Analyse et traitement des images dans le domaine chiffré

Pauline Puteaux

THÈSE POUR OBTENIR LE GRADE DE DOCTEUR DE L'UNIVERSITE DE MONTPELLIER

En Informatique

École doctorale : Information, Structures, Systèmes

Unité de recherche : LIRMM – Laboratoire d'Informatique, de Robotique et de Micro-électronique de Montpellier

Analyse et traitement des images dans le domaine chiffré

Présentée par Pauline PUTEAUX

Le 9 octobre 2020

Sous la direction de William PUECH

Devant le jury composé de

Atilla BASKURT, Professeur des Universités, INSA Lyon / LIRIS	Président
Frédéric DUFAUX, Directeur de Recherche, CNRS, Université Paris-Saclay, CentraleSupélec / L2S	Rapporteur
Caroline FONTAINE, Directeur de Recherche, CNRS, ENS Paris-Saclay / LSV	Rapporteure
François CAYRE, Maître de Conférences, Grenoble-INP / GIPSA-Lab	Examinateur
Fabien LAGUILLAUMIE, Professeur des Universités, Univ. Montpellier / LIRMM	Examinateur
Jean-Marc CHASSERY, Directeur de Recherche, CNRS, GIPSA-Lab	Invité
William PUECH, Professeur des Universités, Univ. Montpellier / LIRMM	Directeur de thèse

UNIVERSITÉ DE MONTPELLIER

Table des matières

Introduction

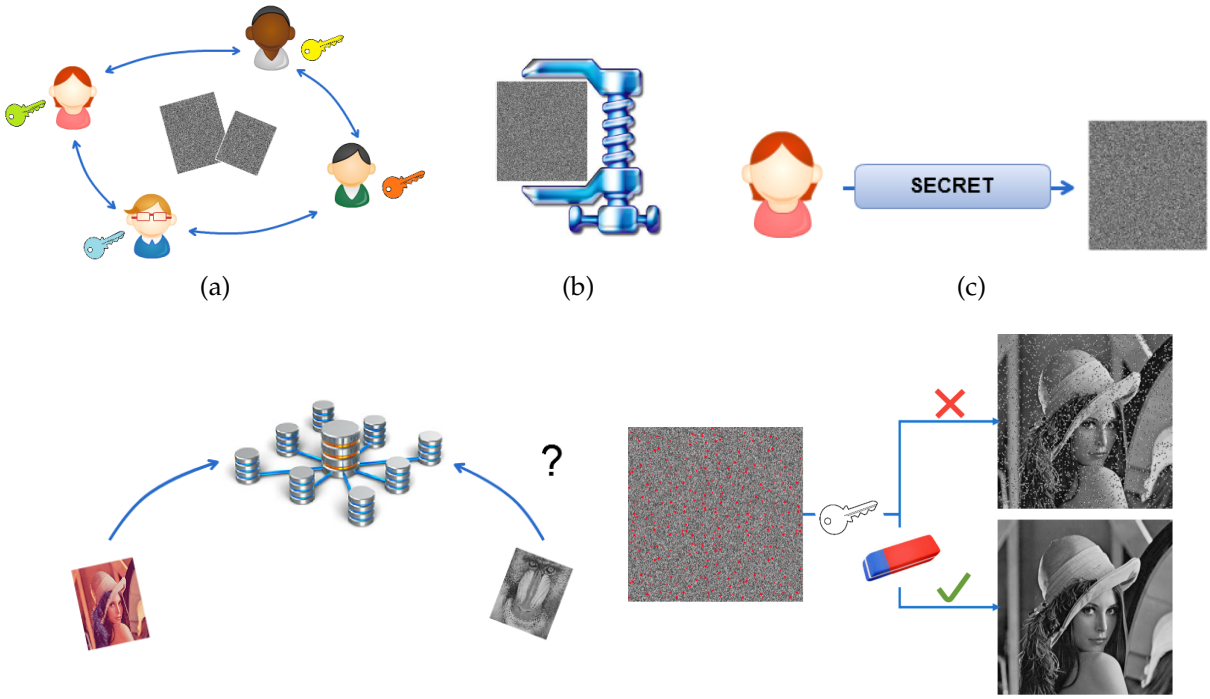
Contexte de la thèse et applications

De nos jours, la sécurité des données visuelles joue un rôle important dans tous les domaines, en particulier lorsqu'un niveau élevé de confidentialité est exigé, comme par exemple pour des applications militaires ou médicales. Avec le développement des services informatiques en nuage (*cloud computing*), de plus en plus de données transitent sur les réseaux. D'après CISCO, les données multimédia représentent plus de 80% du volume de ces données [20]. Cela introduit nécessairement de sérieux problèmes de sécurité où la confidentialité, l'authentification et l'intégrité sont constamment menacées par des activités illégales telles que le piratage, la production de contrefaçons, ou encore l'usage mal attentionné de ces données.

Pour des raisons de protection de la vie privée, les données multimédia sont généralement chiffrées avant d'être transférées ou archivées sur un serveur. En effet, un chiffrement sélectif ou complet peut assurer leur confidentialité visuelle et empêcher une personne non autorisée d'accéder à leur contenu original. Par ailleurs, lors de la transmission ou de l'archivage de ces données chiffrées, il peut être intéressant d'être en mesure de les analyser ou de les traiter de façon sécurisée, c'est-à-dire directement dans le domaine chiffré. Cela constitue un véritable challenge. En effet, dans ce cas, ni la clé de chiffrement utilisée, ni le contenu original des données en clair ne sont connus. De plus, les données multimédia n'ont pas les mêmes propriétés statistiques dans le domaine chiffré que dans le domaine clair.

Comme présenté en fig. 0.1, les principales applications visées par l'analyse et le traitement des données multimédia dans le domaine chiffré sont : le partage de secret visuel entre plusieurs personnes, l'insertion de données cachées dans des données multimédia chiffrées (pour que la personne qui insère ou extrait les données n'ait pas accès au contenu original du support), la recompression d'images ou de vidéos crypto-compressées (pour de la transmission sécurisée sur des réseaux bas débit), l'indexation et la recherche de contenus multimédia dans des bases de données chiffrées, ou encore la correction d'images chiffrées bruitées.

Dans ces travaux de thèse, nous nous intéressons particulièrement à trois applications présentées dans le domaine de l'analyse et du traitement des images dans le domaine chiffré. Nous développons tout d'abord trois méthodes d'insertion de données cachées dans les images chiffrées basées sur la substitution du ou des bit(s) le(s) plus significatif(s) de chaque pixel. Nous décrivons ensuite un nouvel algorithme efficace pour corriger les images chiffrées bruitées. Enfin, nous proposons une méthode de recompression d'images JPEG crypto-compressées directement dans le domaine chiffré.



(a) (b) (c) (d) (e)

FIGURE 0.1 – Applications visées par le traitement des données multimédia dans le domaine chiffré : a) Partage de secret visuel, b) Recompression d'images crypto-compressées, c) Insertion de données cachées dans le domaine chiffré, d) Recherche et indexation dans des bases de données chiffrées, e) Correction d'images chiffrées bruitées.

Plan

Le reste du manuscrit se compose de deux parties, à savoir un état-de-l'art et une présentation détaillée de nos contributions.

Dans la première partie, nous présentons un état-de-l'art de différentes disciplines liées à la sécurité multimédia. Après un brève introduction de la structure d'une image, l'insertion de données cachées (IDC) dans les images est présentée dans le chapitre 1. Dans le chapitre 2, nous détaillons les méthodes et approches existantes de l'état-de-l'art en chiffrement et plus particulièrement, en chiffrement d'images. Enfin, dans le chapitre 3, nous décrivons la problématique d'analyse et de traitement dans le domaine chiffré, puis nous nous intéressons aux méthodes d'IDC dans le domaine chiffré (IDCDC) appliquées aux images numériques.

Dans la deuxième partie de ce manuscrit, nous présentons nos contributions. Dans le chapitre 4, nous proposons une nouvelle méthode d'IDCDC basée sur la prédiction des bits de poids fort (MSB), déclinée en deux approches. Dans le chapitre 5, nous décrivons d'abord une extension de cette méthode, puis nous améliorons les résultats obtenus à l'aide d'une nouvelle approche d'IDCDC récursive permettant d'exploiter pleinement la redondance entre les pixels dans le domaine clair. Nous nous intéressons alors à la problématique des images chiffrées bruitées dans le chapitre 6, où nous proposons une nouvelle méthode efficace pour analyser et corriger ces images. Enfin,

dans le chapitre 7, nous expliquons comment recompresser des images JPEG crypto-compressées de manière sécurisée.

Nous concluons alors ce manuscrit avec un bilan de nos différentes contributions et proposons des perspectives sur l'analyse et le traitement des images dans le domaine chiffré.

Première partie

Etat-de-l'art

CHAPITRE 1

Insertion de données cachées

Sommaire

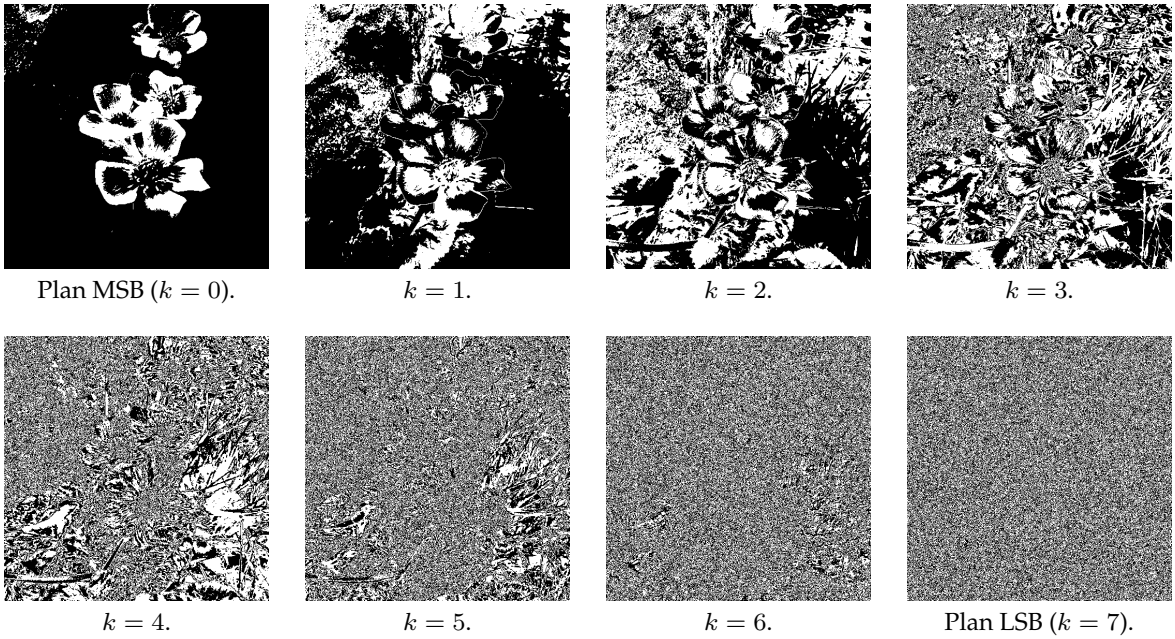
1.1 Introduction

Une image numérique est une matrice en deux dimensions dont les coefficients sont des valeurs discrètes mesurant l'intensité lumineuse et sont appelés pixels. Ces derniers sont généralement notés $p(i, j)$ avec $0 \leq i < m$ et $0 \leq j < n$ si l'image est de taille $m \times n$ pixels. Dans ce manuscrit, nous nous intéressons aux images réelles, c'est-à-dire les images issues d'appareils photo numériques. Nous distinguons alors deux types de formats d'image, à savoir les formats bruts et les formats compressés (décrits dans le chapitre 7). Dans les formats bruts, chaque pixel est représenté indépendamment des autres et est codé sur l bits. Ainsi, les images binaires sont composées de deux couleurs (0 pour noir, 1 pour blanc). Dans les images en niveaux de gris, chaque pixel correspond à une nuance de gris comprise entre 0 et $2^l - 1$ (0 pour noir, $2^l - 1$ pour blanc). En particulier, les images en niveaux de gris avec 8 bits par pixel ($l = 8$, soit 0 pour noir, 255 pour blanc) sont principalement utilisées dans les méthodes décrites dans ce manuscrit. Notons qu'un exemple est illustré en fig. 1.1. En outre, dans les images couleur, l'information peut être décrite dans différents espaces colorimétriques, comme par exemple Rouge, Vert, Bleu (RVB) ou encore YCrCb pour la chrominance et la luminance. Comme trois composantes sont utilisées, le nombre de couleurs possibles pour chaque pixel d'une image couleur est alors $2^{8\times3} = 2^{24}$, soit plus de 16 millions de couleurs.

FIGURE 1.1 – Illustration d'une image en niveaux de gris avec 8 bits par pixel (format PGM) de la base BOWS-2 [3].

Un plan binaire est un ensemble de bits correspondant à la position donnée d'un bit dans chacun des pixels d'une image. Dans la suite de cet ouvrage, nous nous intéressons particulièrement à deux plans binaires, à savoir : le plan MSB (*Most Significant Bit*, plan composé des bits les plus significatifs) et le plan LSB (*Least Significant Bit*, plan composé des bits les moins significatifs). Dans une image avec 256 niveaux de gris, nous pouvons donc dénombrer huit plans binaires.

En fig. 1.2, il est possible d'observer que le plan MSB représente une bonne approximation du contenu de l'image originale qui peut aisément être reconnu. Par ailleurs, les plans binaires suivants sont moins significatifs et permettent de récupérer moins d'information. Par exemple, dans la fig. 1.2, nous pouvons remarquer que les quatre premiers plans binaires ($0 \leq k \leq 3$) renseignent sur le contenu de l'image originale. En revanche, le plan LSB ne permet de distinguer aucun détail et peut être assimilé à



Plan MSB ($k = 0$). $k = 1$. $k = 2$. $k = 3$.

$k = 4$. $k = 5$. $k = 6$. Plan LSB ($k = 7$).

FIGURE 1.2 – Les huit plans binaires de l'image en niveaux de gris illustrée en fig. 1.1, du plus significatif (plan MSB, $k = 0$), au moins significatif (plan LSB, $k = 7$).

du bruit. Notons également que modifier certaines valeurs du plan MSB rend l'image originale dégradée, voire méconnaissable. En revanche, modifier le plan LSB n'entraîne qu'une légère altération de l'image originale non visible pour le système visuel humain (SVH).

Dans ce chapitre, nous allons détailler le concept d'insertion de données cachées (IDC). L'IDC permet d'insérer des données cachées au sein d'un support numérique en modifiant son contenu sans être perceptible visuellement et/ou statistiquement. Les données cachées insérées peuvent être des métadonnées, un identifiant, une empreinte numérique, ou encore un message secret sans lien avec le contenu du support. Lors de l'insertion des données cachées, le format et la taille du support original ne doivent pas être modifiés. Ainsi, le média marqué doit pouvoir être visualisé et manipulé avec les mêmes éditeurs que le support original. De plus, la préservation de la taille du média permet de limiter l'espace de stockage.

En section 1.2, nous commençons par décrire les principes et propriétés des méthodes d'insertion de données cachées. Ensuite, la section 1.3 présente le concept de communication secrète. La section 1.4 développe la notion de tatouage. Les principales méthodes d'insertion de données cachées réversibles sont alors décrites en section 1.5. Enfin, en section 1.6, nous concluons ce chapitre.

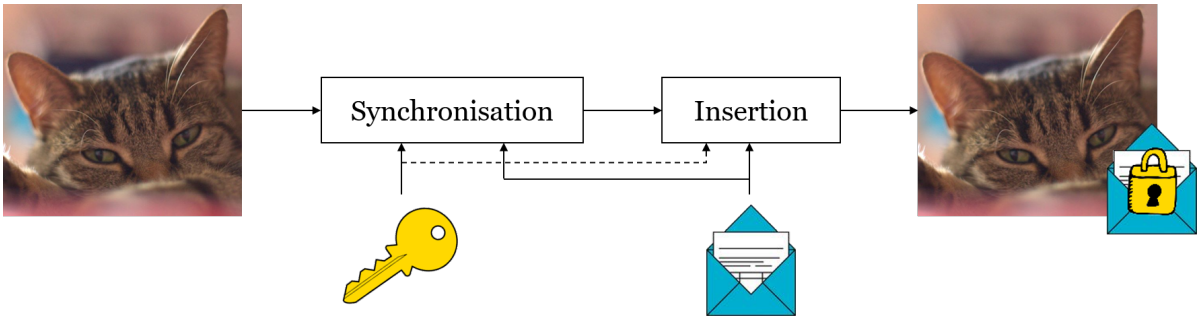
1.2 Principes et propriétés

Les méthodes d'insertion de données cachées (IDC) consistent à dissimuler des données cachées au sein d'un support numérique. En section 1.2.1, nous présentons les deux phases distinctes de ces méthodes, à savoir l'insertion et l'extraction des données cachées. Nous décrivons ensuite les différentes classes pouvant être distinguées et leurs

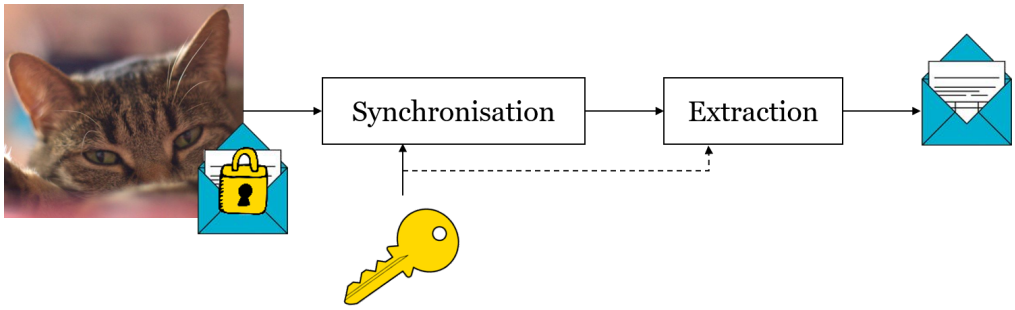
propriétés en section 1.2.2. Enfin, les métriques utilisées pour l'évaluation des méthodes sont explicitées en section 1.2.3.

1.2.1 Insertion et extraction

Dans leur livre, Cox *et al.* définissent le contexte des méthodes d'IDC [22]. Ainsi, elles doivent comporter deux phases distinctes, à savoir une phase d'insertion et une phase d'extraction du message. De plus, une étape de synchronisation est nécessaire dans chacune de ces deux phases.



(a) Phase d'insertion.



(b) Phase d'extraction.

FIGURE 1.3 – Phases d'insertion et d'extraction du message dans les méthodes d'IDC.

La fig. 1.3.a présente la phase d'insertion d'une méthode d'IDC. Une image est utilisée comme support des données cachées. Lors de l'étape de synchronisation, l'ordre des pixels à parcourir et/ou les zones de l'image sélectionnées pour l'insertion sont définis à l'aide d'une clé secrète de façon pseudo-aléatoire ou suivant le contenu de l'image. Notons que l'utilisation d'une telle clé permet de sécuriser cette étape. Par ailleurs, avant de réaliser l'insertion des données cachées, celles-ci sont généralement chiffrées. Différentes stratégies peuvent être adoptées pour l'insertion des données cachées. Dans la plupart des cas, l'insertion est réalisée par substitution. Dans ce type d'algorithmes, la redondance d'information dans le support est exploitée. En effet, certains bits du support sont remplacés par ceux des données cachées. En outre, d'autres méthodes sont basées sur une insertion par injection. Les données cachées sont directement insérées dans le média, en en augmentant la taille et sans exploiter les propriétés statistiques de ses éléments. Généralement, les méthodes par injection sont peu sécurisées face à un attaquant. Enfin, après l'insertion du message, l'image marquée obtenue est visuellement et statistiquement très proche de l'image support.

