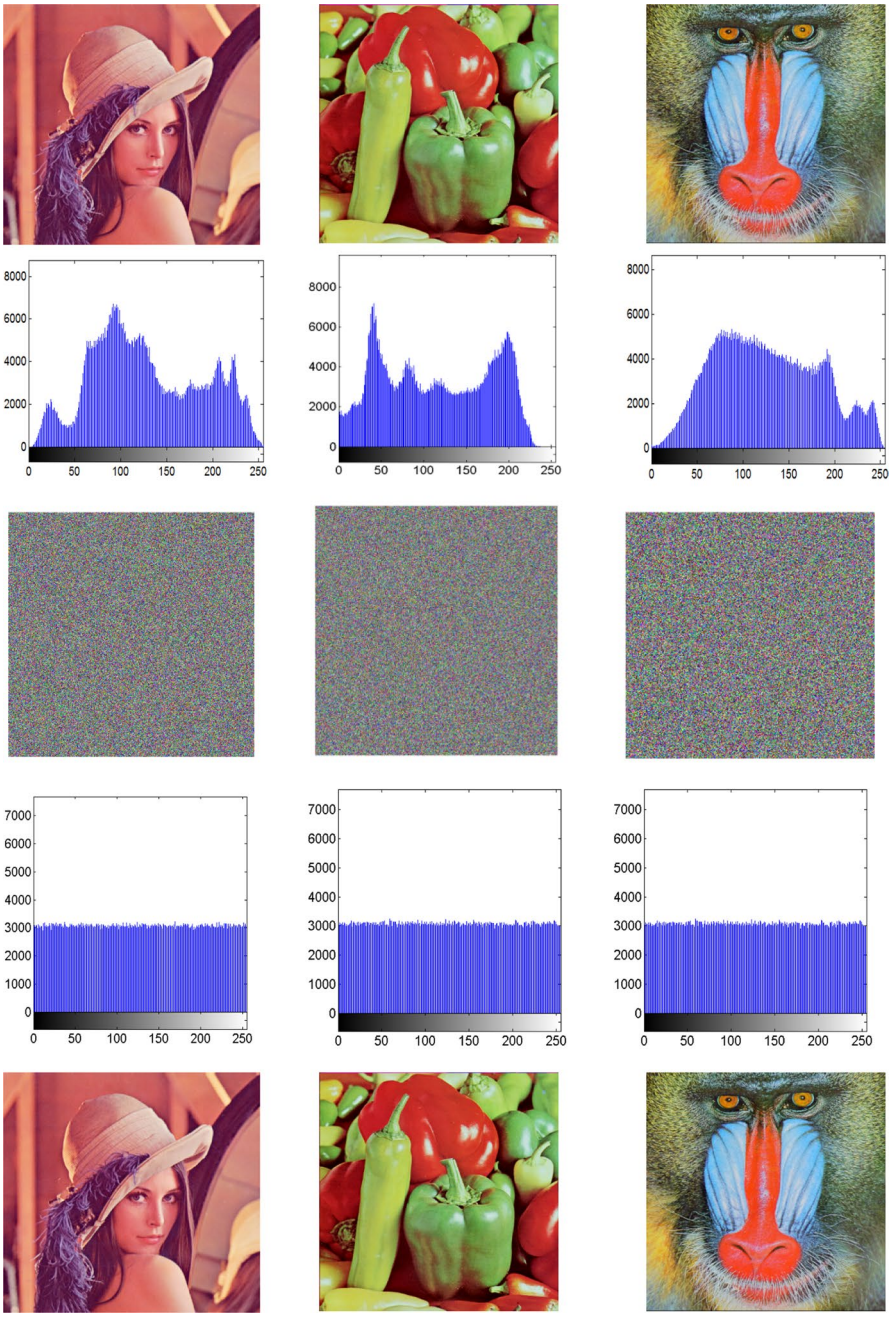


Fig. 6 Encrypted and decrypted colour images and their histograms from the first to the third column are: Lena (512 × 512); peppers (512 × 512); baboon (512 × 512) (For the interpretation of the references to colour in this figure, the reader is referred to the web version of this article)

Table 8 Variance of histograms of some ciphered images

Grey image	Plain image value	Ciphered image value	[15]	[11]
X_ray_chest (256 × 256)	5,129,873.32	982.61	–	–
Lena (512 × 512)	6,255,716.49	1050.87	1027.59	1077.23
Baboon (512 × 512)	6,193,837.23	1002.01	1058.12	971.24
Airport (1024 × 1024)	7,469,849.57	1271.65	–	–

According to [28], variance of histogram values less than 5000 are good enough to characterise a ciphered image presenting a flat histogram. Results recorded in Table 8 from obtained ciphered images are around the values of 1000 and stand nearby good values in literature.