La fig. 1.3.b illustre la phase d'extraction des données cachées. En utilisant la clé secrète, les données cachées sont généralement reconstruites en deux étapes. Tout d'abord, lors de l'étape de synchronisation, les zones d'insertion et l'ordre de lecture sont retrouvés à l'aide de la clé secrète. Les données cachées peuvent alors être extraites dans le bon ordre (et éventuellement déchiffrées). Par ailleurs, les méthodes d'IDC reposent sur le principe de Kerckhoffs [67]. L'algorithme d'insertion doit être publié et connu de tous : la sécurité doit uniquement reposer sur le secret de la clé. Ainsi, l'extraction des données cachées dépend seulement de la connaissance de la clé secrète. Notons toutefois que certaines méthodes utilisent un vôte majoritaire pour reconstruire le message est ne nécessitent pas d'étape de synchronisation au décodage [157]. Elles ont néanmoins été montrées peu sécurisées [49].

1.2.2 Différentes classes et propriétés

Les différentes classes d'IDC sont présentées en fig. 1.4. Nous distinguons trois grandes classes d'application : la stéganographie, le tatouage et l'IDC réversible. Ces classes générales sont décrites en détail dans les sections 1.3, 1.4 et 1.5 de ce chapitre.

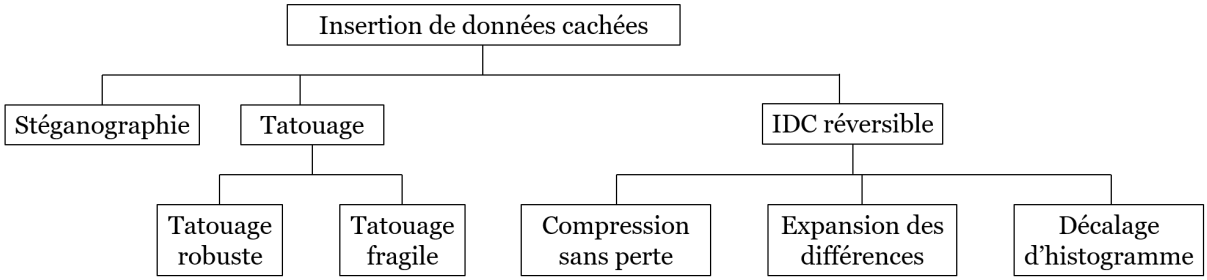


FIGURE 1.4 – Classification des méthodes d'IDC.

Suivant leur classe d'application, les méthodes d'IDC doivent respecter différentes propriétés, à savoir :

— L'**imperceptibilité** : l'invisibilité visuelle et statistique des données cachées dans le support,

— La **capacité** : la quantité de bits pouvant être insérés dans le support,

— La **robustesse** : la résistance des données cachées à une modification du support standard (opération géométrique, compression, filtrage...) ou malicieuse (suppression, désynchronisation du message...),

— La **sécurité des données** : le niveau de sécurité des données cachées insérées,

— La **sécurité d'insertion** : la garantie qu'un attaquant ne puisse pas retrouver la clé d'insertion,

— La **complexité** : la complexité algorithmique de la méthode d'insertion (et de la méthode d'extraction), ainsi que leur temps d'exécution,

— La **réversibilité** : la possibilité de retrouver intégralement le support original après l'extraction des données cachées.

Bien que la robustesse et la sécurité sont des concepts voisins, ils conviennent d'être distingués. Ces deux concepts ne peuvent pas être simplement différenciés selon le caractère « intentionnel » d'une attaque. Par exemple, si une image marquée est compressée (attaque sur la robustesse), la marque peut ainsi avoir été effacée intentionnellement ou non [9]. Notons qu'il convient également de distinguer la sécurité de l'insertion de la sécurité des données.

Par ailleurs, il existe un réel compromis entre ces différentes propriétés. Ainsi, nous pouvons constater qu'améliorer l'une d'entre elles fait systématiquement décroître les autres. Dans le cas de la stéganographie, une attention particulière est portée à l'imperceptibilité des données cachées. En effet, l'insertion de ces dernières ne doit pas provoquer d'artefact dans le support original pour ne pas compromettre la communication secrète. En outre, une méthode de tatouage robuste doit l'être le plus possible car les données cachées insérées doivent pouvoir résister à diverses manipulations sur le support (en particulier dans une application de vérification de la propriété) [9]. Enfin, la plupart des méthodes d'IDC réversibles permettent d'insérer une grande quantité de données cachées : une haute capacité est généralement recherchée.

1.2.3 Métriques

Une méthode d'IDC est évaluée suivant la quantité de bits insérés, le taux de bits erronés lors de l'extraction des données cachées et la qualité visuelle de l'image marquée et/ou de l'image reconstruite après extraction des données cachées. Notons qu'il existe un réel compromis entre ces différentes métriques. En effet, plus le nombre de bits insérés est grand, plus la qualité visuelle risque d'être dégradée et plus le taux de bits extraits erronés risque d'être important.

Quantité de bits insérés

La quantité de bits insérés est exprimée en bits-par-pixel (bpp). Pour une image dont les pixels sont codés sur 256 niveaux de gris, cette quantité est donc comprise théoriquement entre 0 bpp et 8 bpp. Par ailleurs, il existe une distinction entre la capacité d'insertion et la charge utile. En effet, la capacité d'insertion fait référence au nombre total de bits pouvant être insérés dans une image en appliquant une méthode d'IDC. La charge utile, quant à elle, désigne le nombre total de bits de données cachées pouvant être insérés dans l'image. Ainsi, la charge utile est nécessairement inférieure ou égale à la capacité d'insertion. Généralement, les méthodes d'IDC ont une charge utile comprise entre $0,01$ bpp et $0,5$ bpp. Cependant, certaines approches décrites dans les chapitres 3 – 5 sont dites à « haute capacité », lorsque la charge utile est supérieure à 1 bpp.

Taux de bits extraits erronés

Le taux de bits erronés lors de l'extraction des données cachées depuis l'image marquée est calculé en divisant le nombre de bits mal reconstruits par le nombre total de bits de données cachées. Ainsi, celui-ci doit être le plus faible possible pour assurer une bonne transmission des données cachées insérées.

Qualité visuelle

La qualité visuelle de l'image marquée et/ou de l'image reconstruite après extraction des données cachées peut être évaluée à l'aide du rapport signal-bruit (PSNR) ou du score de similarité structurelle (SSIM). Ces deux métriques sont dites « avec référence » puisque l'évaluation de la qualité visuelle s'effectue en comparaison avec l'image originale.

Rapport signal-bruit (PSNR, Peak-Signal-to-Noise Ratio) : Le PSNR est utilisé pour mesurer la similarité entre deux images :

$$\text{PSNR} = 10 \cdot \log_{10} \frac{(2^l - 1)^2}{\frac{1}{m \times n} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} (p(i,j) - p'(i,j))^2}, \tag{1.1}$$

où $p(i,j)$ est un pixel de l'image originale et $p'(i,j)$ le pixel associé dans l'image reconstruite, toutes deux de même taille et dont les pixels sont codés sur 2^l niveaux de gris. Le PSNR se mesure en décibels (dB). Pour des images en niveaux de gris sur 8 *bpp* ($l = 8$), si deux images ont un contenu totalement différent, il est d'environ $10\ dB$. Entre deux images semblables mais altérées par un bruit, il avoisine $15\ dB$. Par ailleurs, lorsque les deux images sont fortement similaires, il est supérieur à $30\ dB$. Enfin, dans le cas d'une reconstruction parfaite, c'est-à-dire lorsque les deux images sont strictement identiques, sa valeur tend vers $+\infty$. Le défaut majeur du PSNR est de ne pas prendre en compte la qualité visuelle de l'image reconstruite. Ainsi, il ne peut pas être considéré comme une mesure totalement objective.

Similarité structurelle (SSIM, Structural Similarity) [158] : Le SSIM permet d'évaluer la similarité de structure entre deux images, plutôt qu'une différence pixel à pixel comme le fait le PSNR. L'hypothèse sous-jacente est que le SVH est plus sensible aux changements dans la structure de l'image :

$$\text{SSIM}(x,y) = \frac{(2E(x)E(y) + \gamma_1)\,(2Cov(x,y) + \gamma_2)}{(E(x)^2 + E(y)^2 + \gamma_1)\,(V(x)^2 + V(y)^2 + \gamma_2)}, \tag{1.2}$$

avec x et y des fenêtres des deux images, $E(x)$ est la moyenne de l'ensemble x, $V(x)$ sa variance, $Cov(x,y)$ la covariance entre les ensembles x et y, $\gamma_1 = (0,01 \times (2^l - 1))^2$ et $\gamma_2 = (0,03 \times (2^l - 1))^2$. Cette formule est appliquée à différentes fenêtres des deux images à comparer. La moyenne entre les différentes valeurs obtenues est alors calculée pour obtenir la valeur globale du SSIM. Celle-ci est comprise entre 0 et 1 ; elle est égale à 1 lorsque les deux images sont parfaitement identiques. Notons qu'il est souvent nécessaire d'observer au moins trois décimales pour que la valeur du SSIM soit significative.

1.3 Communication secrète

La stéganographie – étymologiquement *écriture secrète* – est l'art de cacher un message secret dans un support de couverture sans que celui-ci ne paraisse suspect. Contrairement à certaines méthodes d'IDC, le support de couverture (généralement une image)

n'a aucune valeur en lui-même : il ne sert que d'hôte au message secret. Ainsi, la stéganographie vise à communiquer secrètement. Dans cette section, nous décrivons les principes de la stéganographie (section 1.3.1) et ceux de la stéganalyse, sa discipline duale (section 1.3.2).

1.3.1 Stéganographie

La stéganographie moderne consiste à utiliser un support numérique pour transmettre un message secret de façon indétectable. Son principe général peut être illustré par le problème des prisonniers décrit par Simmons [139] et présenté en fig. 1.5. Supposons qu'Alice et Bob, deux prisonniers enfermés dans deux cellules différentes, souhaitent partager un message d'évasion. Tous leurs échanges sont contrôlés par la gardienne Eve, qui peut à tout moment interrompre leur communication en cas de suspicion. Notons qu'utiliser une des méthodes classiques de cryptographie pour chiffrer le message envoyé n'est pas envisageable : cela éveillerait les soupçons d'Eve. De ce fait, Alice et Bob choisissent de dissimuler leur message dans un support anodin. Ainsi, leurs messages sont autorisés à circuler et leur communication est assurée. Dans ce scénario, Alice et Bob sont les stéganographes et Eve est la stéganalyste. Cette dernière est libre d'examiner les messages circulant entre les deux prisonniers de manière passive, active ou malicieuse. Les méthodes de stéganographie doivent donc être imperceptibles visuellement et statistiquement, mais ne nécessitent pas d'être robustes. Les approches par substitution ou par injection décrites dans la section 1.2 peuvent être utilisées pour réaliser l'insertion d'un message secret. Par ailleurs, l'insertion peut aussi s'effectuer par sélection ou par génération. Dans les approches par sélection, le message est inséré dans un support spécialement sélectionné en tant qu'hôte. Bien que coûteuses en temps de recherche, ces méthodes offrent une bonne résistance à la stéganalyse. En outre, les approches par génération consistent à générer le support hôte en fonction du message à insérer. Elles sont plus difficiles à mettre en place de par la difficulté à générer un support réaliste.

1.3.2 Stéganalyse

La stéganalyse est la discipline duale de la stéganographie. Son principe général est de détecter si un support numérique contient des informations cachées. Ainsi, cela permet au stéganalyste de bloquer sa transmission si tel est le cas. Cependant, son but peut être étendu suivant le scénario considéré et les connaissances *a priori* dont dispose un attaquant. De nombreux travaux ont été développés et plusieurs grands types de stéganalyse peuvent être mis en évidence. Tout d'abord, les méthodes de détection doivent tenir compte du type de média de couverture utilisé lors de l'insertion du message. Dans le cas des images, des méthodes différentes ont été développées pour les formats non-compressés et compressés [27]. Par ailleurs, certaines approches visent à rechercher la valeur de la clé d'insertion, similairement à ce qui fait en cryptanalyse. D'autre part, la stéganalyse quantitative a pour but d'estimer la taille du message secret inséré [94]. Dans ce cas, si la charge utile estimée est grande, le support est alors considéré comme marqué. En outre, la stéganalyse active consiste à modifier légèrement le média transmis – tout en minimisant la distorsion introduite – afin de

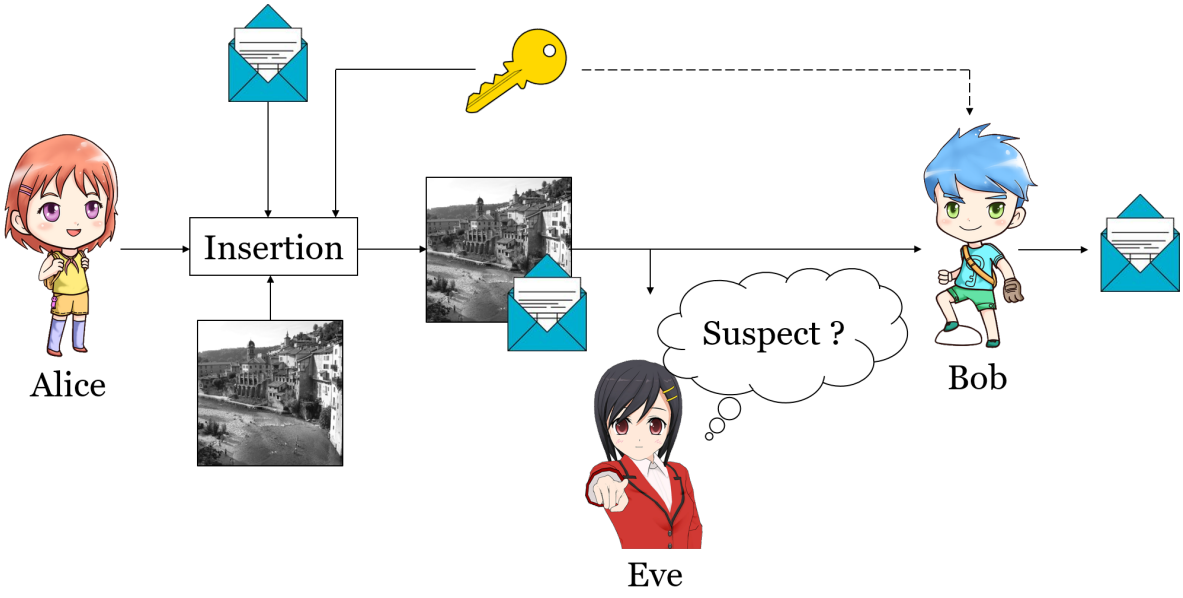


FIGURE 1.5 – Illustration du problème des prisonniers [139].

rendre impossible l'extraction du message caché [12]. Enfin, notons que la stéganalyse a été étudiée comme outil d'évaluation des méthodes de stéganographie. Dans ce contexte, la personne qui procède à l'analyse du support pré-supposé marqué est omnisciente : elle connaît la méthode de stéganographie pouvant être utilisée et ses paramètres (type de média, algorithme d'insertion, taille du message, propriétés statistiques...). Elle ignore seulement si un message secret a été inséré, son éventuel contenu et la valeur de la clé d'insertion. Ce scénario est plutôt pessimiste pour l'expéditeur du message secret. En effet, une stéganalyse aussi précise peut difficilement être mise en place en pratique. Ainsi, la résistance d'une méthode de stéganographie à une telle stéganalyse assure qu'elle est suffisamment sécurisée.

1.4 Tatouage

Comme en stéganographie, les méthodes de tatouage doivent permettre de rendre indiscernables des données cachées [9]. Cependant, la stéganographie recherche une indiscernabilité statistique (robustesse aux tests statistiques), tandis que le tatouage recherche une indiscernabilité perceptuelle (non distinguable par le SVH). Deux grandes catégories de méthodes peuvent être distinguées : le tatouage fragile (section 1.4.1) et le tatouage robuste (section 1.4.2). Le tatouage est très souvent utilisé comme une signature, pour lutter contre les contrefaçons par exemple.

1.4.1 Tatouage fragile

Le tatouage fragile est utilisé pour vérifier l'authenticité ou l'intégrité d'un support numérique. Lorsqu'il est utilisé pour une vérification d'authenticité, si le tatouage est absent ou dégradé dans un média supposé marqué, alors son authenticité n'est pas vérifiée. Dans une problématique de contrôle d'intégrité, le tatouage est réalisé pour

connaître les traitements effectués sur l'image marquée. Les modifications opérées sur le média altèrent les données cachées qui peuvent alors être analysées. Cela peut permettre de détecter une zone de pixels falsifiée [175] et la restaurer [121]. Enfin, le tatouage est dit « semi-fragile » lorsqu'il est conçu pour résister à des modifications classiques en traitement d'images, telles que la compression au format JPEG par exemple.

1.4.2 Tatouage robuste

Le tatouage pour l'ayant-droit permet de marquer un média par un identifiant unique associé à son propriétaire [144]. Les données cachées sont insérées dans le média et une copie est conservée par un tiers. Ce type de tatouage robuste sert alors à la vérification de la propriété d'un média. Il doit donc être robuste aux attaques tant que le support n'est pas trop dégradé pour avoir une quelconque valeur marchande. Par ailleurs, l'insertion d'une empreinte numérique – appelée *fingerprinting* – consiste à marquer une image par un identifiant unique, associé à un utilisateur. Ces données cachées peuvent être insérées pour effectuer du traçage de traitres en cas de piratage ou de divulgation d'information [42]. Dans ce cas d'application, les données cachées doivent être robustes pour permettre la mise en cause des utilisateurs pirates. Les approches de tatouage robuste sont généralement basées sur le principe d'étalement de spectre (*spread spectrum*) [23] ou l'utilisation de fonctions de quantification [13]. Popularisées par Cox *et. al*, les méthodes de tatouage basées sur l'étalement de spectre consistent à réaliser l'insertion de données cachées à partir d'une combinaison linéaire entre le support et un signal de bruit modulé par le signal des données cachées à insérer. D'autre part, en théorie de l'information, l'IDC s'apparente à une communication avec information adjacente et peut être théoriquement décrite par des codes « *Dirty Paper* ». Une implémentation pratique a alors été proposée par Chen et Wornell dans leur méthode QIM (*Quantization Index Modulation*) [13].

1.5 Insertion de données cachées réversible

L'insertion de données cachées réversible (IDCR) est utilisée pour l'authentification et l'ajout d'informations additionnelles dans les images, dans des applications sensibles (milieu militaire, imagerie médicale...) où la distorsion est strictement interdite ou fortement déconseillée. En effet, la spécificité des méthodes d'IDCR est de permettre une reconstruction parfaite de l'image originale après l'extraction des données cachées insérées. Par ailleurs, les approches peuvent être classées en trois catégories principales suivant qu'elles sont basées sur la compression sans perte [40, 10] (section 1.5.1), l'expansion des différences entre les pixels [148, 149] (section 1.5.2), ou encore sur le décalage d'histogramme [87, 143, 43] (section 1.5.3). Dans cette section, nous décrivons exclusivement les méthodes réalisées dans le domaine spatial. Notons qu'un état-de-l'art plus exhaustif des méthodes d'IDCR a été réalisé en 2016 par Shi *et al.* [137].

1.5.1 Compression sans perte

Les premières méthodes significatives d'IDCR sont basées sur la compression sans perte. Dans ces approches, un sous-ensemble de pixels S de l'image originale est d'abord

compressé sans perte pour libérer de l'espace. La phase d'insertion est alors réalisée en remplaçant S par sa forme compressée et les bits des données cachées. Ainsi, nous notons que la performance de ces méthodes en termes de charge utile est directement liée à l'algorithme de compression sans perte utilisé et les caractéristiques du sous-ensemble S. Fridrich *et al.* ont proposé de compresser le plan LSB d'une image originale pour réaliser l'insertion d'une empreinte numérique de 128 bits [40]. Goljan *et al.* ont décrit une approche appelée « schéma-RS » (*RS-scheme*) [45]. L'image originale est divisée en blocs de pixels, qui sont ensuite classés en trois catégories suivant leur homogénéité : blocs réguliers (R-blocs), blocs singuliers (S-blocs) et blocs inexploitables. L'emplacement des R-blocs et des S-blocs est alors renseigné dans un vecteur binaire appelé vecteur-RS. Ce dernier est modifié selon les bits des données cachées à insérer, puis compressé et transmis avec l'image marquée. Par ailleurs, deux méthodes haute capacité ont également été proposées. Afin d'améliorer l'efficacité de l'algorithme de compression sans perte utilisé dans la méthode [40], Celik *et al.* ont décrit une approche de compression généralisée des bits les moins significatifs [10]. Au lieu de modifier seulement le plan LSB, les pixels sont modifiés par quantification pour insérer des données cachées. En outre, Xuang *et al.* ont proposé de compresser certains bits des coefficients IWT (*Integer Wavelet Transform*, transformée en ondelettes entière) des sous-bandes de haute fréquence pour procéder à l'insertion des données cachées.

1.5.2 Expansion des différences

La première méthode par expansion des différences a été introduite par Tian en 2002 [148], puis améliorée en 2003 [149]. La méthode consiste à calculer les différences entre un pixel et les valeurs de son voisinage et à sélectionner un certain nombre de valeurs pour définir l'expansion de la différence. Les bits des données cachées sont ensuite dissimulés dans les valeurs de différences. La première étape consiste à appliquer la transformation en ondelettes de Haar. Soit un couple $(p_0,\ p_1)$ de deux pixels de l'image originale codés sur 256 valeurs de niveaux de gris. La moyenne entière l entre ces deux pixels, ainsi que leur différence h sont d'abord calculés :

$$l = \left\lfloor \frac{p_0 + p_1}{2} \right\rfloor,\ h = p_1 - p_0. \tag{1.3}$$

Pour insérer un bit $b \in \{0, 1\}$ des données cachées, la différence h est étendue :

$$h^* = 2h + b. \tag{1.4}$$

La moyenne entière l, quant à elle, n'est pas modifiée. La paire de pixels $(p_{m0},\ p_{m1})$ de l'image marquée est alors calculée d'après les valeurs l et h^* :

$$p_{m0} = l - \left\lfloor \frac{h^*}{2} \right\rfloor = 2p_0 - \left\lceil \frac{p_0 + p_1}{2} \right\rceil, \tag{1.5}$$

$$p_{m1} = l + \left\lfloor \frac{h^* + 1}{2} \right\rfloor = 2p_1 - \left\lceil \frac{p_0 + p_1}{2} \right\rceil + b. \tag{1.6}$$

Dans l'image marquée, le LSB de la différence $p_{m1} - p_{m0}$ correspond au bit inséré b des données cachées et la paire $(p_0,\ p_1)$ des deux pixels de l'image originale est reconstruite sans erreur :

$$p_0 = l' - \left\lfloor \frac{h'}{2} \right\rfloor, \; p_1 = l' + \left\lceil \frac{h'}{2} \right\rceil, \tag{1.7}$$

$$\text{avec } l' = \left\lfloor \frac{p_{m0} + p_{m1}}{2} \right\rfloor, \; h' = \left\lfloor \frac{p_{m1} - p_{m0}}{2} \right\rfloor. \tag{1.8}$$

Ainsi, comme un bit des données cachées peut être inséré par paire de pixels, la charge utile est haute et égale à $0,5$ *bpp*. Notons que, dans cette méthode, une carte de localisation est utilisée pour renseigner l'emplacement des couples de pixels utilisés pour l'expansion. Bien qu'augmentant la taille des données à transmettre, l'utilisation d'une telle carte est courante dans les méthodes d'IDCR, en particulier pour signaler d'éventuels problèmes de débordement. Le concept d'expansion des différences a largement été étudié et développé, en particulier dans les approches d'IDCR basées sur la transformation d'entier en entier [1, 21], sur l'expansion des erreurs de prédiction (où un prédicteur est utilisé à la place de l'opérateur de différence) [146, 147] ou encore sur l'insertion adaptative [63, 126, 75].

1.5.3 Décalage d'histogramme

Un grand nombre de méthodes d'IDCR par décalage d'histogramme ont été décrites dans l'état-de-l'art. Dans ce type d'approches, un histogramme est d'abord généré et l'insertion des données cachées est réalisée en le modifiant significativement. La première méthode dans ce domaine a été proposée par Ni *et al.* [87]. Tout d'abord, un entier a est sélectionné entre les différentes valeurs possibles des pixels de l'image. Par convention, a prend la valeur du niveau de gris le plus représenté, *i.e.* ayant la classe la plus haute dans l'histogramme, pour maximiser la charge utile. L'insertion d'un bit b des données cachées est alors réalisée d'après la valeur de a. A chaque pixel p_i de l'image originale, le pixel p_{m_i} de l'image marquée est obtenu tel que :

$$p_{m_i} = \begin{cases} p_i - 1, & \text{si } p_i < a, \\ p_i - b, & \text{si } p_i = a, \\ p_i & \text{sinon.} \end{cases} \tag{1.9}$$

La fig. 1.6 illustre cette modification au niveau de l'histogramme de l'image. La classe la plus haute, en rouge, est associée à la valeur a. Les classes à gauche de cette classe sont décalées vers la gauche, ce qui revient à soustraire 1 à toutes les valeurs des pixels associées. Ce décalage résulte en la libération de la classe précédant celle en rouge. L'insertion des données cachées dans les pixels égaux à a entraîne une « division » de la classe en rouge : si le bit des données cachées à insérer est égal à 1, il y a un décalage vers la gauche et, s'il est égal à 0, il n'y a pas de décalage. Enfin, les classes à droite de la classe en rouge ne sont pas modifiées.

Ainsi, le principe des méthodes par décalage d'histogramme est simple : créer de l'espace vacant en déplaçant certaines classes et en étendre d'autres en exploitant l'espace libéré pour insérer les données cachées. De plus, comme chaque pixel est au plus modifié de 1, la qualité de l'image marquée reste très élevée.

Dans des méthodes plus récentes de l'état-de-l'art, l'histogramme des erreurs de prédiction est modifié à la place de celui des valeurs des pixels [150, 143, 126]. De cette

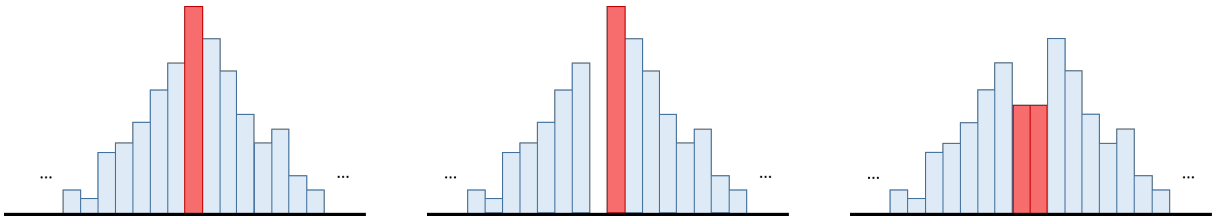


FIGURE 1.6 – Illustration de la méthode d'IDCR par décalage d'histogramme décrite par Ni *et al.* [87] : a) Histogramme de l'image originale ; la classe la plus haute est représentée en rouge, b) Décalage d'un cran vers la gauche des classes à gauche de celle en rouge, c) Histogramme de l'image marquée, en supposant que les valeurs des bits des données cachées sont uniformément distribuées.

façon, le compromis entre la charge utile et la qualité de l'image marquée est plus intéressant. Le principe général des méthodes basées sur le décalage de l'histogramme des erreurs de prédiction peut être formulé comme suit :

1. Prédiction et génération de l'histogramme : Un sous-ensemble de pixels p_i dans l'image originale est tout d'abord sélectionné. D'après leur voisinage, un prédicteur $\hat{p}_i$ est calculé pour chacun de ces pixels. L'erreur de prédiction associée est alors évaluée $e_i = p_i - \hat{p}_i$. Enfin, un histogramme est généré à partir de ces erreurs. Nous remarquons généralement que la distribution des erreurs de prédiction peut être approchée par une distribution laplacienne centrée en zéro.
2. Décalage de l'histogramme : Pour insérer un bit b des données cachées, chaque valeur e_i est modifiée en e_{mi} par expansion ou décalage :

$$e_{mi} = \begin{cases} 2e_i + b, & \text{si } e_i \in [-T,\ T[\,, \\ e_i + T, & \text{si } e_i \in [T,\ +\infty[\,, \\ e_i - T, & \text{sinon,} \end{cases} \tag{1.10}$$

où T est un paramètre entier dont la valeur dépend de la charge utile souhaitée. Ainsi, les classes des valeurs dans l'intervalle $[-T,\ T[$ sont étendues pour insérer les bits des données cachées, et celles dans l'intervalle $]-\infty,\ -T[\cup [T,\ +\infty[$ sont décalées pour créer de l'espace vacant. Enfin, chaque pixel p_i de l'image originale est modifié en $p_{mi} = \hat{p}_i + e_{mi}$ pour obtenir l'image marquée.

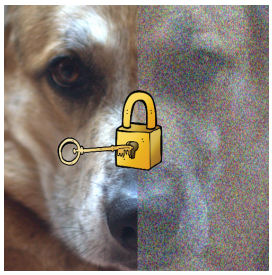
1.6 Conclusion

Ce chapitre a été consacré à la description des caractéristiques, des propriétés et des applications des méthodes d'insertion de données cachées (IDC). Ainsi, nous avons passé en revue les différentes classes d'application de ces méthodes, à savoir : la stéganographie, la tatouage et l'IDC réversible. Par ailleurs, suivant la classe, nous avons vu qu'il existait un réel compromis entre l'imperceptibilité, la capacité, la robustesse, la sécurité et la complexité.

Dans nos travaux de recherche, nous portons un intérêt particulier aux méthodes d'IDC réversibles, afin de concilier une haute capacité de dissimulation et une reconstruction parfaite de l'image originale après l'extraction des données cachées.

Par ailleurs, pour des raisons de confidentialité, il est parfois nécessaire de vouloir sécuriser le contenu de l'image elle-même. Le chapitre 2 présente les différents types possibles de chiffrement d'images.

CHAPITRE 2



Chiffrement

Sommaire

2.1 Introduction

La cryptographie est une des disciplines de la cryptologie – étymologiquement *science du secret* – s'attachant à protéger des messages, en faisant intervenir une ou plusieurs clés. Contrairement à la stéganographie et à l'insertion de données cachées, qui consistent à dissimuler un message dans un autre contenu comme présenté dans le chapitre 1, le chiffrement rend un message inintelligible à autre que qui-de-droit.

Le chiffrement d'images est une méthode efficace pour assurer la confidentialité visuelle de leur contenu, tout en préservant leur format et leur taille. En introduisant de l'aléa, il empêche une personne non autorisée d'accéder au contenu original d'une image en clair. Selon l'application et le niveau de sécurité désiré, le chiffrement peut être total lorsque l'intégralité des données originales est chiffrée ou sélectif lorsque seulement une partie des données est sélectionnée pour être protégée.

Dans ce chapitre, en section 2.2, nous commençons par introduire les fondements de la cryptographie. Nous décrivons ensuite les deux grandes classes d'algorithmes de chiffrement, à savoir : le chiffrement symétrique et le chiffrement asymétrique. Ensuite, la section 2.3 présente les méthodes de chiffrement d'images et les métriques utilisées pour évaluer leur sécurité visuelle. Enfin, en section 2.4, nous concluons ce chapitre.

2.2 Chiffrement

Dans cette section, nous présentons tout d'abord les fondements de la cryptographie moderne (section 2.2.1). Nous décrivons alors les deux classes d'algorithmes de chiffrement, à savoir le chiffrement symétrique et le chiffrement asymétrique. Ainsi, la section 2.2.2 présente les algorithmes symétriques et la section 2.2.3 décrit les modes de chiffrement pouvant être utilisés en chiffrement symétrique. Par ailleurs, le chiffrement asymétrique est développé en section 2.2.4 et la propriété d'homomorphisme possédée par certains algorithmes de cette classe est détaillée en section 2.2.5. Enfin, les algorithmes de chiffrement symétrique et asymétrique sont comparés et une discussion est proposée en section 2.2.6.

2.2.1 Fondements en cryptographie

Bien qu'utilisée depuis l'Antiquité, la cryptographie a connu un réel essor à la fin du XX^{ème} siècle, notamment grâce au développement de l'informatique. Les principes de la cryptographie moderne sont établis par Kerckhoffs dans l'article « La cryptographie militaire » du *Journal des sciences militaires* paru en 1883 [67] :

— La sécurité repose sur le secret de la clé et non sur le secret de la méthode,

— Le déchiffrement du message chiffré sans la clé doit être matériellement, sinon mathématiquement difficile,

— Trouver la clé à partir du texte clair et du texte chiffré doit être difficile.

Ainsi, cela suppose que tous les paramètres du cryptosystème autres que la clé doivent être publics et connus de tous. En 1949, Shannon énonce à son tour la maxime « l'adversaire connaît le système » [134]. Les principes de Kerckhoffs et la maxime de

Shannon sont aujourd'hui considérés comme fondamentaux dans la mise en place de tout cryptosystème. Ils s'opposent à la sécurité par l'obscurité, qui repose sur la non-divulgation d'informations relatives à la structure, au fonctionnement et à l'implémentation d'un algorithme à protéger (par exemple, par offuscation de code, exécution de code distant...).

La cryptanalyse est la technique qui consiste à chercher à déduire le message en clair à partir du message chiffré, sans posséder la clé de chiffrement. Différents scénarios d'attaque peuvent alors être envisagés :

— **Attaque sur le chiffré seul (***Ciphertext-Only***)** : l'attaquant possède plusieurs messages chiffrés avec la même clé et peut faire des hypothèses sur les messages en clair associés (non connus) ; c'est le scénario le plus complexe pour l'attaquant, qui a peu d'informations à sa disposition,

— **Attaque à clair connu (***Known-Plaintext***)** : l'attaquant possède des messages ou des parties des messages en clair ainsi que leur version chiffrée ; la cryptanalyse linéaire fait partie de cette catégorie,

— **Attaque à clair choisi (***Chosen-Plaintext***)** : l'attaquant possède des messages en clair et peut créer les messages chiffrés associés car il a l'algorithme de chiffrement à sa disposition (considéré comme une boîte noire) ; la cryptanalyse différentielle fait partie de cette catégorie,

— **Attaque à chiffré choisi (***Chosen-Ciphertext***)** : l'attaquant possède des messages chiffrés et un oracle peut lui procurer leur version en clair.

Notons également que, si la clé de chiffrement est de petite taille, une attaque simple – appelée attaque par force brute – consiste à effectuer une recherche exhaustive de sa valeur.

Les algorithmes de chiffrement doivent ainsi être développés en considérant ces différents scénarios d'attaque. Deux grandes classes d'algorithmes peuvent être définies : le chiffrement symétrique – appelé aussi chiffrement à clé secrète – et le chiffrement asymétrique – également dit chiffrement à clé publique.

2.2.2 Chiffrement symétrique

En chiffrement symétrique, la même clé, appelée clé secrète, est utilisée lors du chiffrement et du déchiffrement d'un message. Ainsi, la sécurité du cryptosystème repose sur l'échange sécurisé de la clé ; seuls l'expéditeur et le destinataire du message doivent connaître la clé secrète. Par ailleurs, cette clé doit être de taille suffisamment grande pour se prémunir des attaques par force brute. Il existe deux catégories de cryptosystèmes symétriques, à savoir : les algorithmes de chiffrement par flot et les algorithmes de chiffrement par blocs. Les algorithmes de chiffrement par flot considèrent le message en clair comme un flux de données (bits ou octets). Lors du chiffrement, chaque caractère est chiffré de manière indépendante, ce qui rend les opérations très simples et l'exécution de l'algorithme très rapide. En effet, le chiffrement par flot peut s'opérer en temps réel, sans attendre la réception complète des données à chiffrer. Le principe du chiffrement par flot, appelé masque jetable ou chiffre de Vernam, a été défini par Vernam en 1926 dans [155]. Dans cette méthode, une clé secrète est utilisée comme graine d'initialisation (*seed*) d'un générateur de nombres pseudo-aléatoires (GNPA)

cryptographiquement sécurisé, et donc déterministe, pour générer une séquence binaire pseudo-aléatoire, appelée flot de chiffrement et de la même taille que le message à chiffrer. Le chiffrement du message consiste alors à réaliser un ou-exclusif entre les données à chiffrer et la séquence binaire pseudo-aléatoire générée. Les algorithmes de chiffrement par bloc consistent à découper les données en clair en blocs de taille fixe (souvent 64, 128 ou 256 bits). Les blocs sont ensuite chiffrés les uns après les autres. Notons qu'il existe plusieurs modes de chiffrement par bloc [32] décrits en section 2.2.3. Les algorithmes de chiffrement par bloc les plus connus sont l'algorithme Data Encryption Standard (DES) apparu en 1976 et utilisant une clé de 56 bits [26], le Triple DES (3DES) [64] dérivé du DES utilisant deux ou trois clés de la même taille, et l'algorithme Advanced Encryption Standard (AES) [25], aujourd'hui considéré comme étant le standard en matière de chiffrement par bloc, pouvant utiliser une clé allant jusqu'à 256 bits [140].

Conçu en 1999 par Joan Daemen et Vincent Rijmen, l'AES est composé d'un ensemble d'opérations, répétées sur plusieurs itérations appelées tours. Ce nombre de tours dépend de la taille de la clé de chiffrement : 10 cycles de répétition pour les clés de 128 bits, 12 cycles de répétition pour les clés de 192 bits ou 14 cycles de répétition pour les clés de 256 bits. Afin de chiffrer une séquence de 128 bits, l'opération *AddRoundKey* est d'abord appliquée. Chaque octet de la séquence est combiné avec un bloc associé dans la clé de tour à l'aide d'une opération de ou-exclusif. Ensuite, au cours de chaque tour, quatre opérations différentes sont effectuées : *SubBytes*, *ShiftRows*, *MixColumns* et *AddRoundKey*. L'opération *SubBytes* est une étape de substitution non linéaire où chaque octet est substitué par un autre selon une table définie par convention. L'opération *ShiftRows* est une étape de transposition où les trois dernières lignes du bloc sont décalées de manière cyclique. Le *MixColumns* est une opération de mélange linéaire qui opère sur les colonnes du bloc, combinant les quatre octets de chaque colonne. Enfin, le dernier tour se compose des mêmes opérations, mais sans effectuer le *MixColumns*.