5.2 Correlation Analysis

Breaking down the strong correlation between adjacent pixels of the plain image is necessary to prevent statistical attack. Correlation tests of the proposed encryption algorithm is performed by randomly selecting 5000 pairs of adjacent pixel in horizontal (HC), vertical (VC) and diagonal (DC) direction, then, their correlation coefficients Cr is calculated using the formula.

$$Cr = \frac{K \times \sum_{i=1}^{K} X_i Y_i - \sum_{i=1}^{K} X_i^2 \times \sum_{i=1}^{K} Y_i^2}{\sqrt{\left(K \times \sum_{i=1}^{K} (X_i)^2 - \left(\sum_{i=1}^{K} X_i\right)^2\right) \times \left(N \times \sum_{i=1}^{K} (Y_i)^2 - \left(\sum_{i=1}^{K} Y_i\right)^2\right)}} \tag{15}$$

where X and Y are grey scale values of two adjacent pixels in the image, K is the number of pair of pixels. Cr is the value of correlation belonging to the range [-1,1].

Neighbouring image pixels are lowly correlated when values of Cr are close to 0, and highly correlated for values of Cr close to 1 or –1. Table 9 presents the results of correlation tests of x_ray_chest, lena, baboon, and their ciphered counterparts. It is observed in these results that, in contrast with the plain images which correlation coefficients is about 0.9001, the ciphered images are almost free of any correlation as their correlation coefficients is around a mean of 0.003. This values are proximate to ones yielded by author in [18] and [15]. Furthermore, Fig. 7 shows through the graphics of the first row, how the high correlation in the plain image (Lena) displayed as concentrated dots drawing a thick line, are scattered uniformly on the graphics of the second row as a materialization of low correlation. Any correlation attack on the proposed algorithm could surely not succeed.

Table 9 Correlation coefficient of two adjacent pixels

Image	Size	Test	Plain image	ciphered image	[16]	[13]
X_ray_chest	(256 × 256)	HC	0.9377	– 0.002	–	–
		VC	0.9535	0.010	–	–
		DC	0.9043	– 0.004	–	–
Lena	(512 × 512)	HC	0.9679	0.001	– 0.010	0.003
		VC	0.9845	0.003	0.001	0.001
		DC	0.9580	– 0.006	0.006	0.002
Baboon	(512 × 512)	HC	0.9090	0.002	– 0.029	0.006
		VC	0.8989	– 0.004	– 0.022	0.003
		DC	0.8610	0.008	0.007	0.001

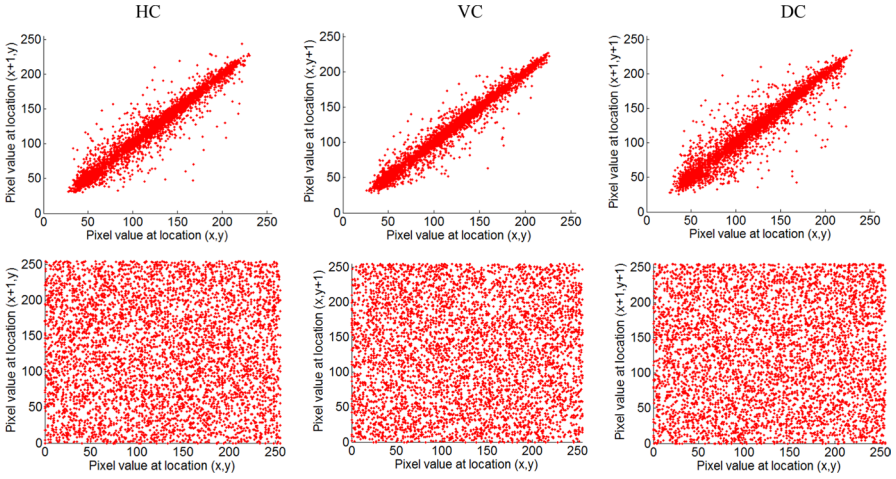


Fig. 7 Pixel values distribution of plain and ciphered Lena

5.3 Entropy Analysis

5.3.1 Information Entropy

Information entropy *H (m)* is the criterion commonly used to measure the level of randomness in a given message *m*. It is expressed by the formula,

$$H(m) = \sum_{i=0}^{2^L-1} p(m_i) \log_2 \left(1/p(m_i)\right) \tag{16}$$

where $p(m_i)$ stands for the probability of symbol m_i, L is the number of bits of the message and 2^L all possible values. For a 256 grayscale image the pixel data has 2^8 possible values, then the ideal entropy of a true random image $H(m) = 8$.