2.2.3 Modes d'opération

Les algorithmes de chiffrement par bloc peuvent prendre en charge différents modes de chiffrement, tels que ECB (Dictionnaire de codes, *Electronic Code Book*), CBC (Enchaînement des blocs, *Cipher Block Chaining*), CFB (Chiffrement à rétroaction, *Cipher FeedBack*), OFB (Chiffrement à rétroaction de sortie, *Output FeedBack*) ou CTR (Chiffrement basé sur un compteur, *CounTeR*) par exemple [32].

ECB – Dictionnaire de codes : Le mode ECB est très simple d'utilisation. Soit B_i un bloc de pixels en clair. Le chiffré B_{ci} associé est obtenu en appliquant la fonction de chiffrement $\mathcal{E}(\cdot)$ avec la clé de chiffrement K :

$$B_{ci} = \mathcal{E}_K(B_i). \tag{2.1}$$

Lors du déchiffrement avec la fonction $\mathcal{D}(\cdot)$, la valeur de B_i est reconstruite à partir des valeurs de B_{ci} et K :

$$B_i = \mathcal{D}_K(B_{ci}). \tag{2.2}$$

CBC – Enchaînement des blocs : Le mode CBC ajoute un retour d'information. En effet, une opération ou-exclusif est réalisée entre chaque bloc en clair B_i et le bloc précédemment chiffré B_{ci-1} avant d'être chiffré par la fonction de chiffrement $\mathcal{E}(\cdot)$ avec

la clé de chiffrement K. Notons qu'un vecteur d'initialisation IV [1] est utilisé lors du chiffrement du premier bloc :

$$\begin{aligned} B_{c0} &= IV, \\ B_{ci} &= \mathcal{E}_K(B_i \oplus B_{ci-1}), \end{aligned} \tag{2.3}$$

où $\oplus$ est l'opérateur ou-exclusif.

Déchiffrer B_{ci} revient alors à réaliser les opérations en ordre inverse :

$$\begin{aligned} B_{c0} &= IV, \\ B_i &= \mathcal{D}_K(B_{ci}) \oplus B_{ci-1}. \end{aligned} \tag{2.4}$$

CFB – Chiffrement à rétroaction : Le mode CFB est très proche du mode CBC. De la même façon, IV est utilisé pour la première itération. La fonction de chiffrement $\mathcal{E}(\cdot)$ est alors appliquée au bloc précédemment chiffré B_{ci-1} avant de réaliser l'opération ou-exclusif avec chaque bloc B_i :

$$\begin{aligned} B_{c0} &= IV, \\ B_{ci} &= \mathcal{E}_K(B_{ci-1}) \oplus B_i. \end{aligned} \tag{2.5}$$

B_i est reconstruit sans utiliser la fonction $\mathcal{D}(\cdot)$ à partir de B_{ci} :

$$\begin{aligned} B_{c0} &= IV, \\ B_i &= \mathcal{E}_K(B_{ci-1}) \oplus B_{ci}. \end{aligned} \tag{2.6}$$

OFB – Chiffrement à rétroaction de sortie : Le mode OFB consiste à générer une séquence binaire pseudo-aléatoire $\{S_i\}$ à l'aide de la fonction de chiffrement $\mathcal{E}(\cdot)$. Cette séquence est alors utilisée pour effectuer un chiffrement par flot de chaque bloc B_i comme décrit en section 2.2.2. Comme pour les modes précédents, un vecteur d'initialisation IV est utilisé pour la première itération :

$$\begin{aligned} S_0 &= IV, \\ S_i &= \mathcal{E}_K(S_{i-1}), \\ B_{ci} &= B_i \oplus S_i. \end{aligned} \tag{2.7}$$

Notons que l'opération de déchiffrement est identique au chiffrement car l'opération ou-exclusif est symétrique :

$$\begin{aligned} S_0 &= IV, \\ S_i &= \mathcal{E}_K(S_{i-1}), \\ B_i &= B_{ci} \oplus S_i. \end{aligned} \tag{2.8}$$

CTR – Chiffrement basé sur un compteur : Le mode CTR consiste à générer une séquence binaire pseudo-aléatoire en chiffrant les valeurs successives d'un compteur. Cette séquence est alors utilisée pour effectuer un chiffrement par flot de chaque bloc.

1. Même si l'utilisation d'un vecteur d'initialisation permet un chiffrement par clé dynamique, cette notion ne doit pas être confondue avec celle de clé secrète. En effet, le vecteur d'initialisation est public et sert seulement à introduire de la diversité lors du chiffrement.

Ces différents modes de chiffrement ont chacun des avantages et des inconvénients. Par exemple, bien que simple à mettre en place, le mode ECB n'est pas sécurisé car deux blocs identiques en clair sont chiffrés de la même façon. Par ailleurs, avec les modes ECB et OFB, si un bloc chiffré est altéré, le bloc en clair associé ne peut pas être reconstruit. En revanche, cela n'a aucun impact sur le déchiffrement des blocs voisins. Ce n'est pas le cas avec les modes CBC et CFB. Notons que ces différentes propriétés sont discutées et illustrées dans le chapitre 6.

2.2.4 Chiffrement asymétrique

En chiffrement asymétrique, deux clés sont utilisées : une clé publique et une clé privée. Ces clés sont générées par l'utilisateur qui souhaite recevoir les messages. Elles sont reliées mathématiquement du fait que l'opération de déchiffrement correspond à l'inverse de l'opération de chiffrement. Pour cela, des fonctions à sens unique sont utilisées. Ces fonctions ont la particularité d'être faciles à évaluer, mais difficiles à inverser. La clé publique est générée en calculant l'image de la clé privée par la fonction à sens unique choisie. En d'autres termes, cela signifie que si la clé publique est connue, alors il est très difficile de retrouver (*i.e.* impossible à calculer en temps polynômial) la clé privée associée. Ainsi, la clé publique est transmissible sans restriction, tandis que la clé privée doit rester secrète et n'est jamais transmise.

La première méthode de chiffrement asymétrique, nommée RSA du nom de ses inventeurs Rivest, Shamir et Adleman, a été développée en 1978 [124]. Cette approche se base sur le problème de la factorisation des grands nombres en produit de deux nombres premiers :

- Alice choisit deux nombres premiers distincts p et q.
- Elle calcule le module de chiffrement $n = pq$.
- Elle calcule ensuite $\phi(n) = (p-1)(q-1)$, l'indicatrice d'Euler en n.
- Elle choisit un entier naturel e premier avec $\phi(n)$ et strictement inférieur à $\phi(n)$. Cet entier e est appelé exposant de chiffrement.
- Elle calcule d, inverse de e modulo $\phi(n)$, en utilisant l'algorithme d'Euclide étendu. L'entier d est appelé exposant de déchiffrement.
- Ainsi, le couple (n, e) est la clé publique et est largement diffusé, tandis que d est la clé privée, connue seulement par Alice.
- Bob choisit un message m (avec $m < n$) qu'il souhaite envoyer à Alice. Il calcule alors c, le chiffré de m, en utilisant la clé publique d'Alice (n, e) : $c = m^e \mod (n)$.
- Avec sa clé privée d, Alice déchiffre c et retrouve la valeur de m : $m = c^d \mod (n)$.

D'autres méthodes sont basées sur des problèmes difficiles différents. Par exemple, la méthode d'El Gamal s'appuie sur l'utilisation du logarithme discret [33] :

- Alice choisit un nombre premier p et deux entiers naturels a, tel que $0 \leq a \leq p-2$, et x, tel que $0 \leq x \leq p-1$.
- Elle calcule ensuite $n = x^a \mod (p)$.
- La clé publique est alors le triplet (p, x, n) et la clé privée est a.

— Bob choisit un message m (avec $m < p$) qu'il souhaite envoyer à Alice. Il commence par générer aléatoirement un entier k, tel que $0 \leq k \leq p-1$. Notons que le choix d'un tel k garantit la propriété de non déterminisme du cryptosystème d'El Gamal. Il calcule alors (c_1, c_2), le chiffré de m, en utilisant la clé publique d'Alice (p, x, n) : $c_1 = x^k \mod (p)$ et $c_2 = m \cdot n^k \mod (p)$.

— Alice déchiffre (c_1, c_2) à l'aide de sa clé privée a et retrouve la valeur de m : $m = c_1^{p-1-a} \cdot c_2 \mod (p)$.

En outre, le cryptosystème de Paillier utilise la résiduosité quadratique [91] :

— Alice choisit deux nombres premiers distincts p et q, tels que pq et $(p-1)(q-1)$ sont premiers entre eux, *i.e.* $pgcd(pq, (p-1)(q-1)) = 1$.

— Elle calcule le module de chiffrement $n = pq$ et $\lambda = ppcm((p-1), (q-1))$.

— Elle sélectionne alors un nombre $g \in (\mathbb{Z}/n^2\mathbb{Z})^*$, tel qu'il existe un nombre μ égal à $(L(g^\lambda \mod (n^2)))^{-1} \mod (n)$, où $L(x) = \frac{x-1}{n}$, avec $x \in \mathbb{N}^*$.

— La clé publique est alors (n, g) et la clé privée est (λ, μ).

— Bob choisit un message m (avec $m < n$) qu'il souhaite envoyer à Alice. Il commence par générer aléatoirement un entier $r \in (\mathbb{Z}/n\mathbb{Z})^*$, garantissant ainsi la propriété de non déterminisme du cryptosystème de Paillier. Il calcule alors c le chiffré de m, en utilisant la clé publique d'Alice (n, g) : $c = g^m \cdot r^n \mod (n^2)$. Notons que l'élévation au carré de n implique une expansion de la taille de c par rapport à celle de m (facteur 2).

— Alice déchiffre c à l'aide de sa clé privée (λ, μ) et retrouve la valeur de m : $m = L(c^\lambda \mod (n^2)) \cdot \mu \mod (n)$.

Notons que la taille des clés utilisées dans les cryptosystèmes asymétriques varie entre 1024 et 4096 bits.

2.2.5 Chiffrement homomorphe

Une fonction $f(\cdot)$ est dite homomorphe si la condition suivante est vérifiée [37] :

$$f(x_1 \triangle x_2) = f(x_1) \square f(x_2), \tag{2.9}$$

où $\triangle$ et $\square$ sont des opérations arithmétiques.

Par extension, un algorithme de chiffrement $\mathcal{E}(\cdot)$ est dit homomorphe (avec $\mathcal{D}(\cdot)$ l'algorithme de déchiffrement associé), si lorsque les versions chiffrées de deux messages en clair m_1 et m_2 sont connues, il est possible d'obtenir le chiffré d'une opération entre ces deux messages :

$$\mathcal{D}\left(\mathcal{E}\left(m_1 \triangle m_2\right)\right) = \mathcal{D}\left(\mathcal{E}\left(m_1\right)\right) \square \mathcal{D}\left(\mathcal{E}\left(m_2\right)\right). \tag{2.10}$$

Notons que $\triangle$ et $\square$ peuvent désigner une addition, une soustraction ou une multiplication et ne sont pas nécessairement les mêmes entre les messages en clair et leurs versions chiffrées.

De nombreux cryptosystèmes asymétriques sont homomorphes :

— RSA [124] est partiellement homomorphe vis-à-vis de la multiplication :

$$\begin{aligned} \mathcal{E}(m_1 \cdot m_2) &= (m_1 \cdot m_2)^e \mod (n), \\ &= m_1^e \cdot m_2^e \mod (n), \\ &= \mathcal{E}(m_1) \cdot \mathcal{E}(m_2). \end{aligned} \tag{2.11}$$

— Le cryptosystème d'El Gamal [33] est également partiellement homomorphe vis-à-vis de la multiplication.

— Le cryptosystème de Paillier [91] est un homomorphisme additif :

$$\begin{aligned} \mathcal{E}(m_1 + m_2) &= g^{m_1+m_2} \cdot (r_1 \cdot r_2)^n \mod (n^2), \\ &= (g^{m_1} \cdot r_1^n) \cdot (g^{m_2} \cdot r_2^n) \mod (n^2), \\ &= \mathcal{E}(m_1) \cdot \mathcal{E}(m_2). \end{aligned} \tag{2.12}$$

L'avantage principal de la propriété d'homomorphisme est de permettre d'effectuer des opérations dans le domaine chiffré sans révéler aucune information sur le contenu dans le domaine clair. Le chapitre 3 décrit plus en détail cette problématique.

2.2.6 Comparaison et discussion

Les algorithmes symétriques et asymétriques possèdent tous deux des avantages et des inconvénients. Une première différence notable est que le chiffrement symétrique nécessite un échange de clés, alors que le chiffrement asymétrique pallie ce problème. Par ailleurs, les algorithmes asymétriques sont plus compliqués à implémenter, plus lents et nécessitent de plus grandes ressources matérielles que les algorithmes symétriques. Le logiciel PGP propose d'exploiter les avantages de ces deux classes d'algorithmes cryptographiques en proposant un système hybride [177]. Pour envoyer un message de façon sécurisée, une clé secrète à usage unique est générée puis utilisée pour chiffrer le message à l'aide d'un algorithme de chiffrement symétrique. Cette clé est alors chiffrée en utilisant un algorithme de chiffrement asymétrique tel que RSA. Enfin, l'expéditeur envoie le message et la clé chiffrés au destinataire. Notons que le temps d'exécution de l'algorithme asymétrique est limité car seul le chiffrement de la clé est réalisé de cette manière.

Les derniers travaux de recherche en cryptologie concernent la cryptographie quantique et post-quantique. La cryptographie quantique consiste à utiliser les propriétés de la physique quantique – en particulier le principe d'Heisenberg – et de la théorie de l'information, pour développer des algorithmes de cryptographie. Par exemple, la distribution quantique de clés permet de distribuer une clé entre deux interlocuteurs distants, à la demande. Si l'information est interceptée et lue par un tiers, des phénomènes physiques introduisent naturellement des erreurs et permettent la détection de l'attaque. La cryptographie post-quantique, quant à elle, vise a créer des méthodes de cryptographie résistantes à un attaquant possédant un ordinateur quantique. En effet, avec un tel calculateur, les algorithmes cryptographiques dont la sécurité repose sur le problème du logarithme discret ou sur le problème de la factorisation en produit de deux nombres entiers sont vulnérables. Néanmoins, la technologie quantique n'est pas encore opérationnelle. Ainsi, malgré de nombreux travaux de recherche théoriques,

les algorithmes développés ne sont que peu utilisés en pratique à ce jour. Dans ce contexte, le NIST (*National Institute of Standards and Technology*) a lancé une compétition internationale visant à établir une standardisation des algorithmes cryptographiques post-quantiques en novembre 2017 pour une durée de 4 ou 5 ans[2].

2.3 Chiffrement d'images

Le but du chiffrement d'images est de garantir la sécurité visuelle du contenu en clair d'une image. Le chiffrement est dit « total » lorsque l'intégralité du contenu original de l'image est protégée après l'opération de chiffrement. Dans ce cas, aucune information relative à l'image en clair ne peut être extraite de l'image chiffrée. Par ailleurs, lorsque seulement une partie des données est sélectionnée pour être chiffrée, le chiffrement est dit « sélectif ». Enfin, il est dit « partiel », lorsque seulement une zone spécifique de l'image est chiffrée et que les pixels en dehors de cette zone restent en clair. La fig. 2.1 illustre la différence entre ces trois types de chiffrement.

(a) (b) (c) (d)

FIGURE 2.1 – Illustration de la différence entre les trois types de chiffrement : a) Image originale, b) Chiffrement total, c) Chiffrement sélectif, d) Chiffrement partiel.

Par ailleurs, les méthodes de chiffrement d'images doivent vérifier deux propriétés :

— Conformité au format : le format de l'image chiffrée doit être le même que celui de l'image originale,

— Préservation de la taille : la taille de l'image chiffrée doit être identique à (ou très proche de) celle de l'image originale.

La première propriété indique que les données avant et après chiffrement doivent être structurées de la même manière. Ainsi, les méthodes classiques de chiffrement ne peuvent pas être directement appliquées aux images sans être adaptées pour tenir compte de la spécificité des formats d'images. Par ailleurs, notons que la propriété de conformité au format implique que les données après chiffrement peuvent être visualisées avec les mêmes éditeurs d'images que les données originales. Concernant

2. https ://csrc.nist.gov/Projects/post-quantum-cryptography

la deuxième propriété, nous observons qu'une augmentation limitée de la taille des données après chiffrement peut être tolérée dans certaines applications.

Par ailleurs, d'après Shannon, une méthode de chiffrement doit introduire de la confusion et de la diffusion [134]. Le principe de confusion appliqué aux images correspond à une volonté de rendre la relation entre la clé de chiffrement et l'image chiffrée la plus complexe possible. La propriété de diffusion, quant à elle, indique que la redondance statistique entre les pixels d'une image en clair doit être dissipée dans les statistiques de l'image chiffrée. En d'autres termes, la corrélation entre les pixels d'une image en clair ne doit pas se retrouver dans l'image chiffrée. Pour introduire de la confusion et augmenter la diffusion, des opérations de permutation et de substitution des pixels sont réalisées.

Si les algorithmes classiques de chiffrement décrits en section 2.2 peuvent être adaptés aux images, des algorithmes spécifiques ont également été développés pour ce type de données. Notons que ces algorithmes s'appuient généralement sur une méthode de chiffrement symétrique. Du fait du grand volume de données à traiter, le chiffrement asymétrique est moins utilisé. En effet, celui-ci est bien plus coûteux en temps de calcul et implique une augmentation de la taille des données originales. Dans un premier temps, nous détaillons les approches naïves de chiffrement d'images, basées sur l'utilisation d'un GNPA et sur le principe de substitution ou de permutation (section 2.3.1). Ensuite, nous expliquons comment la théorie du chaos a été appliquée au chiffrement d'images (section 2.3.2). Enfin, nous décrivons les approches de Chiffrement-puis-Compression, permettant de chiffrer une image puis de la compresser (section 2.3.3).

2.3.1 Méthodes naïves

Les approches de chiffrement d'images basées sur les opérations de substitution et/ou de permutation impliquent l'utilisation de GNPA cryptographiquement sécurisés, comme décrits en section 2.2. Les données fournies à ces générateurs sont une clé secrète K, le type des éléments et la longueur de la séquence à générer.

Efficaces et faciles à implémenter, l'objectif des méthodes de chiffrement par mélange est de transformer une image en clair en une image inintelligible, en permutant les positions des pixels. Soit une image $I = \{p(i)\}_{0 \leq i < m \times n}$, de $m \times n$ pixels. Le chiffrement par mélange de I s'effectue de la façon suivante. A l'aide d'une clé secrète K, un GNPA est utilisé pour définir les nouvelles positions des pixels de I. Une séquence $S = \{s(i)\}_{0 \leq i < m \times n}$, de $m \times n$ positions pseudo-aléatoires $s(i)$ telles que $0 \leq s(i) < m \times n$ et $\forall\, i,\ \forall\, j,\ 0 \leq j < m \times n,\ i \neq j \Rightarrow s(i) \neq s(j)$ (de manière à éviter les collisions), est générée. L'image chiffrée $I_c = \{p_c(i)\}_{0 \leq i < m \times n}$ est alors obtenue en recopiant les valeurs des pixels de I à des positions pseudo-aléatoires données par la séquence S :

$$p_c(i) = p(s(i)). \tag{2.13}$$

Par ailleurs, certains auteurs suggèrent de permuter aléatoirement non pas l'emplacement de tous les pixels de l'image, mais des lignes et des colonnes de l'image [152, 98] ou encore des blocs de pixels [161].

Des méthodes de chiffrement par substitution ont également été développées. Elles sont basées sur l'opération ou-exclusif entre la séquence pseudo-aléatoire générée et le contenu d'une image en clair. Par ailleurs, cette opération peut s'effectuer de deux façons

différentes. L'image peut être chiffrée pixel par pixel, bit par bit (par exemple du bit le moins significatif au plus significatif), ou plan binaire par plan binaire, (par exemple du plan le moins significatif au plus significatif). Soit une image $I = \{p(i)\}_{0 \leq i < m \times n}$, de $m \times n$ pixels, le chiffrement pixel par pixel de I s'effectue alors de la façon suivante. A l'aide d'une clé secrète K et d'un GNPA, une séquence binaire pseudo-aléatoire $S = \{s(i)\}_{0 \leq i < m \times n}$, de $m \times n$ octets $s(i)$, est générée. Comme présenté en fig. 2.2, l'image chiffrée $I_c = \{p_c(i)\}_{0 \leq i < m \times n}$ est alors obtenue en effectuant un ou-exclusif entre chacun des pixels $p(i)$ de l'image I et l'octet $s(i)$ associé dans la séquence pseudo-aléatoire S :

$$p_c(i) = p(i) \oplus s(i). \tag{2.14}$$

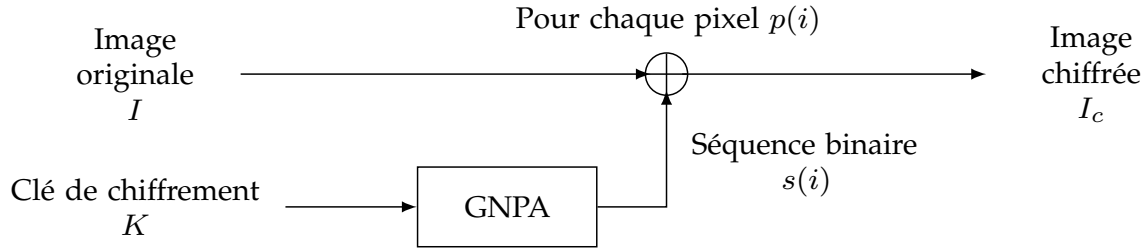


FIGURE 2.2 – Chiffrement par substitution.

La figure 2.3 présente une image en clair, ses versions chiffrées en utilisant un chiffrement par mélange et un chiffrement par substitution, ainsi que les histogrammes associés. En fig. 2.3.d, nous pouvons remarquer que la distribution des pixels de l'image en clair comporte un certain nombre de modes relatifs au contenu. Par ailleurs, comme l'illustre la fig. 2.3.e, l'histogramme des pixels de l'image chiffrée par mélange est identique à celui de l'image en clair. Cette méthode de chiffrement, bien que facile à mettre en place, a l'inconvénient de préserver certaines propriétés statistiques de l'image originale. En revanche, la distribution des pixels de l'image chiffrée par substitution (fig. 2.3.f) est proche d'une distribution uniforme. En effet, si les pixels voisins ont des valeurs similaires et sont fortement corrélés dans le domaine clair, ce n'est pas le cas dans le domaine chiffré en utilisant cette méthode. Ainsi, contrairement aux résultats obtenus en utilisant le chiffrement par mélange, l'analyse de l'image chiffrée par substitution montre que le contenu original de l'image en clair est visuellement confidentiel.

2.3.2 Méthodes basées chaos

Avec le développement rapide de la théorie et des applications au chaos, de nombreuses techniques de chiffrement basées sur la théorie du chaos ont été présentées. La première méthode proposée, créée par Arnold et Avez, date de 1967 et porte le nom de carte du chat d'Arnold [2]. Après cela, Scharinger et Pichler ont appliqué la carte du boulanger au chiffrement d'images [130]. La figure 2.4 illustre ces deux méthodes très connues.

Fridrich a alors étendu la version discrétisée de cette carte en trois dimensions (3D) et l'a associée à un mécanisme de diffusion [38, 39]. Dans la plupart des cas, les algorithmes de chiffrement développés dans cette section intègrent à la fois des

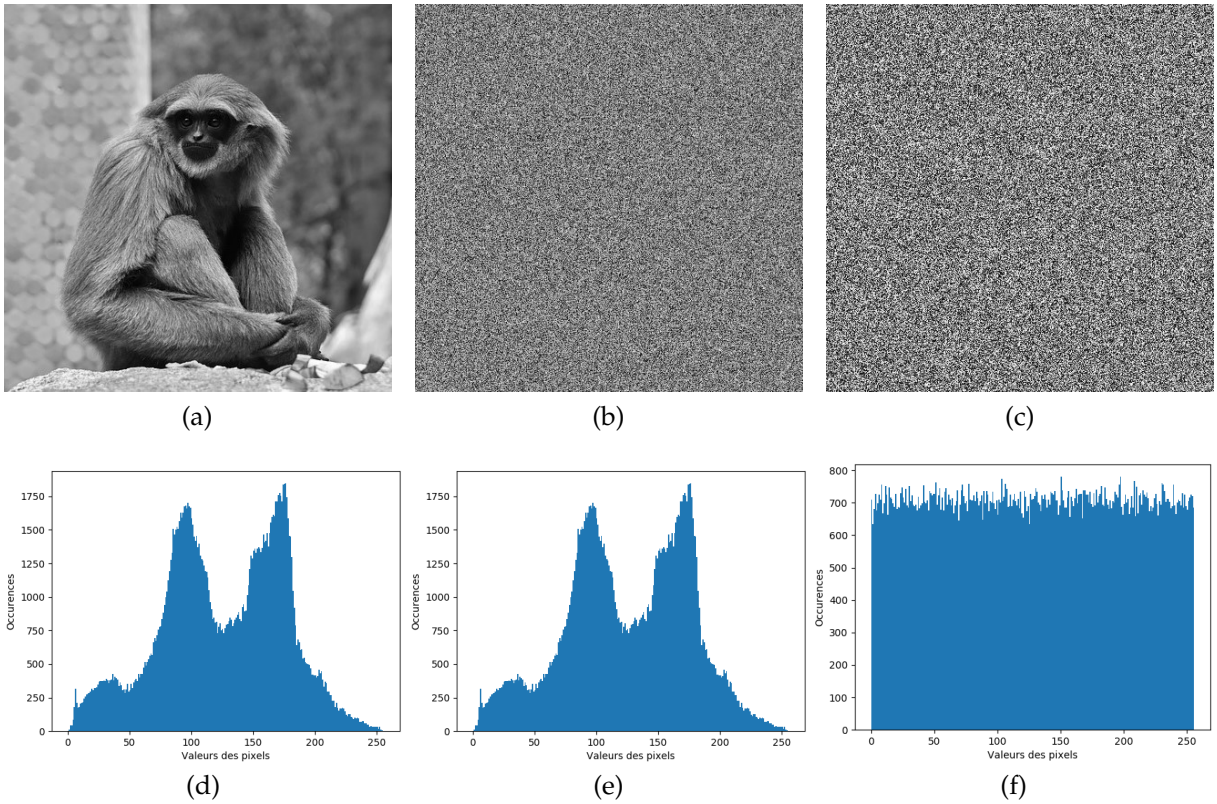
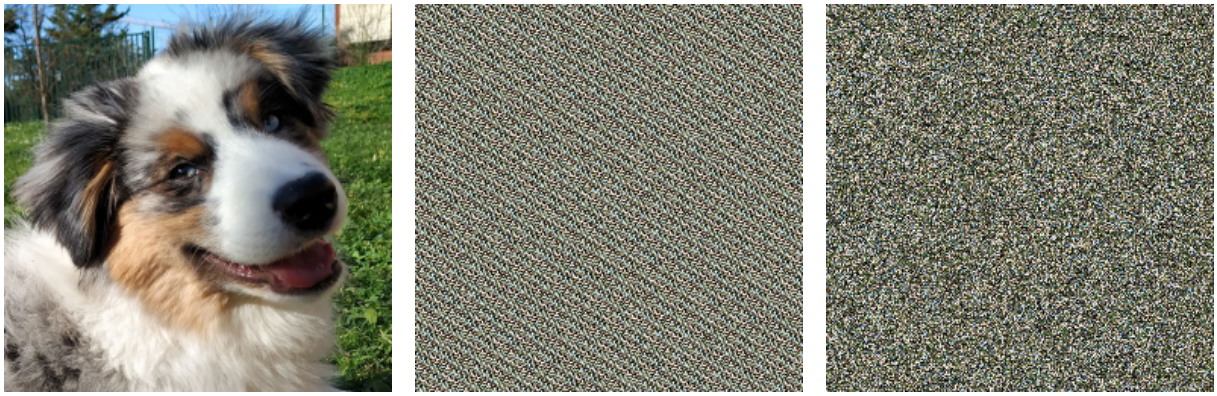


FIGURE 2.3 – Illustration du chiffrement d'images par mélange et par substitution : a) Image originale, b) Image (a) chiffrée par mélange des pixels, c) Image (a) chiffrée par substitution, pixel par pixel, d) Histogramme associé à (a), e) Histogramme associé à (b), f) Histogramme associé à (c).

mécanismes de permutation et de substitution et sont basés sur la combinaison de plusieurs cartes chaotiques. Les cryptosystèmes basés chaos peuvent être divisés en deux classes distinctes. Dans la première catégorie, un pixel est considéré comme étant le plus petit élément [14, 82, 47] et, dans la seconde, un pixel est composé de bits, sur lesquels des opérations binaires sont effectuées [167, 176]. Dans [14], les auteurs emploient une carte du chat d'Arnold en 3D et dans [82], une carte du boulanger en 3D est utilisée pour permuter les positions des pixels durant la phase de substitution. Guan *et al.* ont appliqué à leur tour la carte du chat d'Arnold pour mélanger les positions des pixels d'une image dans le domaine spatial et après cela, ils ont utilisé le système chaotique défini dans [15] pour modifier les valeurs des pixels [47]. Pour réduire le temps d'exécution, Xiang *et al.* ont défini une méthode de chiffrement sélectif chaotique, où ils suggèrent de chiffrer seulement les quatre bits les plus significatifs de chaque pixel et laissent les quatre bits les moins significatifs en clair [167]. Zhu *et al.* proposent en 2011 un cryptosystème où la carte du chat d'Arnold est utilisée pour effectuer des permutations au niveau binaire, ce qui permet de modifier à la fois l'emplacement et les valeurs des pixels de l'image [176]. L'emploi d'une carte logistique permet ensuite d'introduire de la diffusion.



(a) (b) (c)

FIGURE 2.4 – Illustration de deux méthodes de chiffrement d'images basées sur la théorie du chaos : a) Image originale, b) Carte du chat d'Arnold de l'image (a) [2], c) Carte du boulanger de l'image (a) [130].

2.3.3 Chiffrement-puis-Compression

Les méthodes de Chiffrement-puis-Compression consistent à chiffrer une image en premier lieu, puis la compresser. Le défi de ce type de méthodes est que la compression doit être effectuée dans le domaine chiffré, sans avoir accès à la clé secrète, ni à l'image originale en clair. A première vue, il semble impossible de compresser les données chiffrées, car les redondances entre les pixels ont été supprimées par l'opération de chiffrement. Pourtant, en 2004, Johnson *et al.* ont affirmé que les données chiffrées par flot étaient compressibles, sans en altérer la sécurité visuelle [62]. En plus des résultats théoriques, des algorithmes pratiques pour compresser sans perte les images binaires chiffrées ont été proposés. Par la suite, Kumar et Makur ont appliqué cette méthode aux erreurs de prédiction calculées dans le domaine chiffré et ainsi amélioré le taux de compression [68]. Lazzeretti et Barni ont présenté plusieurs méthodes de compression sans perte d'images chiffrées en niveaux de gris et en couleur [72]. Les auteurs décomposent les images en plans binaires et exploitent la corrélation intra et inter-plans. Par ailleurs, d'autres méthodes de Chiffrement-puis-Compression sont basées sur un mélange des blocs d'une image [70, 69]. L'image originale est divisée en blocs de 16×16 pixels qui sont alors permutés pseudo-aléatoirement à l'aide d'une première clé secrète. Une deuxième clé secrète est utilisée pour effectuer des rotations et/ou des inversions sur les blocs. Une transformation est également appliquée aux pixels de chaque bloc : à l'aide d'une troisième clé secrète, le bit de poids fort de certains pixels est modifié. Enfin, si les images sont en couleur, les trois composantes sont permutées d'après la valeur d'une quatrième clé secrète. En utilisant cette méthode de chiffrement, l'image chiffrée obtenue est robuste à une compression JPEG classique. Malgré les efforts considérables déployés ces dernières années, les systèmes de Chiffrement-puis-Compression existants sont encore très insuffisants en termes de performances de compression, par rapport aux codeurs d'images et vidéo avec ou sans perte prenant en entrée des données en clair. De plus, un grand nombre d'entre eux ne sont pas sécurisés. En particulier, beaucoup sont vulnérables aux attaques par puzzle [18]. Le format d'images compressées le plus utilisé est JPEG. Ainsi, de nombreuses méthodes proposent de réaliser conjointement le

chiffrement et la compression JPEG. Dans le chapitre 7, nous décrivons les enjeux et le fonctionnement des approches de crypto-compression d'images.

2.3.4 Évaluation du niveau de sécurité visuelle

Une fois qu'une image a été chiffrée, il est nécessaire d'évaluer son niveau de sécurité visuelle. En 2009, Engel *et al.* ont défini trois niveaux de sécurité visuelle en fonction de l'application visée [34] :

1. **Niveau transparent** : la haute résolution de l'image originale est préservée mais son contenu peut être pré-visualisé grâce à une version dégradée en clair,
2. **Niveau suffisant** : le contenu de l'image originale est protégé mais certaines formes et contours peuvent être distingués,
3. **Niveau confidentiel** : aucune information relative au contenu de l'image en clair ne peut être extraite de l'image chiffrée.

A ce jour, aucune métrique n'a été définie pour l'évaluation spécifique du niveau de sécurité visuelle des images chiffrées. De ce fait, en se basant sur les travaux de Preishuber *et al.* [97], nous décrivons les métriques utilisées dans de nombreux articles pour démontrer expérimentalement la sécurité visuelle des méthodes de chiffrement d'images.

Coefficient de corrélation : Une métrique classique consiste à observer la corrélation entre les pixels dans les directions horizontale, verticale et diagonale. M paires de pixels voisins (x_i, y_i) dans les trois directions, avec $x_i \in x$ et $y_i \in y$, sont ainsi choisies pour le calcul du coefficient de corrélation :

$$corr_{x,\,y} = \frac{\frac{1}{M}\sum_{i=1}^{M}(x_i - E(x)) \times (y_i - E(y))}{\sqrt{\frac{1}{M}\sum_{i=1}^{M}(x_i - E(x))^2}\sqrt{\frac{1}{M}\sum_{i=1}^{M}(y_i - E(y))^2}}. \tag{2.15}$$

où $E(x)$ est la moyenne de l'ensemble x.

La valeur de ce coefficient de corrélation est comprise en -1 et 1, où -1 et 1 indiquent une forte corrélation et 0, l'absence de corrélation. Comme les valeurs des pixels voisins dans le domaine clair sont fortement corrélées, $corr_{x,\,y}$ est généralement élevé dans l'image originale en clair. En revanche, il doit être proche de zéro dans le domaine chiffré.

Entropie de Shannon [133] : L'entropie de Shannon est une mesure de quantité d'information utilisée pour évaluer le caractère aléatoire de la distribution des pixels d'une image chiffrée :

$$H(I) = -\sum_{k=0}^{2^l-1} P(\alpha_k)\log_2(P(\alpha_k)), \tag{2.16}$$

où I est une image de $m \times n$ pixels codés sur 2^l valeurs α_k $(0 \leq k < 2^l)$ et $P(\alpha_k)$ est la probabilité associée à α_k. La valeur de l'entropie est exprimée en bits-par-pixel (*bpp*) et comprise entre 0 *bpp* et $\log_2(2^l) = l$ *bpp*, lorsque la distribution des pixels est parfaitement uniforme. En général, les images en niveaux de gris sont codées sur 256 valeurs. Dans ce cas, l'entropie maximale est alors de $\log_2(256) = 8$ *bpp*. Ainsi, la valeur de l'entropie

d'une image chiffrée doit être très proche de la valeur de l'entropie maximale.

Test du χ^2 : Le caractère uniforme de la distribution des pixels d'une image chiffrée peut également être évalué par le test du chi-carré donné par :

$$\chi^2 = 2^l \sum_{k=0}^{2^l-1} \left(P(\alpha_k) - \frac{1}{2^l} \right)^2, \tag{2.17}$$

où les pixels de l'image sont codés sur 2^l valeurs α_k $(0 \leq k < 2^l)$ et $P(\alpha_k)$ est la probabilité associée à α_k. Plus la valeur obtenue est faible, plus la distribution des pixels de l'image chiffrée est proche de la distribution uniforme, indiquant un niveau de sécurité visuelle plus élevé. Notons que la racine carrée de la valeur χ^2 est souvent considérée.

Taux de pixels modifiés (NPCR, Number of Changing Pixel Rate) [165] : Le NPCR entre deux images de taille $m \times n$ pixels $p(i,j)$ et $p'(i,j)$ $(0 \leq i < m,\ 0 \leq j < n)$ est donné par :

$$\text{NPCR} = \frac{\sum_{i=0}^{m-1} \sum_{j=0}^{n-1} d(i,j)}{m \times n} \times 100, \tag{2.18}$$

où $d(i,j)$ est défini par :

$$d(i,j) = \begin{cases} 1, & \text{si } p(i,j) \neq p'(i,j), \\ 0, & \text{sinon.} \end{cases} \tag{2.19}$$

Il est exprimé en $\%$ et est utilisé pour savoir à quel point une image chiffrée diffère de l'image originale. Ainsi, plus sa valeur est proche de 100%, plus les deux images sont différentes et donc, plus le niveau de sécurité visuelle est élevé.

Moyenne unifiée des changements d'intensité (UACI, Unified Averaged Changed Intensity) [165] : L'UACI est aussi utilisée pour mesurer la différence entre deux images de $m \times n$ pixels et dont les pixels $p(i,j)$ et $p'(i,j)$ $(0 \leq i < m,\ 0 \leq j < n)$ sont codés sur 2^l valeurs de niveaux de gris :

$$\text{UACI} = \frac{100}{m \times n} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} \frac{|p(i,j) - p'(i,j)|}{2^l - 1}. \tag{2.20}$$

Elle est également exprimée en $\%$. Plus sa valeur est haute, plus le niveau de sécurité visuelle est haut. Notons que sa valeur idéale dépend de la gamme de tons de l'image. Cette métrique peut être utilisée pour effectuer des tests de sensibilité de la clé utilisée lors du chiffrement. L'image originale est chiffrée avec deux clés dont un seul bit diffère et les deux images chiffrées associées sont comparées. Dans ce cas, la valeur optimale de l'UACI est de $33,33\%$ [97]. En revanche, si l'UACI est utilisée pour comparer une image originale en clair et sa version chiffrée, la valeur est souvent plus basse. Notons qu'il n'existe pas de critère statistique de décision pour ce test, dont l'observation des valeurs optimales est purement expérimentale.