Table 10 Information entropy of some plain images and their ciphered images

Gray image	Plain image	Ciphered image	[30]	[18]	[15]
X_ray_chest (256 × 256)	7.0097	7.9981	–	–	–
Lena (512 × 512)	7.4455	7.9994	7.9993	7.9994	7.9974
Baboon (512 × 512)	7.5714	7.9993	–	7.9993	7.9992
Peppers (512 × 512)	7.3013	7.9994	7.9993	7.9993	7.9993
Airport (1024 × 1024)	7.5235	7.9998	–	7.9998	–

Table 11 Localized information entropy of some plain images and their ciphered images

Gray image	Plain image	Ciphered image	[22]	[35]
Lena (512 × 512)	7.36894	7.90225	7.90273	–
Baboon (512 × 512)	7.07223	7.90183	7.90185	7.902278
Goldhill (512 × 512)	6.69691	7.90250	7.90205	–
Airport (1024 × 1024)	5.689569	7.902274	–	–7.902184

Table 10 reports the entropy values of some images encrypted by the proposed encryption algorithm. They are very close to 8 as expected, and slightly better than the ones in literature [15, 18, 30].

5.3.2 Local Information Entropy

According to author [50], the local information entropy is more accurate to evaluate the uniformity of pixel distribution. The equation implemented for that purpose is define as follow:

$$\overline{H_{(k,T_B)}(m)} = \sum_{i=1}^{k} \frac{H(m_i)}{k} \tag{17}$$

where S_i are image block with T_B number of pixels, randomly selected k-times from the image, and $H(m_i)$ is the information entropy as defined in Eq. 16. Author in [50] states that the ideal value of this metric is worth 7.902469317, and acceptable values are within the range [7.901722822, 7.903215812]. In Table 11 we reported values obtained after simulations. These values are within the good range, and are similar to the ones obtained by [20, 33] as result of a good randomness of pixel value distributions in a cipher image.

5.4 Key Space

The key space must be large enough to resist brute force attack. The encryption key is made of eight couples of initial conditions and control parameters ((IC_0^1,CP^1), (IC_0^2

,CP^2), (IC_0^3,CP^3), (IC_0^4,CP^4), (IC_0^5,CP^5), (IC_0^6,CP^6), (IC_H,CP_H), (IC_V,CP_V)) set in the range of $[0, 1]$ and $[0, 4]$ respectively. With a decimal precision set at 10^{-15} the key space tend to $10^{120} \approx 2^{400}$ which is largely superior to $10^{40} \approx 2^{128}$ considered as large enough to resist brute force attack [23, 31, 32].

5.5 Key Sensitivity

A low key sensitivity has the consequence of possibility of weak and equivalent keys for a cryptosystem. In the opposite side, a high key sensitivity results of high sensitive characteristics of the chaotic map is being used. The key of the cryptosystem is designed by mixing computer dependent chaotic maps PRNS and plain image pixels. Tables 12 , 13 prove that diffusion keys IC (initial condition) and CP (control parameter) are highly fluctuating at each new execution of the algorithm for the same input image, or to the less significant bit (LSB) change of a pixel. Given that the decryption starts by the reverse of permutation, we modified one bit of its decryption keys and obtained $k_1 = IC_v + 10^{-15}$ for the first modification; $k_2 = IC_H + 10^{-15}$ for the second; and $k_3 = CP_v + 10^{-15}$ for the third, then the encrypted image Lena is decrypted with the modified versions (k_1, k_2, k_3). The decrypted images still confuse as shown in Fig. 8 It is therefore certain that encryption keys are highly sensitive to prevent weak and equivalent keys.

5.6 Differential Attack

The differential attack principle is used to find out the difference between two ciphered images encrypted using two images differentiated by one bit or one pixel change. The criterions of number of pixels change rate (NPCR) and unified average changing intensity (UACI) [14] are usually applied to examine the performance of resistance against differential attack.

$$NPCR = \frac{\sum_{i,j} D(i,j)}{W \times H} \times 100\% \tag{18}$$

$$UACI = \frac{1}{W \times H}\left[\sum_{i,j} \frac{|C_1(i,j) - C_2(i,j)|}{255}\right] \times 100\% \tag{19}$$

where C_1 and C_2 are two images with same size $W \times H$. If $C_1(i,j) \neq C_2(i,j)$ then $D(i,j) = 1$, otherwise, $D(i,j) = 0$.

Table 14 displays the measurement of NCPR and UACI between two ciphered images of some images, when a less significant bit (LSB) changes on grey value in the first, middle, or last pixel position. It can be noticed that values obtained are around the mean of 99.62 for NCPR, and 33.51 for UACI. These values are very close to the good ones proposed in literature [15, 18] and reported in Table 15. A significant result of NCPR and UACI is obtained for unchanged image encrypted