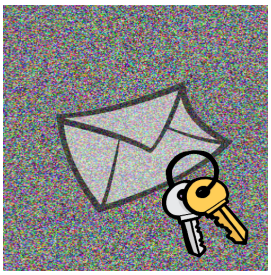
Le NPCR et l'UACI peuvent aussi être utilisés pour analyser la robustesse aux attaques

différentielles. Ce type d'attaques est utilisé pour tester la sensibilité d'un cryptosystème à des changements minimes dans l'image prise en entrée. Généralement, un bit de l'image originale est modifié. L'image originale et l'image modifiée sont alors chiffrées en utilisant la même clé. Les images chiffrées obtenues sont ensuite comparées : bien que les images en clair soient presque identiques, les images chiffrées doivent être très différentes.

2.4 Conclusion

Dans ce chapitre, nous avons présenté les domaines du chiffrement classique et du chiffrement d'images. Tout d'abord, nous avons introduit les fondamentaux de la cryptographie. Nous avons ensuite développé les deux catégories de méthodes de chiffrement, à savoir le chiffrement symétrique et le chiffrement asymétrique. En outre, nous avons décrit les différents modes de chiffrement pouvant être utilisés, ainsi que la propriété d'homomorphisme que de certaines méthodes de chiffrement asymétrique possèdent. Par ailleurs, les méthodes classiques de chiffrement ne peuvent pas être appliquées directement aux images. En effet, elles doivent être adaptées pour tenir compte de leur format et de leurs spécificités afin de permettre la visualisation des données chiffrées (contrainte dans certaines applications). Un intérêt particulier est porté à la préservation de la taille de l'image originale pour limiter au maximum l'expansion des données, déjà volumineuses avant l'opération de chiffrement. Ainsi, nous avons détaillé différentes approches de chiffrement d'images non compressées et les métriques utilisées pour l'évaluation du niveau de sécurité. Notons qu'un état-de-l'art des méthodes de chiffrement d'images compressées – appelées méthodes de crypto-compression – est proposé dans le chapitre 7.

Pendant la transmission ou le stockage des images chiffrées, il est souvent nécessaire de les analyser et de les traiter sans avoir accès à leur contenu en clair et sans connaître la clé de chiffrement utilisée. Dans ce contexte, des méthodes d'analyse et de traitement des images dans le domaine chiffré ont été proposées et sont décrites dans les chapitres 3 – 7.



CHAPITRE 3

Insertion de données cachées dans le domaine chiffré

Sommaire

3.1 Introduction

Ces dernières années, avec le développement de l'informatique en nuage (*cloud computing*), de plus en plus d'utilisateurs téléchargent leurs données personnelles sur des serveurs distants. Cependant, cela peut entraîner d'importantes failles de sécurité, où la confidentialité, l'authentification et l'intégrité sont menacées. Pour pallier ces problèmes, les données multimédia sont chiffrées avant leur transmission et leur stockage. Dans ce chapitre, nous nous intéressons à la problématique du traitement de ces données multimédia chiffrées, et plus particulièrement à l'insertion de données cachées dans le domaine chiffré.

En section 3.2, nous commençons par décrire les principales applications visées par le traitement des données multimédia dans le domaine chiffré. Dans les sections suivantes, nous nous intéressons particulièrement aux méthodes d'insertion de données cachées dans le domaine chiffré (IDCDC). Ainsi, la section 3.3 présente les motivations de ces méthodes. La section 3.4 développe leurs différentes classes et caractéristiques. Les principales méthodes de l'état-de-l'art sont décrites en section 3.5 et comparées en section 3.6. Enfin, nous concluons en section 3.7.

3.2 Traitement des données multimédia dans le domaine chiffré

Pour des raisons de sécurité, de plus en plus de données numériques sont chiffrées avant d'être transférées ou archivées. Durant la transmission ou l'archivage de ces données numériques, il est souvent nécessaire de les analyser ou de les traiter directement dans le domaine chiffré et sans connaître leur contenu original en clair [35].

Dans cette section, nous détaillons trois des applications visées par l'analyse et le traitement des données multimédia dans le domaine chiffré illustrées en fig. 0.1, à savoir le partage de secret visuel (section 3.2.1), l'indexation et la recherche de contenus multimédia dans des bases de données chiffrées (section 3.2.2) et l'insertion de données cachées dans le domaine chiffré (section 3.2.3). Notons que la correction d'images chiffrées bruitées est développée dans le chapitre 6 et qu'une méthode de recompression d'images crypto-compressées est proposée dans le chapitre 7.

3.2.1 Partage de secret visuel

Il existe deux catégories de méthodes permettant le partage d'un secret visuel, à savoir la cryptographie visuelle proposée par Naor et Shamir en 1994 [86] et le partage d'images secrètes proposé par Thien et Lin en 2002 [145]. La cryptographie visuelle (*visual cryptography*) consiste à partager de façon sécurisée une information visuelle (texte ou image) entre plusieurs personnes. Dans l'approche [86] développée pour les images binaires, deux images appelées parties (*shares*) sont générées à l'issue du partage. Sur celles-ci, les pixels noirs de l'image secrète sont protégés et les pixels blancs sont rendus aléatoires. Le secret est alors reconstruit en effectuant un ou-exclusif entre les deux parties. Par ailleurs, le partage d'image secrète (*secret image sharing*) est inspiré des méthodes de partage de secret développées indépendamment en 1979 par Blakley [6] et

Shamir [132]. Basé généralement sur le concept d'interpolation polynômiale, il permet de partager une image entre n utilisateurs de façon sécurisée [145]. Chaque utilisateur reçoit une partie personnelle sous la forme d'une image. Cette partie est unique et semble visuellement avoir été générée aléatoirement. Le contenu original ne peut alors être reconstruit qu'après réunion d'au moins k de ces parties avec $k \leq n$. Avec $k-1$ parties, il est en effet impossible d'obtenir quelconque information. Le paramètre k peut être plus ou moins élevé suivant le niveau de confiance au sein du groupe de partage. Notons que l'image secrète reconstruite I' est très similaire à l'image secrète originale I et peut même être strictement identique à I avec les méthodes dites parfaites. Le processus de partage d'image secrète est illustré en figure 3.1.

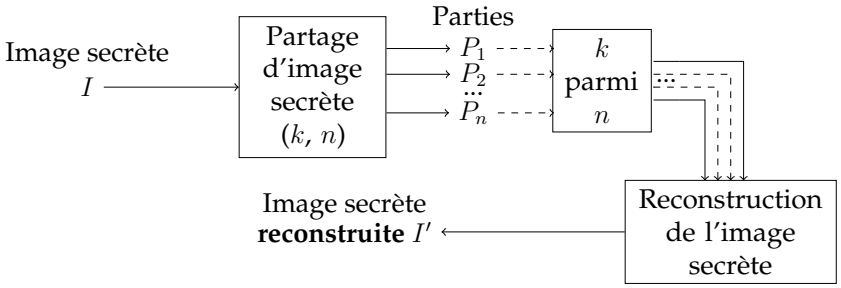


FIGURE 3.1 – Illustration du processus de partage d'image secrète.

3.2.2 Recherche et indexation dans des bases d'images chiffrées

Le chiffrement recherchable (*searchable encryption*) permet le stockage sécurisé d'une base de données sur un serveur non fiable, tout en conservant les fonctionnalités de recherche et d'indexation de contenu. Avec le développement de l'informatique en nuage, l'efficacité du chiffrement recherchable est devenue cruciale. Ainsi, les méthodes utilisées doivent permettre d'assurer la sécurité des données stockées sans augmenter leur taille. De plus, le coût de calcul des opérations nécessaires à la recherche et l'indexation doit être faible. Si la plupart des méthodes existantes ont été développées pour la recherche de documents textuels, les méthodes de recherche d'images basées sur l'analyse du contenu sont particulièrement utiles dans de nombreuses applications. Par exemple, dans le domaine médical, les médecins peuvent les utiliser pour confronter les images médicales des patients ayant des symptômes similaires pour faciliter l'établissement d'un diagnostic [92]. Dans le domaine juridique, les organismes chargés de l'application des lois peuvent également s'en servir pour comparer les éléments de preuves provenant de la scène du crime avec des enregistrements présents dans leurs archives [59]. Lu *et al.* ont été les premiers à proposer un système de recherche d'images dans le domaine chiffré basé sur l'analyse du contenu [79]. Ainsi, les auteurs permettent de préserver la confidentialité visuelle des données originales. Pour cela, ils utilisent l'indice inversé et le min-Hash sécurisés. Dans un autre travail, les mêmes auteurs ont étudié trois techniques de protection des caractéristiques d'une image, à savoir rendre aléatoires leurs plans binaires, les projeter ou les coder aléatoirement [80]. Ils montrent ainsi que les caractéristiques chiffrées en rendant aléatoires leurs plans binaires ou en effectuant un codage aléatoire peuvent être utilisées pour calculer une distance de Hamming dans le domaine chiffré. Par ailleurs, les caractéristiques chiffrées en effectuant

une projection aléatoire peuvent être utilisées pour calculer une distance L1. Hsu *et al.* proposent une méthode basée sur le chiffrement homomorphe où l'utilisation de la transformation de caractéristiques visuelles invariante à l'échelle (*Scale-Invariant Feature Transform, SIFT*) pour extraire les caractéristiques de l'image ne remet pas en cause la confidentialité de son contenu original [52]. Ferreira *et al.* ont décrit une nouvelle méthode de chiffrement multimédia permettant la recherche d'images dans le domaine chiffré [36]. Les informations de texture et de couleur sont séparées pour être chiffrées indépendamment. Le chiffrement des informations de texture est réalisé en utilisant un cryptosystème probabiliste pour protéger le contenu de l'image, tandis que celui des informations de couleur est fait par un cryptosystème déterministe pour permettre une recherche basée sur les caractéristiques de couleur. Enfin, Xia *et al.* ont récemment constaté qu'aucune des méthodes de l'état-de-l'art ne considérait le fait que les utilisateurs pouvaient effectuer des requêtes malhonnêtes et distribuer illégalement les images récupérées [166]. Les auteurs proposent alors une solution à ce problème via un protocole basé sur l'insertion de données cachées.

3.2.3 Insertion de données cachées dans des images chiffrées

Les méthodes d'insertion de données cachées dans le domaine chiffré (IDCDC) sont employées pour dissimuler des données cachées dans le domaine chiffré, sans connaître la clé utilisée lors du chiffrement des données multimédia, ni leur contenu original en clair. Elles sont principalement utilisées pour annoter des données multimédia ou pour permettre leur authentification. Les données cachées insérées peuvent ainsi être un label, des données d'horodatage, ou encore des informations sur l'origine des données, telles que des données EXIF pour les images. Dans ce type d'approches, le propriétaire des données multimédia et le propriétaire du message caché peuvent être deux personnes distinctes. Par exemple, dans le cas du stockage d'images sur une plateforme *cloud*, le chiffrement est réalisé par le propriétaire de l'image pour la protéger, puis l'image chiffrée résultante est stockée sur le serveur *cloud*. Le gestionnaire du serveur n'a donc pas accès au contenu original de l'image. Néanmoins, il peut tout de même dissimuler des données cachées directement dans le domaine chiffré. Dans cette application précise, les labels insérés dans les images chiffrées peuvent permettre une meilleure gestion du serveur *cloud* pour les administrateurs [117] [1]. Lors de la phase de décodage, les données cachées doivent pouvoir être extraites sans erreur. De plus, si un utilisateur autorisé télécharge l'image chiffrée marquée depuis le serveur *cloud*, il doit pouvoir retrouver sans perte le contenu original de l'image après l'opération de déchiffrement. Ainsi, les méthodes d'IDCDC fournissent un moyen alternatif aux systèmes traditionnels de gestion de fichiers en dissimulant des informations supplémentaires à l'intérieur des données multimédia chiffrées elles-mêmes, au lieu d'utiliser un fichier auxiliaire de métadonnées. À cet effet, de nombreuses méthodes ont été proposées depuis 2008, afin d'obtenir le meilleur compromis entre la charge utile (c'est-à-dire la quantité de données insérées), le nombre de bits de données cachées extraits de manière erronée et la qualité des données multimédia reconstruites par rapport aux données originales attendues.

Dans la suite de ce chapitre, nous nous intéressons aux méthodes d'IDCDC et en

1. Notons que ces opérations doivent s'effectuer dans un cadre sécurisé. En particulier, l'utilisateur doit connaître les traitements réalisés et avoir donné son accord.

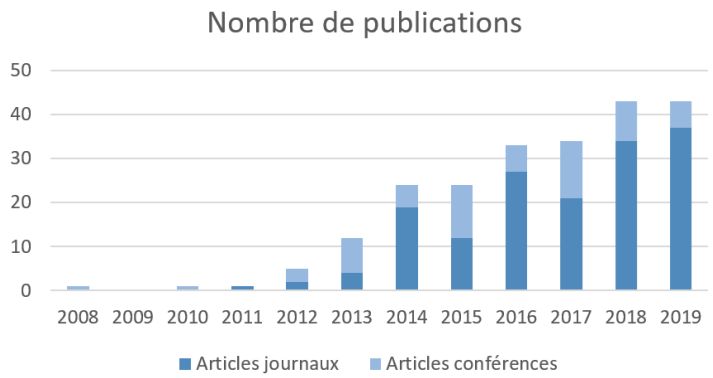


FIGURE 3.2 – Historique des publications en insertion de données cachées dans le domaine chiffré (de 2008 à 2019).

particulier celles appliquées aux images numériques.

3.3 Motivations

L'insertion de données cachées est une approche permettant de dissimuler des données secrètes dans un signal (*e.g.* une image). Après leur extraction, il est primordial de pouvoir reconstruire l'image initiale sans que celle-ci soit altérée. Par ailleurs, pour des raisons de confidentialité, il est parfois nécessaire de vouloir rendre illisible une image. Pour cela, il existe de nombreuses méthodes de chiffrement. En outre, l'insertion de données cachées et le chiffrement peuvent être réalisés conjointement. En 2008, Puech *et al.* ont décrit une des toutes premières méthodes d'insertion de données cachées dans le domaine chiffré [100]. Depuis, de nombreuses méthodes ont été développées. En effet, comme illustré en figure 3.2, le nombre de publications en 2019 est 40 fois supérieur à celui de 2009. Les applications visées par l'insertion de données cachées dans le domaine chiffré sont principalement :

— **Gestion des droits numériques** (protocole acheteur-vendeur) : Avant de vendre ses données multimédia, le distributeur les chiffre pour garantir leur confidentialité visuelle. De plus, pour empêcher leur distribution illégale, il les marque d'un filigrane contenant des informations sur l'acheteur. Dès lors, lorsque ses données multimédia sont achetées, chaque copie vendue est unique. Cela est particulièrement utile à des fins de suivi et de traçage pour pouvoir identifier un client lorsqu'une copie suspecte est trouvée. Cependant, puisque le distributeur connaît le filigrane, un client malhonnête peut prétendre que des copies illégales sont distribuées par le distributeur lui-même. Inversement, il est aussi possible pour le distributeur de faire passer un client honnête pour malhonnête en insérant son filigrane dans les données multimédia et en en distribuant des copies. Afin d'empêcher cela, des solutions ont été proposées pour que le distributeur ne connaisse pas la forme finale du filigrane [96].

— **Stockage sur le** *cloud* : Dans la perspective de protéger sa vie privée, un utilisateur chiffre ses données multimédia personnelles avant de les stocker sur une plateforme de *cloud*. Pour faciliter la gestion du serveur, l'administrateur de la

plateforme de *cloud* insère des données nécessaires à l'indexation par exemple, directement dans le domaine chiffré [119].

— **Préservation de la vie privée des patients** (monde médical) : Dans les hôpitaux, les images médicales des patients sont chiffrées pour préserver leur caractère confidentiel [7]. Par ailleurs, des informations sur les patients sont insérées dans les images chiffrées résultantes pour permettre de facilement les identifier. L'infirmière ou le secrétariat qui les manipule peut extraire ces informations sans rien déchiffrer. D'autre part, le médecin peut avoir accès à la fois aux informations d'identification du patient ainsi qu'aux images en clair.

— **Données classées** (monde militaire) : Dans le monde militaire, un officier de grade inférieur (un lieutenant par exemple) peut extraire le label inséré dans des données multimédia chiffrées pour les administrer (les copier, les archiver, les déplacer, *etc.*) sans avoir accès à leur contenu en clair. Par ailleurs, un officier d'un grade supérieur (un général par exemple) peut déchiffrer les données chiffrées, ce qui lui permet d'accéder à la fois au label inséré et aux données en clair originales [8].

— **Journalisme** : Un journaliste chiffre des données multimédia avant de le transmettre à l'entreprise pour laquelle il travaille afin que seules les personnes autorisées puissent avoir accès au contenu. Ainsi, cela assure une couverture exclusive de l'incident ou de l'événement et évite que les entreprises concurrentes en soient informées. Par ailleurs, des informations telles que l'identifiant de l'expéditeur et la position GPS du rapport de terrain peuvent être insérées à des fins d'authentification [160] pour éviter la contrefaçon de contenu.

— **Vidéosurveillance** : L'enregistrement vidéo de la caméra de surveillance est chiffré de manière sélective (masquage de la région d'intérêt telle qu'un visage par exemple) pour éviter toute atteinte à la vie privée [30]. Par ailleurs, des informations concernant l'acquisition des images (ID de la caméra, heure, date, *etc.*) doivent être présentes dans les images enregistrées à des fins d'authentification, en particulier lorsque le flux vidéo sert de preuve lors d'une audience.

— **Analyse de données** : De nos jours, les données sont massivement générées et collectées [50]. Ces données doivent être étiquetées mais leur confidentialité doit être également assurée.

3.4 Différentes classes et caractéristiques

Comme nous l'avons présenté en section 3.3, de nombreuses méthodes d'IDCDC appliquées aux images numériques ont été développées ces dernières années. De par leur diversité, plusieurs classes et caractéristiques peuvent être définies. Dans cette section, nous commençons par définir les différentes propriétés permettant de catégoriser les méthodes de l'état-de-l'art en section 3.4.1. Nous décrivons alors les approches classiques pour le chiffrement des images dans les méthodes d'IDCDC en section 3.4.2. Enfin, en section 3.4.3, nous détaillons les critères utilisés pour l'évaluation de leurs performances.

3.4.1 Propriétés

Les propriétés permettant de définir et de catégoriser les méthodes d'IDCDC sont détaillées dans cette section, à savoir : la notion de compromis entre la capacité d'insertion et la qualité de l'image reconstruite ainsi que les approches pouvant être employées pour la phase d'encodage et celle de décodage.

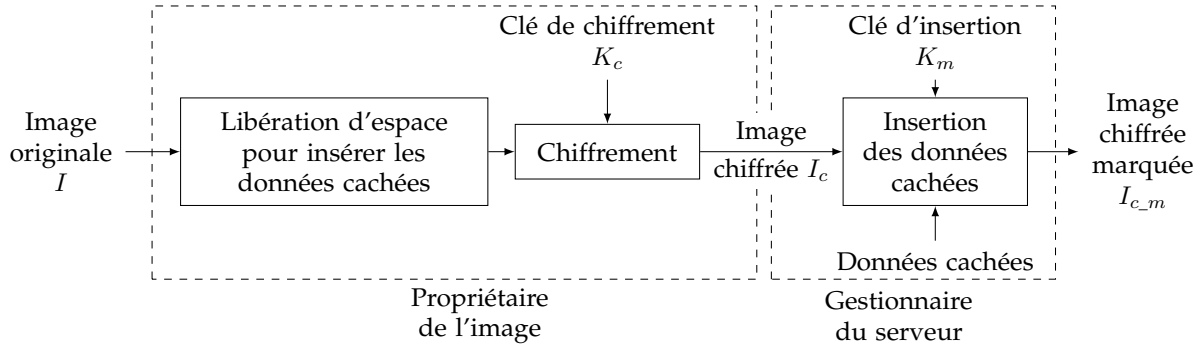
Compromis capacité et qualité

Dans les méthodes d'IDCDC, il existe un réel compromis entre le nombre de bits secrets insérés dans l'image chiffrée et la qualité de l'image reconstruite après déchiffrement de l'image chiffrée marquée. Ainsi, nous distinguons deux types de méthodes suivant la quantité de bits de données cachées pouvant être insérée. Les méthodes d'IDCDC avec une faible capacité sont celles dont la charge utile est inférieure à $0,5$ bpp. A l'inverse, une méthode est dite à haute capacité quand la charge utile est proche ou supérieure à 1 bpp. Notons que, jusqu'en 2018, aucune des méthodes de l'état de l'art ne permettait d'insérer plus d'un bit par pixel. Par ailleurs, après le déchiffrement de l'image chiffrée marquée, certaines méthodes permettent de conserver les données cachées dans l'image déchiffrée en clair. Dans ce cas, l'image reconstruite doit être similaire à l'image originale. De plus, une méthode est dite totalement réversible lorsque l'image originale peut être reconstruite sans perte après l'extraction des données cachées. Finalement, notons que plus la quantité de bits des données cachées à insérer est importante, plus l'image reconstruite lors de la phase de décodage risque d'être dégradée.

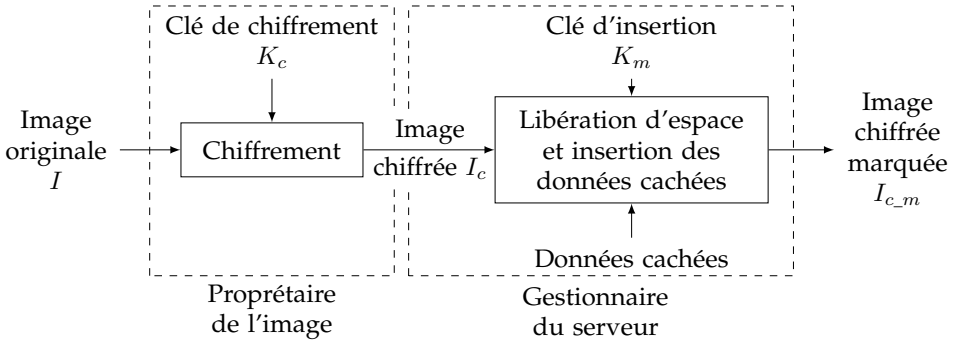
Deux approches pour l'encodage

Durant l'encodage, les méthodes d'IDCDC peuvent être divisées en deux catégories, à savoir les approches basées sur la libération de l'espace pour l'insertion des données cachées avant le chiffrement (LEIC, Libération d'Espace pour l'Insertion avant Chiffrement) ou sur la libération de l'espace pour l'insertion des données cachées après le chiffrement (CLEI, Chiffrement puis Libération d'Espace pour l'Insertion), comme illustré en fig. 3.3. D'une part, dans les méthodes LEIC, l'image originale est pré-traitée par son propriétaire avant son chiffrement afin de libérer de l'espace pour dissimuler les bits des données cachées. L'image est ensuite chiffrée et une autre personne – par exemple le gestionnaire d'un serveur *cloud* – peut insérer les bits des données cachées aux positions spécifiques dédiées à cet effet [81, 8, 174, 106]. D'un autre côté, dans les méthodes CLEI, le contenu de l'image originale est chiffré directement par son propriétaire sans aucun pré-traitement. Le gestionnaire du serveur *cloud* modifie alors les données chiffrées afin de dissimuler les bits des données cachées [100, 51, 172, 173, 119]. Ces deux approches sont efficaces, mais présentent toutes les deux certaines limites. Dans les méthodes LEIC, une quantité plus importante de bits peut être insérée mais une phase de pré-traitement avant le chiffrement est nécessaire. Cela peut être un problème qui s'avère peu pratique si le propriétaire de l'image ne sait pas que l'image chiffrée doit être analysée ou traitée ultérieurement. Dans les méthodes CLEI, le destinataire de l'image chiffrée marquée doit prédire le contenu de l'image d'origine pour la reconstruire. Par conséquent, l'image récupérée est généralement une estimation de l'image d'origine.

Ainsi, une réversibilité parfaite ne peut pas être obtenue. De plus, afin de minimiser la distorsion introduite, nous ne pouvons pas insérer un grand nombre de bits secrets.



(a) Approche LEIC.



(b) Approche CLEI.

FIGURE 3.3 – Deux approches possibles pour l'encodage : a) Libérer de l'espace pour l'insertion des données cachées avant le chiffrement (LEIC, Libération d'Espace pour l'Insertion avant Chiffrement), b) Libérer de l'espace pour l'insertion des données cachées après le chiffrement (CLEI, Chiffrement puis Libération d'Espace pour l'Insertion).

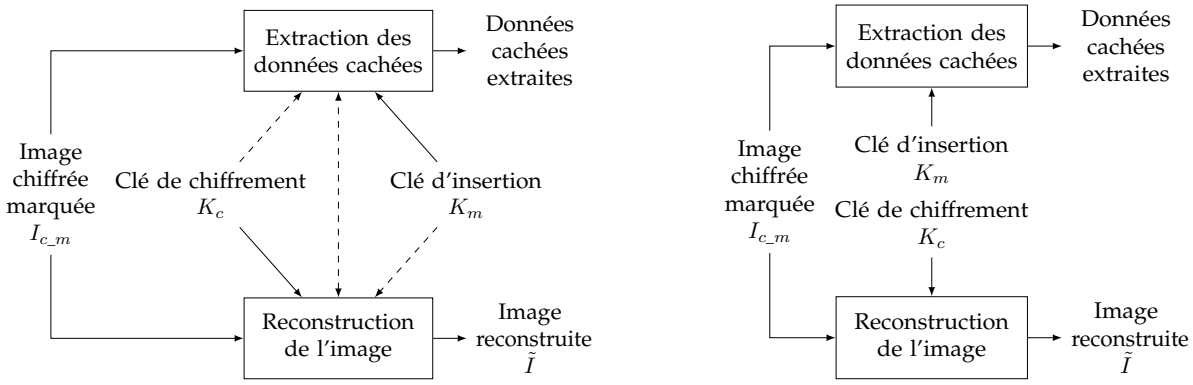
Deux approches pour le décodage

Pendant la phase de décodage, l'extraction des données cachées et la reconstruction de l'image originale peuvent être réalisées conjointement ou séparément, comme illustré en fig. 3.4. Dans le cas joint, cela signifie que l'image en clair ne peut pas être obtenue sans connaître la clé d'insertion. En effet, en utilisant seulement la clé de chiffrement, seule une version dégradée de l'image originale peut être obtenue [100]. Par ailleurs, dans certains méthodes, la connaissance seule de la clé d'insertion ne permet pas l'extraction des données cachées [172]. En revanche, dans le cas séparatif, les opérations d'extraction des données cachées et de reconstruction de l'image originale peuvent s'effectuer séparément, c'est-à-dire par deux personnes différentes. Lorsqu'un utilisateur connaît seulement la clé de chiffrement, nous distinguons deux cas :

— Une image en clair marquée par les données cachées, mais très similaire à l'image originale, peut être obtenue [173].

— L'image originale peut être parfaitement reconstruite sans qu'elle ne soit marquée par les données cachées [106].

Lorsqu'un utilisateur connaît seulement la clé d'insertion, il peut extraire les données cachées directement depuis l'image chiffrée. Notons qu'elles peuvent aussi être extraites dans le domaine clair dans le cas où la méthode utilisée permet la conservation des données cachées dans le domaine clair.



(a) Approche jointe. (b) Approche séparative.

FIGURE 3.4 – Deux approches possibles pour le décodage. Extraire les données cachées et reconstruire l'image originale : a) Conjointement, b) Séparément.

3.4.2 Approches classiquement utilisées pour le chiffrement

Les méthodes de chiffrement par flot, comme présentées en section 2.3.1, sont généralement utilisées pour protéger la confidentialité des données originales dans les méthodes d'IDCDC [172]. Par ailleurs, certaines méthodes de l'état-de-l'art exploitent la corrélation au sein des blocs de pixels d'une image dans le domaine clair [100]. De ce fait, elles utilisent un chiffrement symétrique par bloc pour conserver la structure de blocs de pixels dans l'image chiffrée. La méthode la plus utilisée est l'algorithme AES [25] décrit en section 2.2.2. Enfin, de nombreuses méthodes d'IDCDC exploitant les propriétés probabilistes et homomorphiques de certains cryptosystèmes à clé publique sont basées sur l'utilisation du cryptosystème de Paillier [91] (section 2.2.5) pour le chiffrement de l'image originale [17].

3.4.3 Critères d'évaluation

Nous évaluons une méthode d'IDCDC suivant la quantité de bits insérés en termes de charge utile, le taux de bits erronés lors de l'extraction des données cachées et la qualité visuelle après reconstruction par rapport à l'image originale. Par ailleurs, nous nous attachons également à l'évaluation du niveau de sécurité visuelle de l'image chiffrée marquée, et ce de la même façon que pour les images chiffrées (section 2.3.4).

Quantité de bits insérés

Comme pour les méthodes d'IDC dans le domaine clair, il existe une distinction entre la capacité d'insertion et la charge utile (section 1.2.3). Ainsi, la charge utile peut être nettement inférieure à la capacité d'insertion, en particulier quand une méthode

d'IDCDC nécessite d'utiliser une partie de la capacité pour insérer des informations additionnelles pour la gestion des débordements [54] ou des pixels difficiles à prédire [106] par exemple.

Taux de bits extraits erronés

Comme décrit en section 1.2.3, le taux de bits extraits erronés permet d'évaluer la qualité de la transmission des données cachées insérées dans l'image chiffrée. Dans la plupart des méthodes de l'état-de-l'art, le taux de bits erronés est nul, ce qui atteste que les données cachées sont extraites de l'image chiffrée marquée sans erreur.

Qualité visuelle

La qualité visuelle de l'image reconstruite après déchiffrement de l'image chiffrée marquée par rapport à l'image originale en clair peut être évaluée à l'aide du PSNR et du SSIM décrits en section 1.2.3. Notons que la qualité de l'image reconstruite peut être évaluée avant et après extraction des données cachées. En effet, si la méthode d'IDCDC utilisée permet de conserver les données cachées dans l'image déchiffrée, il est intéressant d'évaluer la distorsion introduite par l'insertion des données cachées dans l'image originale. Par ailleurs, notons que certaines méthodes sont irréversibles et ne permettent pas de retrouver l'image originale sans altération après extraction des données cachées.

Niveau de sécurité visuelle

Les métriques décrites dans la section 2.3.4 pour évaluer la sécurité visuelle des images chiffrées sont également utilisées dans le cas des images chiffrées marquées. Ainsi, nous considérons qu'un bon niveau de sécurité visuelle est atteint dans l'image chiffrée marquée lorsque la phase d'insertion des données cachées n'impacte pas le niveau de sécurité de la méthode de chiffrement. Par ailleurs, si les données cachées insérées sont elles-mêmes chiffrées, insérer ces données cachées dans une image chiffrée revient à dissimuler « du bruit dans du bruit ». Ainsi, les données cachées chiffrées sont *a priori* plus difficilement détectables, par stéganalyse par exemple, que dans le domaine clair.

3.5 Principales méthodes

Cette section est consacrée à la description des méthodes d'IDCDC que nous avons jugées comme étant représentatives de l'état-de-l'art actuel. Nous les avons classées selon les concepts phares sur lesquels elles s'appuient, à savoir : une partition de l'image (section 3.5.1), un décalage d'histogramme (section 3.5.2), un codage (section 3.5.3), une prédiction (section 3.5.4) ou un chiffrement à clé publique (section 3.5.5). Notons que les critères et caractéristiques définis en section 3.4 sont également discutés.

3.5.1 Partition de l'image

Dans plusieurs méthodes d'IDCDC, les pixels de l'image sont divisés en deux groupes. En effet, ceux du premier groupe sont utilisés pour effectuer l'insertion de données cachées, tandis que ceux du deuxième groupe ne sont pas marqués et servent à la reconstruction de l'image originale en clair. Ainsi, cette partition de l'image s'effectue avant ou après le chiffrement suivant la méthode utilisée. Certains auteurs réalisent cette partition de manière pseudo-aléatoire en utilisant la clé d'insertion [172]. Par ailleurs, des techniques plus astucieuses consistent à analyser les propriétés des pixels en clair [81].

Utilisation de la clé d'insertion [172]

Dans l'une des premières méthodes d'IDCDC, Zhang *et al.* ont proposé de réaliser le chiffrement de l'image originale par flot [172]. L'image chiffrée est ensuite divisée en blocs de $s \times s$ pixels, dans le but de dissimuler un bit des données cachées dans chacun d'entre eux. Dans chaque bloc, les pixels sont alors partitionnés en deux groupes S_0 et S_1 en utilisant la clé d'insertion. Si le bit des données cachées à insérer est égal à 0 (resp. 1), les trois bits de poids faible (*Least Significant Bit*, LSB) de chaque pixel dans S_0 (resp. S_1) sont inversés. Pendant la phase de décodage, l'image chiffrée marquée est déchiffrée dans le but d'obtenir une approximation de l'image originale. En effet, les cinq plans binaires les plus significatifs (*Most Significant Bit*, MSB) sont parfaitement reconstruits et seuls les bits de poids faible peuvent être altérés. Ensuite, une fonction de fluctuation est utilisée pour évaluer les irrégularités dans les groupes S_0 et S_1 des pixels reconstruits. Ainsi, les bits des données cachées peuvent être extraits et la configuration originale de chaque bloc peut être retrouvée. Cependant, bien que l'augmentation de la taille des blocs augmente les chances de reconstruire parfaitement l'image originale, la méthode de Zhang n'est pas réversible car elle n'en garantit pas une reconstruction parfaite. De plus, la charge utile est faible (inférieure à $0,1$ *bpp*) puisque seul un bit par bloc est inséré. Des améliorations à cette méthode ont de ce fait été apportées dans la littérature [51, 172].

Utilisation d'une fonction de fluctuation [81]

Ma *et al.* sont les premiers à avoir proposé une méthode LEIC, prenant ainsi à contre-pied toutes les méthodes de l'état-de-l'art développées jusqu'alors [81]. Les auteurs commencent par diviser l'image originale en blocs. Dans chaque bloc, la corrélation entre les pixels est évaluée en utilisant une fonction de fluctuation. Les blocs sont alors partitionnés en deux groupes A et B : A est composé des blocs texturés et B, de ceux relativement homogènes. Les blocs du groupe A sont placés au début de l'image et ceux de B, à la suite. Dans le but de libérer de l'espace pour insérer des données cachées, les bits des plans LSB de A sont insérés par décalage d'histogramme dans le domaine clair dans les pixels du groupe B. L'image obtenue est alors chiffrée par flot et le nombre de pixels pouvant être marqués est stocké dans les LSB des premiers pixels de A. Avec cette information, l'insertion des données cachées peut ainsi être réalisée simplement par substitution des LSB des autres pixels de A. Nous notons que les trois premiers plans LSB de chaque pixel peuvent être utilisés. La charge utile totale de

l'image chiffrée marquée peut ainsi atteindre $0,5$ bpp. Par conséquent, par rapport aux méthodes précédentes de l'état-de-l'art, la charge utile est dix fois plus grande. Enfin, lors de la phase de reconstruction, l'extraction des données cachées et la reconstruction de l'image originale peuvent s'effectuer séparément.

3.5.2 Décalage d'histogramme

De nombreuses méthodes d'IDCDC par décalage d'histogramme ont été développées du fait de leur simplicité à implémenter et de leur capacité à produire des images de très haute qualité après déchiffrement de l'image chiffrée marquée. Dans les images naturelles en clair, les pixels voisins sont très fortement corrélés. De ce fait, la distribution des différences entre ceux-ci est modélisée par une distribution laplacienne centrée en zéro. Ainsi, ces données statistiques peuvent être exploitées lors de la phase d'insertion de données cachées dans les méthodes d'IDCDC par décalage d'histogramme [54, 44, 168].

Histogramme des différences entre pixels ou des erreurs de prédiction [54]

En 2016, Huang *et al.* font le constat que les algorithmes d'insertion de données cachées dans le domaine clair précédemment proposés ne peuvent pas être appliqués dans le domaine chiffré [54]. En effet, les méthodes classiques de chiffrement ne permettent pas de conserver la corrélation entre les pixels voisins sans introduire de faille de sécurité. Dans leurs travaux, les auteurs développent une nouvelle stratégie pour le chiffrement d'image robuste à l'application des méthodes classiques d'IDC dans le domaine clair. Ainsi, l'image originale est découpée en blocs sans chevauchement. Dans chaque bloc, tous les pixels sont chiffrés en appliquant un ou-exclusif avec un même octet généré pseudo-aléatoirement. Ces blocs sont alors permutés pseudo-aléatoirement. Notons que les pixels au sein du même bloc ne sont pas permutés : seul l'ordre des blocs est modifié. Avec cette méthode de chiffrement, les propriétés statistiques de l'image en clair sont préservées. En particulier, l'histogramme des différences entre pixels (ou des erreurs de prédiction) reste inchangé. Ainsi, les algorithmes d'insertion de données cachées dans le domaine clair précédemment proposés peuvent être appliqués dans le domaine chiffré, mais la capacité d'insertion est limitée par la gestion des problèmes de débordement.

Homomorphisme et ordonnancement des valeurs des pixels [168]

Xiao *et al.* ont proposé d'adapter le concept d'ordonnancement des valeurs des pixels, défini dans le domaine clair par Li *et al* [74], dans le domaine chiffré homomorphe [168]. Les auteurs ont commencé par remarquer qu'en adaptant la méthode de chiffrement utilisée, il était possible d'obtenir un histogramme des différences entre les pixels suivant un modèle laplacien centré en zéro dans le domaine chiffré, et donc identique à celui dans le domaine clair. Après l'opération de chiffrement, l'image chiffrée est divisée en blocs de 2×2 pixels sans chevauchement. Les quatre pixels de chaque bloc sont alors réorganisés dans l'ordre croissant, notés $\{p_{c1}, p_{c2}, p_{c3}, p_{c4}\}$ avec $p_{c1} \leq p_{c2} \leq p_{c3} \leq p_{c4}$. Le pixel avec la valeur la plus élevée (p_{c4}) est sélectionné pour l'insertion de données cachées. Pour réaliser cette opération, la différence d de p_{c4} avec p_{c3} est calculée. En

fonction de la valeur de d, l'insertion d'un bit $b \in \{0, 1\}$ des données cachées est effectuée dans p_{c4} pour obtenir sa version marquée p_{c_m4} telle que :

$$p_{c\_m4} = \begin{cases} p_{c4} + b & \text{si } d = 1, \\ p_{c4} + 1 & \text{si } d > 1, \\ p_{c4} + b & \text{si } d = 0, \\ p_{c4} + 1 & \text{si } d < 0. \end{cases} \tag{3.1}$$

Notons qu'une opération similaire peut également être appliquée pour insérer un bit dans le pixel avec la valeur la plus faible p_{c1}. Dans cette méthode, l'ordre des pixels reste inchangé après l'insertion de données cachées. Du point de vue de l'histogramme, cette opération revient à étendre les bins associés aux valeurs de différences égales à 0 et à 1 et à décaler les autres bins. Cette méthode a permis de résoudre le problème d'expansion des données présent dans beaucoup de méthodes similaires dans le domaine chiffré. Néanmoins, la capacité d'insertion est limitée à $0,2\ bpp$. En effet, une carte de localisation des blocs sujets aux problèmes de débordements doit être stockée dans l'image chiffrée marquée, ce qui entraîne une forte diminution de la charge utile.