Table 12 Diffusion keys sensitivity after a new execution on the same image

Image component	Keys	Cryptosystem First execution	Cryptosystem Second execution	Cryptosystem third execution
Logistic-sine	CP	0.050806527028044	3.462748715203475	2.452589242210649
	IC	0.012701631757011	0.365687178800869	0.613147310552662
Sine-logistic	PC	3.366920858719027	3.101109167352206	2.888531953059669
	IC	0.841730214679757	0.275277291838052	0.722132988264917
Logistic-gompertz	CP	2.789479515359403	3.720867178257322	1.780973390869974
	IC	0.397369878839851	0.930216794564330	0.445243347717494
Gompertz-logistic	CP	1.148046892358934	3.367484139204294	2.766298271822222
	IC	0.537011723089734	0.841871034801073	0.691574567955556
Sine-gompertz	CP	3.393514776813618	2.018480117100694	1.703620377560753
	IC	0.848378694203404	0.504620029275173	0.925905094390188
Gompertz-sine	CP	0.122746011364143	1.791447110916124	2.415711388678631
	IC	0.050806527028044	0.447861777729031	0.603927847169658

Table 13 Diffusion keys sensitivity with LSB change on the same original image

Image component	Keys	LSB change on the First pixel	LSB change on the middle pixel	LSB change on the last pixel
Logistic-sine	CP	3.434934158025442	3.702306439307563	1.880533152349187
	IC	0.858733539506360	0.925576609826891	0.470133288087297
Sine-Logistic	PC	1.505422705658816	2.516311042686766	0.042904682874564
	IC	0.376355676414704	0.629077760671692	0.010726170718641
Logistic-gompertz	CP	0.831459540569597	2.171152601390077	1.890857657688855
	IC	0.207864885142399	0.542788150347519	0.472714414422214
Gompertz-logistic	CP	2.931050779229793	3.830929328898492	1.987032747016599
	IC	0.732762694807448	0.957732332224623	0.496758186754150
Sine-gompertz	CP	0.772367975138707	0.692313440530029	2.868294796643966
	IC	0.193091993784677	0.173078360132507	0.717073699160991
Gompertz-sine	CP	2.576872921111260	2.509619371139820	0.884947860704699
	IC	0.644218230277815	0.627404842784955	0.221236965176175

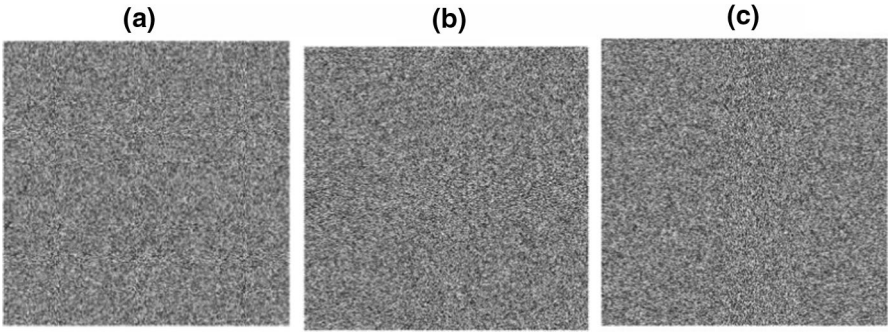


Fig. 8 Confuse image Lena deciphered with (k_1, k_2, k_3): **a** deciphered Lena with k_1, **b** deciphered Lena with k_2, **c** deciphered Lena image with k_3

Table 14 Proof of image sensitivity to a LSB change

Image	Test	Original image after new run	LSB change on the first pixel	LSB change onthe middle pixel	LSB change on the last pixel
Cameraman.tif	NCPR	99.60	99.59	99.63	99.62
	UACI	33.45	33.33	33.51	33.42
X_ray_chest.jpg	NCPR	99.61	99.62	99.60	99.61
	UACI	33.39	33.35	33.44	33.50
X_ray_skull.jpg	NCPR	99.59	99.63	99.61	99.61
	UACI	33.56	33.52	33.43	33.48
Airport.tiff	NCPR	99.62	99.59	99.63	99.62
	UACI	33.50	33.43	33.38	33.45
Finger_print.jpg	NCPR	99.62	99.62	99.60	99.63
	UACI	33.45	33.35	33.47	33.39
Fruit.bmp	NCPR	99.59	99.60	99.61	99.61
	UACI	33.36	33.52	33.39	33.40

Table 15 Comparison of NCPR and UACI measures with proposed values in literature

Image component	Test	Proposed algorithm	[15]	[18]
Lena	NCPR	99.63	99.63	99.59
	UACI	33.52	33.31	33.50
Mandrill	NCPR	99.61	99.60	99.60
	UACI	33.55	33.34	33.52
Peppers	NCPR	99.64	99.61	99.61
	UACI	33.45	33.43	33.52

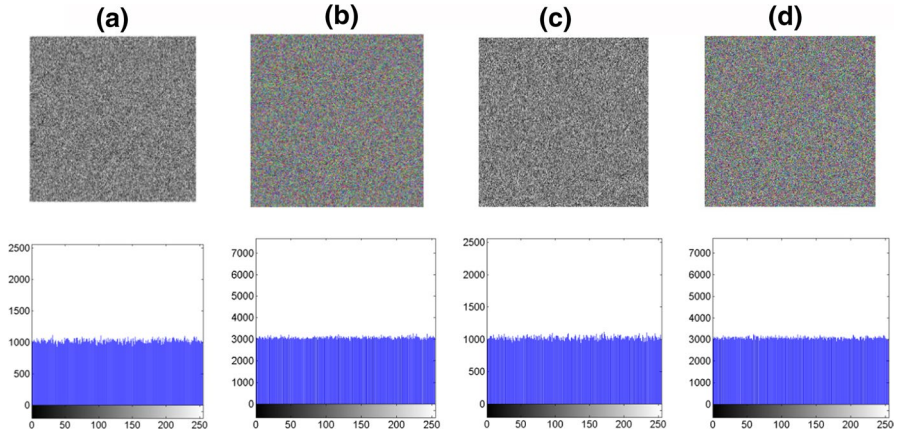


Fig. 9 Chosen plain-image and chosen cipher-image attacks: **a** result of chosen plain-image attack on Baboon, **b** result of chosen plain-image attack on colour Lena, **c** result of chosen cipher-image attack on Baboon, **d** result of chosen cipher-image attack on colour Lena

after a new execution of the algorithm (Table 14), this result is a proof that the cryptosystem gives out a totally different encrypted image, from the same original after each new run. Differential attacks on the proposed cryptosystem is completely impossible with regards to the previous consideration.

5.7 Chosen Plain-Text Attack (CPA)/Chosen Cipher-Text Attack (CCA) Cryptanalysis

It is shown in refs. [43, 49] that the CPA is the most powerful attack among classical ones. In its procedure, the attacker has obtained temporary access to the encryption machinery. Hence he can choose a plain-text string, and construct the corresponding cipher-text string. He can try for example to extract a subkey sequence using a plain and cipher version counterpart of a null-image (or all-one image). Then the subkey is used to recover a target plain image $P^{i,j}$ from its ciphered image $C^{i,j}$ (Eq. 20).

$$P^{i,j} = C^{i,j} \oplus \left(M^{i,j} \oplus D^{i,j}\right) \tag{20}$$

where $M^{i,j} = \bigcup_n m_n^{i,j} = 000\ldots$ is a null-image (or all one-image), and $D^{i,j} = \bigcup_n d_n^{i,j} = d_1 d_2 d_3 \ldots$ its corresponding ciphered image with the same size as $P^{i,j}$, (i, j) denotes the 2D positions of the pixel.

On the other hand, the chosen cipher-attack is possible when an attacker, in the same condition as for CPA attack, possesses a ciphered image made of null-image (or all one-image $D^{i,j}$), and constructs its corresponding plain image $M^{i,j} = \bigcup_n m_n^{i,j}$ [51]. He uses both to determine the key-stream necessary to recover a plain image $P^{i,j}$ from its ciphered version image $C^{i,j}$ according to Eq. 20.

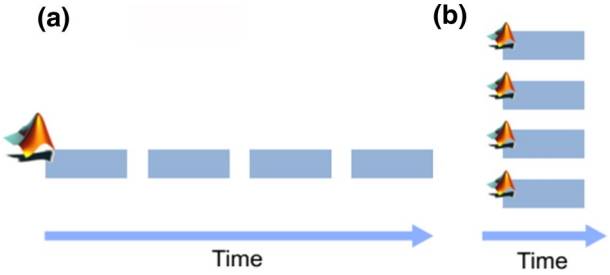


Fig. 10 Types of computation architecture. From left to right; serial computation and parallel computation. Every blue rectangle represents a single and independent computation process. (For the interpretation of the references to colour in this figure, the reader is referred to the web version of this article)

Figure 9a, b, c and d are results of CPA (Fig. 9a, b) and CCA (Fig. 9c, d) carried-out on grey Baboon and colour image Lena. In both cases, recovering images are still confused.

CPA and CCA attacks can't succeed, because each execution of the encryption algorithm produces a totally different sequence of the diffusion sub-key values, same as shuffling keys. Moreover an encryption algorithm which can resist CPA attack can also resist to the rest of attacks as demonstrated in [51].

5.8 Encryption Time Analysis

5.8.1 Computational Complexity of the Proposed Scheme

The analysis of the computational complexity of the proposed cryptosystem is founded on encryption process complexity. First of all, the initial conditions and control parameters are calculated using six arrays of length $(W \cdot H/6)$, then the diffusion in the next step operates on the six arrays in the same time, while the permutation process doesn't require any calculation apart from iteration. From this estimation, we can set the total time complexity of the proposed scheme to $\Theta(W \cdot H/6)$ for a computer using multiple-core processor platform, and in the worst case to $\Theta(W \cdot H)$ if the algorithm is run on a single-core processor platform. These times complexity are both better that those of [28, 30, 45] claimed to be $\Theta(4 \cdot W \cdot H)$, $\Theta(24 \cdot W \cdot H)$, $\Theta(100 \cdot W \cdot H)$ respectively.

5.8.2 Encryption Speed

Time consumption was carried under windows 8 operating system, Intel (R) Core (TM) i5–2430 M CPU @ 2.40 GHz and 8 GB RAM. Parallelizing tasks architecture was part of implementation through the use of parfor-loops for assignment reductions in Matlab 2012 (a) platform [52].

Traditionally, encryption algorithm are written for serial computation (Fig. 10a) which doesn't take advantages of multi-core processors. In that configuration, only one instruction may be executed at a time. Parallel computing on the other hand,

Table 16 Encryption time in milliseconds (ms)

Image	Size	Type	Proposed algorithm	[22]	[18]	[53]
Cameraman	(256×256)	Grey	112	117	195	223 ms
Lena	(512×512)	Grey	402	274	650	–
Airport	(1024×1024)	Grey	1200	789	2897	–
Lena	(512×512)	Colour	983	–	2100	–

offers the possibility to use multiple processing elements simultaneously, to solve a problem by breaking it into independent parts. Therefore, each processing core element can execute its part of the algorithm synchronously with the others (Fig. 10b), and consequently speed-up the encryption.

Table 16 reports the time in milliseconds (ms) spent by the proposed cryptosystem to encrypt some images, and its comparison to some fast cryptosystem in literature. The encryption time is better for [18] and [51] but smaller than [21], except for cameraman.

5.9 Advantages of the Proposed Scheme and Comparison with Other Cryptosystem

The algorithm proposes a simple and efficient method for key generation thoroughly dependent on plain image characteristics. Since the chaotic maps used have excellent chaotic properties, the key space is large enough (Sect. 5.2) to prevent the usage of equivalent keys due to the cross-combination of initial conditions and control parameters.

The designed encryption algorithm yields a totally different ciphered image each time it encrypts the same original image with the same initialisation keys of the new maps in CDC mode (Table 14). This trick makes all types of differential, chosen plain and cipher-image attacks impossible (Sect. 5.6, 5.7).

Pixel block arrays are independent during the diffusion step in CBC mode, thus the propagation error which is an error in a pixel spreading from one pixel to another is thereby limited in a block or in each block.

Our proposed method is based on a diffusion-permutation architecture as the proposed method in Ref.[11, 20, 33], while the author in Ref [18]. have proposed a scrambling-masking combined S-box method and the one in Ref [20]. a DNA architecture. Their algorithms were designed with enhanced chaotic maps as PRNG. Results obtained from our evaluation metrics of variance of histogram, correlation, entropy, NCPR, UACI, and depicted in Tables 8, 9, 10, 11, 15 respectively. They have similar or better values compared to those of the authors above. Furthermore our method generated a larger key space than ones in Ref [21, 29, 30]. (see Sect. 5.4), and a better time complexity than those of ref [29, 31, 48]. (see Sect. 5.8.1). The encryption/decryption time, implemented taking advantage of independency of different blocks, have better values (see Table 16) than those in Ref. [18, 53].

6 Conclusion

In this paper, a new combination theorem of 1D chaotic maps was proposed and used as a tool for generation of new multiple 1D chaotic maps of different modes (SC, IDC, CDC). They exhibit a very good chaotic properties certified by Lyapunov exponents and bifurcation diagram, and were generated from logistic, sine and Gompertz maps proven to have some defects. The example of maps designed are used in a new encryption algorithm built in a diffusion-permutation architecture. The key encryption for diffusion and permutation are all extracted from both PRNS of the chaotic maps and image. The diffusion process occurs in many independent block arrays of image pixels as many chaotic maps are concerned, and in a synchronized way in CBC mode followed by a pixel shuffling. Security tests of brute force attack, differential attack, CPA and CCA, are demonstrated to be inefficient as the algorithm employs a large number of maps of different types, some depending on a random state of the computer. Other tests like variance of histogram, correlation analysis, and entropy gave out results which attest that statistical attacks will fail since the algorithm inherits excellent statistical properties of the designed maps. Finally, the time consumption is very low because of parallel computing implementation, and really suggests the possibility of a true multimedia application.

Funding This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Ethical approval This article does not contain any studies with human participants performed by any of the authors.

References

1. Chen, G., Mao, Y., & Chui, C. K. (2004). A symmetric image encryption scheme based on 3d chaotic cat maps. *Chaos Solitons and Fractals, 21*(3), 749–761.
2. Fridrich, J. (1998). Symmetric ciphers based on two-dimensional chaotic maps. *International Journal of Bifurcation and Chaos, 8*(6), 1259–1284.
3. Ghebleh, M., Kanso, A., & Noura, H. (2014). An image encryption scheme based based on irregulary decimated chaotic maps. *Signal Processing: Image Communication, 29*(5), 618–627.
4. Wang, X., Liu, L., & Zhang, Y. (2015). A Novel Chaotic block image encryption algorithm based on dynamic random growth technique. *Optics and Lasers in Engineering, 66,* 10–18.
5. Matthews, R. (1989). On the derivation of a chaotic encryption algorithm. *Cryptologia XIII, London, 1,* 29–42.
6. Wu, Y., Zhou, Y., Saveriades, G., Agaian, S., Noonan, J. P., & Natarajan, P. (2013). Local Shannon entropy measure with statistical tests for image randomness. *Information Sciences, 222*(10), 323–342.

7. Jakimoski, G., & Koracev, L. (2001). Chaos and cryptography: block encryption ciphers based on chaotic Maps IEEE Transactions on circuits and systems 1. *Fundamental Theory and Applications, 48*(2), 163–169.
8. Schneier, B. (1996). *Applied cryptography-protocols, algorithms, and source code in C* (2nd ed., p. 1996). Hoboken: Wiley.
9. Ye, G., & Zhou, J. (2014). A block chaotic image encryption scheme based on self-adaptive modelling. *Applied Soft Computing, 22,* 351–357.
10. Liu, H., & Wang, X. (2010). Color image encryption based on one-time keys and robust chaotic maps. *Computers and mathematics with application, 59,* 3320–3327.
11. Abanda, Y., & Tiedeu, A. (2016). Image encryption by chaos mixing. *IET Image Processing, 10*(10), 742–750.
12. Zhu, Z. L., Zhang, W., Wong, K. W., & Yu, H. (2011). A chaos-based symmetric image encryption scheme using a bit-level permutation. *Information Sciences, 181,* 1171–1186.
13. Zhang, Y., & Wang, X. (2014). Analysis and improvement of a chaotic-based symmetric image encryption scheme using a bit-level permutation. *Nonlinear Dynamic, 77*(4), 687–698.
14. Liu, H., & Wang, X. (2011). Color image encryption using spatial bit-level permutation and high-dimension chaotic system. *Optics communication, 284,* 3895–3903.
15. Wu, J., Liao, X., & Yang, B. (2018). Image encryption using 2D Hénon-Sine map and DNA approach. *Signal processing, 153,* 11–23.
16. Jain, A., & Rajpal, N. (2015). A robust image encryption algorithm resistant to attacks using DNA and chaotic logistic maps. *Multimedia Tools and Applications, 75*(10), 5455–5472.
17. Liu, H., Wang, X., & Kadir, A. (2012). Image encryption using DNA complementary rule and chaotic maps. *Applied Soft Computing, 12,* 1457–1466.
18. Nkandeu, K. P. Y., & Tiedeu, A. (2019). An image encryption algorithm based on substitution technique and chaos mixing. *Multimedia Tools and Applications, 78*(8), 10013–10034.
19. Belazi, A., El-Latif, A. A. A., Diaconu, A.-V., Rhouma, R., & Belghith, S. (2017). Chaos-based partial image encryption scheme based on linear fractional and lifting wavelet transforms. *Optics and Lasers in Engineering, 88,* 37–50.
20. Wang, X., Feng, L., & Zhao, H. (2019). Fast image encryption algorithm based on parallel computing system. *Information sciences, 486,* 340–358.
21. Wang, X., & Gao, S. (2020a). Image encryption algorithm for synchronously updating Boolean networks based on matrix semi-tensor product theory. *Information sciences, 507,* 16–36.
22. Wang, X., & Gao, S. (2020b). Image encryption algorithm based on the matrix semi-tensor product with a compound secret key produced by a boolean network. *Information sciences, 539,* 195–214.
23. Wang, X., Yang, L., Lui, R., & Kadir, A. (2010). A chaotic image encryption algorithm based on perceptron model. *Nonlinear Dynamic, 62,* 615–621.
24. Song, C., Qia, Y., & Zhang, X. (2013). An image encryption scheme based on new spatiotemporal chaos. *Optik, 124,* 3329–3334.
25. Zhao, J., Wang, S., Chang, Y., & Li, X. (2015). A novel image encryption scheme based on an improper fractional-order chaotic system. *Nonlinear Dynamic, 80*(4), 1721–1729.
26. Hsiao, H., & Lee, J. (2015). Color image encryption using chaotic nonlinear filter. *Signal Processing, 117,* 281–309.
27. Volos, C. K., Kyprianidis, I. M., Stouboulos, I., & Pham, V. T. (2015). Image encryption scheme based on non-autonomous chaotic systems. In N. Daras & M. Rassias (Eds.), *Computation, Cryptography, and Network Security* (pp. 591–612). Cham: Springer.
28. Seyedzadeh, S. M., Norouzi, B., Mosavi, M. R., & Mirzakuchaki, S. (2015). A novel color image encryption algorithm based on spatial permutation and quantum chaotic map. *Nonlinear Dynamic, 81*(2), 511–529.
29. Wang, X., & Zhang, Y. (2014). A symmetric image encryption algorithm based on mixed linear–nonlinear coupled map lattice. *Information Sciences, 273,* 329–351.
30. Wang, X., Zhang, Y., & Bao, X. (2015). A novel chaotic image encryption scheme using DNA sequence operations. *Optics and Lasers in Engineering, 73,* 53–61.
31. Wang, X., & Li, Z. (2019). A color image encryption algorithm based on Hopfield chaotic neural network. *Optics and Lasers in Engineering, 115,* 107–118.
32. Zhang, Y., & Wang, X. (2015). A new image encryption algorithm based on non-adjacent coupled map lattices. *Applied Soft Computing, 26,* 10–20.
33. Chen, R. J., & Lai, J. L. (2007). Image security system using recursive cellular automata substitution. *Pattern Recognition, 40,* 1621–1631.

34. Shao, Z., Shang, Y., Zhang, Y., Liu, X., & Guo, G. (2016). Robust watermarking using orthogonal fourier-mellin moments and chaotic map for double images. *Signal processing, 120,* 522–531.
35. Sharma, M. (2020). Image encryption based on a new 2D logistic adjusted logistic map. *Multimedia Tools and Applications, 79,* 355–374.
36. Wang, X., Zhang, Y., & Liu, L. (2016). An enhanced sub-image encryption method. *Optics and Lasers in Engineering, 86,* 248–254.
37. Liu, H., & Liu, Y. (2014). Cryptanalyzing an image encryption scheme based on hybrid chaotic system and cyclic elliptic curve. *Optics & Laser Technology, 56,* 15–19.
38. Bechikh, R., Hermassi, H., El-Latif, A. A. A., Rhouma, R., & Belghith, S. (2015). Breaking an image encryption scheme based on a spatiotemporal chaotic system. *Signal Processing: Image Communication, 39,* 151–158.
39. Fan, H., Li, M., & Liu, D. (2018). Cryptanalysis of a plaintext-related chaotic RGB image encryption scheme using total plain image characteristics. *Multimedia Tools and Applications, 77,* 20103–20127.
40. Zhang, X., Nie, W., & Ma, Y. (2017). Cryptanalysis and improvement of an image encryption algorithm based on hyper-chaotic system and dynamic S-box. *Multimedia Tools Applications, 76*(14), 15641–15659.
41. Arroyo, D., Alvarez, G., Fermandez, V. (2008). On the inadequacy of the logistic map for cryptographic applications. *arXiv:0805.4355v1[nlin.CD]* 28 May 2008.
42. Arroyo, D., Alvarez, G., Fermandez, V. (2008). A basic framework for the cryptanalysis of digital chaos-based cryptography. *arXiv:0811.1859v1[cs.CR]* 12 Nov 2008.
43. Li, C., Li, S., & Muhammad, A. (2009). On the security defects of an image encryption scheme. *Image Vision Computing, 27*(9), 1371–1381.
44. Li, C., Li, S., & Lo, K. (2011). Breaking a modified substitution–diffusion image cipher based on chaotic standard and logistic maps. *Communication in Nonlinear Science and Numerical Simulations, 16,* 837–843.
45. Sam, I. S., Devaraj, P., & Bhuvaneswaran, R. S. (2014). An efficient quasigroup based image encryption using modified nonlinear chaotic maps. *Sensing and Imaging, 15,* 92.
46. Zhou, Y., Bao, L., & Chen, C. L. P. (2014). A new 1D chaotic system for image encryption. *Signal Processing, 97,* 172–182.
47. Pak, C., & Huang, L. (2017). A new color image encryption using combination of the 1D chaotic map. *Signal Processing, 138,* 129–137.
48. Patidar, V., Pareek, N., Purohit, G., & Sud, K. (2010). Modified substitution–diffusion image cipher using chaotic standard and logistic maps. *Communication in Nonlinear Science and Numerical Simulations, 15,* 2755–2765.
49. Chai, X., Fu, X., Gan, Z., Lu, Y., & Chen, Y. (2019). A color image cryptosystem based on dynamic DNA encryption and chaos. *Signal processing, 155,* 44–62.
50. Kepner, J. (2009). Parallel matlab for multicore and multinode computers. *SIAM, 2009,* 55–140.
51. Wolf, A., Swift, J. B., Swinney, H. L., & Vastano, J. A. (1985). Determining Lyapunov exponents from a time series. *Physical D: Nonlinear Phenomena, 16*(3), 285–317.
52. Wang, X., Lin, T., & Qin, X. (2012). A novel colour image encryption algorithm based on chaos. *Signal Processing, 92,* 1101–1108.
53. Rukhin A., L., Soto J., Nechvatal, JR., et al. (1982). A statistical test suite for random and pseudorandom number generators for cryptographic applications. Nist Special Publication. Special Publication 800–22, Revision1a; 1982 (pp. 1–131).
54. Wang, H., Xiao, D., Chen, X., & Huang, H. (2018). Cryptanalysis and enhancements of image encryption using combination of the 1d chaotic map. *Signal Processing, 144,* 444–452.