Histogramme des valeurs des pixels [44]

Récemment, Ge *et al.* ont proposé une méthode d'IDCDC par décalage d'histogramme des valeurs des pixels [44]. Au lieu d'utiliser l'histogramme des différences de pixels ou celui des erreurs de prédiction comme dans [54], les auteurs proposent de modifier directement leur valeur au sein d'un bloc. En outre, contrairement à [168], les pixels utilisés pour l'insertion ne sont pas nécessairement ceux qui ont la valeur la plus basse ou la plus élevée dans le bloc. En effet, deux pixels de référence p_{ci} et p_{cj}, avec $p_{ci} < p_{cj}$, sont sélectionnés de manière pseudo-aléatoire à l'aide d'une clé d'insertion. Tout autre pixel p_{ck} au sein du bloc est alors modifié en p_{c_mk} pour insérer un bit b des données cachées :

$$p_{c\_mk} = \begin{cases} p_{ck} - 1 & \text{si } p_{ck} < p_{ci}, \\ p_{ck} - b & \text{si } p_{ck} = p_{ci}, \\ p_{ck} & \text{si } p_{ci} < p_{ck} < p_{cj}, \\ p_{ck} + b & \text{si } p_{ck} = p_{cj}, \\ p_{ck} + 1 & \text{si } p_{ck} < p_{cj}. \end{cases} \tag{3.2}$$

Cette opération est réalisée dans tous les blocs de l'image. De plus, l'insertion de données cachées peut être réitérée plusieurs fois sur l'ensemble de l'image. Cela a pour effet d'augmenter la capacité d'insertion. Avec une seule passe sur l'image, la méthode [44] permet d'insérer la moitié de la charge utile obtenue dans l'approche [168] car la taille de la carte de localisation des blocs sujets aux problèmes de débordements est plus grande. En revanche, si plusieurs passes sont réalisées, elle permet d'atteindre une charge utile de $0,8\ bpp$, lorsqu'une dégradation de la qualité de l'image originale est tolérée.

3.5.3 Codage

Comme présenté dans certaines méthodes de l'état-de-l'art, un codage peut être appliqué aux données de l'image – avant ou après le chiffrement – dans le but d'optimiser

le nombre de bits nécessaires à leur représentation. Grâce à cette phase de compression, un gain d'espace mémoire est réalisé. Ainsi, l'espace libéré est utilisé pour insérer les bits de données cachées. Beaucoup d'algorithmes sont efficaces pour réaliser ce codage. En particulier, la méthode de Qian et Zhang, basée sur un codage source distribué [118], et celle de Cao *et al.*, basée sur un codage parcimonieux [8], ont montré des performances intéressantes.

Codage source distribué [118]

En 2016, Qian et Zhang ont proposé d'utiliser le codage source distribué dans une méthode d'IDCDC [118]. Lors de la phase d'encodage, l'image originale est d'abord chiffrée par flot. Après cela, l'image chiffrée I_c de pixels $p_c(i,j)$, avec $0 \leq i < m$ et $0 \leq j < n$, est divisée en quatre sous-images $I_c^{(k)}$ ($1 \leq k \leq 4$), dont les pixels $p_c^{(k)}(i,j)$, avec $0 \leq i < \frac{m}{2}$ et $0 \leq j < \frac{n}{2}$, sont tels que :

$$\begin{cases} p_c^{(1)}(i,j) &= p_c(2i-1, 2j-1), \\ p_c^{(2)}(i,j) &= p_c(2i-1, 2j), \\ p_c^{(3)}(i,j) &= p_c(2i, 2j-1), \\ p_c^{(4)}(i,j) &= p_c(2i, 2j). \end{cases} \tag{3.3}$$

Nous notons que le déchiffrement de chacune des sous-images $I_c^{(k)}$ résulte en l'obtention d'une miniature de l'image originale. Après l'obtention des sous-images $I_c^{(k)}$, des bits issus des trois plans MSB de $I_c^{(2)}$, $I_c^{(3)}$ et $I_c^{(4)}$ sont d'abord permutés, puis compressés avec des codes LDPC [141]. Cette compression a pour effet de générer un espace disponible pour réaliser l'insertion de données cachées. La phase de décodage est totalement séparative. La sous-image $I_c^{(1)}$ qui n'a pas été modifiée est déchiffrée puis sur-échantillonnée par interpolation bilinéaire de manière à obtenir une image de référence pour reconstruire l'image marquée en clair. Par ailleurs, pour reconstruire l'image originale en clair, les codes LDPC sont décodés par un algorithme somme-produit [78].

Codage parcimonieux [8]

Dans le but d'achever une grande charge utile, Cao *et al.* suggèrent d'utiliser le codage parcimonieux (*sparse coding*) dans leur méthode d'IDCDC [8]. La phase d'encodage LEIC est composée de trois étapes. Tout d'abord, l'image originale est divisée en patchs. Ces patchs sont alors représentés à l'aide d'un dictionnaire redondant, en utilisant un codage parcimonieux. Ensuite, les patchs les plus homogènes, avec les erreurs résiduelles les plus petites, sont sélectionnés pour effectuer l'insertion de données cachées. Pour cela, ils sont représentés par les coefficients parcimonieux. Les erreurs résiduelles, quant à elles, sont codées et dissimulées dans les patchs non sélectionnés pour l'insertion en utilisant un algorithme classique d'IDC dans les images en clair. Enfin, un chiffrement par flot est réalisé de manière à protéger les données en clair. Une fois l'image chiffrée, les bits de données cachées peuvent être insérés dans l'espace libéré précédemment. Finalement, la phase de décodage est séparative et réversible. En effet, l'image originale peut être reconstruite sans perte à l'aide des erreurs résiduelles extraites des patchs non marqués. Ainsi, les données cachées peut être retrouvées.

3.5.4 Prédiction

Dans des méthodes de l'état-de-l'art, les bits de certains pixels de l'image chiffrée sont substitués par des bits de données cachées lors de la phase d'encodage. Ainsi, leur valeur originale est perdue et doit être prédite lors de la phase de décodage afin de garantir une reconstruction de haute qualité de l'image originale en clair. Cette prédiction peut être réalisée en exploitant la différence entre un bloc de pixels en clair et sa version chiffrée [100], ou encore la forte corrélation entre un pixel et son voisinage dans le domaine clair [164].

Prédiction basée sur le calcul de l'écart-type local [100]

Dans leur article de 2008 [100], Puech *et al.* ont proposé une des premières méthodes d'IDCDC. Lors de la phase d'encodage, la méthode opte pour une approche CLEI. L'image originale est chiffrée par bloc de 16 pixels en niveaux de gris (128 bits) en utilisant l'algorithme AES en mode ECB. Un bit des données cachées est alors inséré dans chaque bloc de l'image chiffrée, ce qui correspond à une charge utile de $0,0625\ bpp$. Notons qu'une clé d'insertion est utilisée comme graine d'un générateur pseudo-aléatoire pour connaître le pixel à marquer et l'emplacement du bit à substituer par un bit des données cachées. Ainsi, l'image chiffrée marquée est obtenue lorsque tous les blocs ont été parcourus. Pendant la phase de décodage, l'extraction des données cachées est réalisée en lisant, à l'aide de la clé d'insertion, les bits des pixels qui ont été marqués. Cependant, après l'extraction, les pixels sont toujours marqués par les bits des données cachées, ce qui rend le déchiffrement de l'image difficile. Pour pallier ce problème, une analyse locale de l'écart-type dans chaque bloc est réalisée. Pour chaque bloc de l'image chiffrée marquée, grâce à la clé d'insertion, le bit marqué est localisé et remplacé par les deux valeurs possibles du bit original substitué (0 et 1). Deux configurations sont alors obtenues et déchiffrées : l'une correspond au bloc de l'image originale en clair, l'autre est erronée et a l'apparence d'un bloc totalement chiffré. L'hypothèse suivante est alors formulée : l'écart-type dans un bloc chiffré (mal déchiffré) est plus grand que celui d'un bloc en clair (correctement déchiffré). Ainsi, l'écart-type associé aux deux configurations déchiffrées est calculé. La configuration qui a la plus faible valeur d'écart-type est considérée comme étant le bloc en clair recherché. Notons que, comme la reconstruction de l'image originale implique la connaissance de la clé d'insertion, l'étape de décodage est jointe.

Prédiction par interpolation [164]

Wu et Sun ont développé une méthode d'IDCDC déclinée en deux versions : une approche jointe et une approche séparative [164]. Dans un premier temps, quelle que soit l'approche, l'image originale est chiffrée par flot. Suivant la clé d'insertion, un sous-ensemble de pixels est sélectionné pour réaliser l'insertion de données cachées. Notons que les voisins des pixels sélectionnés servent à leur prédiction lors de la phase décodage. Dans l'approche jointe, pour insérer un bit de données cachées, les LSB des pixels sélectionnés sont inversés si le bit des données cachées est égal à 1, sinon, ils restent inchangés. Lors de la phase de décodage, les pixels voisins non marqués sont interpolés pour prédire la valeur originale de chaque pixel marqué ainsi que la valeur

du bit inséré. Dans l'approche séparative, les LSB des pixels sélectionnés sont substitués par la valeur d'un bit des données cachées. Pour reconstruire une approximation de l'image originale lors du décodage, un filtre médian est alors utilisé. Par la suite, des améliorations ont été apportées à ces deux approches [28, 29].

3.5.5 Chiffrement à clé publique

Les méthodes d'IDCDC exploitant les propriétés homomorphiques des cryptosystèmes à clé publique [37] peuvent être classées en deux catégories suivant l'approche de chiffrement utilisée. En effet, on distingue les méthodes dont le chiffrement est basé sur le cryptosystème de Paillier [17] de celles utilisant un chiffrement post-quantique [65].

Méthodes basées sur l'utilisation du cryptosystème de Paillier

En 2014, Chen *et al.* ont proposé la première méthode d'IDCDC basée sur l'utilisation du cryptosystème de Paillier [17]. Chaque pixel de l'image originale est divisé en deux parties distinctes : un entier pair, composé de ses sept MSB, et de son LSB. Chaque partie est alors chiffrée indépendamment et un bit des données cachées est inséré dans chaque paire de pixels voisins. Pendant le décodage, en comparant toutes les paires de pixels déchiffrés, le receveur peut reconstruire l'intégralité des données cachées et l'image originale en clair. Le principal défaut de cette méthode est le fait de ne pas gérer les problèmes de débordement. Shiu *et al.* ont alors proposé une solution à cet inconvénient [138]. Les auteurs ont suggéré d'appliquer le concept d'expansion de la différence au domaine chiffré homomorphe. Nous notons que ces deux méthodes sont basées sur une approche LEIC. Des méthodes CLEI basées sur l'utilisation du cryptosystème de Paillier ont ensuite été proposées par Wu *et al.* [162] et Zhang *et al.* [174].

Méthodes basées sur l'utilisation du chiffrement post-quantique

La première méthode d'IDCDC fondée sur l'utilisation d'une méthode de chiffrement post-quantique a été proposée en 2016 par Ke *et al.* [65]. En effet, les auteurs ont expliqué qu'utiliser un algorithme basé sur le problème de la résolution de systèmes linéaires d'équations avec erreurs LWE (*Learning With Errors*) permettait d'obtenir à la fois un niveau élevé de sécurité, une implémentation simple et rapide et une redondance contrôlable pour l'insertion de données cachées. Ils ont alors fixé les paramètres du chiffrement et décrit leur approche d'IDCDC multi-niveaux. Cette dernière est basée sur le recodage de la redondance dans le domaine chiffré à l'aide d'opérations homomorphiques. Le principal inconvénient de cette approche est le fait de ne pas être totalement séparative.

3.6 Comparaison et discussion

Le tableau 3.1 présente un comparatif entre les différentes méthodes de l'état-de-l'art décrites dans la section 3.5. Ainsi, les méthodes sont classées par année (de 2008 à 2019), en fonction de l'approche utilisée lors de l'encodage (LEIC ou CLEI), du type de décodage (joint ou séparatif), de leur caractère réversible et de la charge utile

Année	Méthode	Encodage	Décodage	Réversibilité	Charge utile
2008	Puech *et al.* [100]	CLEI	Joint	Non	$< 0,1$ *bpp*
2011	Zhang [172]	CLEI	Joint	Non	$< 0,1$ *bpp*
2012	Hong *et al.* [51]	CLEI	Joint	Non	$< 0,1$ *bpp*
	Zhang [173]	CLEI	Séparatif	Non	$< 0,1$ *bpp*
2013	Ma *et al.* [81]	LEIC	Séparatif	Oui	$< 0,5$ *bpp*
2014	Chen *et al.* [17]	LEIC	Joint	Non	$< 0,001$ *bpb**
	Wu et Sun [164] (1)	LEIC	Joint	Non	$< 0,5$ *bpp*
	Wu et Sun [164] (2)	LEIC	Séparatif	Non	$< 0,5$ *bpp*
2015	Shiu *et al.* [138]	LEIC	Joint	Oui	$< 0,001$ *bpb**
2016	Cao *et al.* [8]	LEIC	Séparatif	Oui	< 1 *bpp*
	Huang *et al.* [54]	LEIC	Séparatif	Oui	$< 0,1$ *bpp*
	Wu *et al.* [162] (1)	CLEI	Séparatif	Non	$< 0,5$ *bpb**
	Wu *et al.* [162] (2)	CLEI	Joint	Oui	$< 0,01$ *bpb**
	Qian et Zhang [118]	CLEI	Séparatif	Non	$< 0,5$ *bpp*
	Zhang *et al.* [174] (1)	CLEI	Joint	Oui	$< 0,001$ *bpb**
	Zhang *et al.* [174] (2)	LEIC	Séparatif	Non	$< 0,001$ *bpb**
	Ke *et al.* [65]	CLEI	Joint	Oui	$< 0,5$ *bpb**
2017	Xiao *et al.* [168]	LEIC	Séparatif	Oui	$< 0,5$ *bpp*
	Dragoi *et al.* [28] (1)	LEIC	Joint	Oui	$< 0,1$ *bpp*
	Dragoi *et al.* [28] (2)	LEIC	Séparatif	Oui	$< 0,1$ *bpp*
2018	Dragoi et Coltuc [29] (1)	LEIC	Joint	Oui	$< 0,1$ *bpp*
	Dragoi et Coltuc [29] (2)	LEIC	Séparatif	Oui	$< 0,1$ *bpp*
2019	Ge *et al.* [44]	CLEI	Séparatif	Oui	< 1 *bpp*

TABLE 3.1 – Tableau comparatif des méthodes significatives de l'état-de-l'art en fonction de l'approche utilisée lors de l' encodage (LEIC, Libération d'Espace pour l'Insertion avant Chiffrement ou CLEI, Chiffrement puis Libération d'Espace pour l'Insertion), du type de décodage (joint ou séparatif), de leur caractère réversible (au sens strict du terme, *i.e* PSNR$\rightarrow +\infty$), et de la charge utile (en *bpp*, bits-par-pixel, ou en *bpb*, bits-par-bit, pour les méthodes signalées par un *).

obtenue. Tout d'abord, nous remarquons que les premières méthodes de l'état-de-l'art avaient toutes les mêmes caractéristiques, à savoir une approche CLEI lors de la phase d'encodage, un décodage joint, une impossibilité de reconstruire l'image originale sans erreur et une charge utile très faible ($< 0,1$ *bpp*). La première méthode séparative a été décrite par Zhang en 2012 [173]. Nous notons que cette propriété confère une application plus concrète aux méthodes d'IDCDC. Après 2013 et à la suite de la méthode de Ma *et al.* [81], de plus en plus de méthodes ont été basées sur une approche CLEI lors de l'encodage. Cela a permis d'atteindre une valeur de la charge utile plus élevée ($> 0,1$ *bpp*), mais restant relativement basse ($< 0,5$ *bpp*). Au fil des années, beaucoup de méthodes ont permis d'atteindre une réversibilité complète lors de la phase de reconstruction de l'image originale. En revanche, aucune ne permet d'obtenir une haute capacité, *i.e.* une charge utile proche ou supérieure à un bit par pixel. Par ailleurs, nous remarquons que celles basées sur un chiffrement à clé publique, signalées dans le tableau par un astérisque ($*$), ont une charge utile exprimée non pas en bits-par-pixel (*bpp*), mais en bits-par-bit de l'image chiffrée (*bpb*). En effet, il est important de noter que l'utilisation des cryptosystèmes à clé publique de Paillier ou LWE implique une augmentation de la

taille de l'image après son chiffrement. Selon la méthode utilisée, si les pixels de l'image originale sont codés sur 8 bits, ils peuvent correspondre à près de 2048 bits dans le domaine chiffré comme l'ont montré Ke *et al.* [66]. Ainsi, une comparaison de la charge utile exprimée en *bpp* obtenue par la méthode dans l'image chiffrée marquée n'est pas adaptée et entraînerait une mauvaise interprétation des résultats.

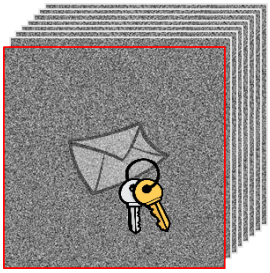
3.7 Conclusion

Dans ce chapitre, nous avons présenté la problématique d'insertion de données cachées dans les images chiffrées. Après une brève description des différentes applications à l'analyse et au traitement dans le domaine chiffré, nous avons détaillé les motivations et le contexte applicatif des méthodes d'IDCDC. Ensuite, les classes et caractéristiques de ces méthodes ont été présentées. L'état-de-l'art a alors été exposé et, à l'aide d'un tableau comparatif des méthodes, nous avons pu constater qu'aucune d'entre elles ne permettait d'obtenir à la fois une grande charge utile et très bonne qualité visuelle de l'image reconstruite par rapport à l'image originale.

Dans nos travaux de recherche, nous avons proposé des solutions à ce problème basées sur la prédiction des MSB. Les approches développées sont décrites en détail dans les chapitres 4 et 5. Par ailleurs, nous nous sommes exclusivement intéressés aux méthodes d'IDCDC appliquées aux images numériques non-compressées. Ainsi, il est important de noter que des extensions ont été proposées pour les images JPEG [119] (dont une piste de recherche est détaillée dans les perspectives de cette thèse), les vidéos [169], ou encore les objets 3D [61].

Deuxième partie

Contributions



CHAPITRE 4

Insertion de données cachées dans les images chiffrées par substitution des bits de poids fort

Sommaire

4.1 Introduction

Dans ce chapitre, nous proposons une nouvelle méthode réversible d'insertion de données cachées dans les images chiffrées (IDCDC) basée sur la prédiction des bits de poids fort (MSB). Nous présentons deux approches différentes, à savoir : l'approche d'insertion de données cachées haute capacité avec correction des erreurs de prédiction (IDCHC-CEP) et l'approche d'insertion de données cachées haute capacité avec signalisation des erreurs de prédiction (IDCHC-SEP). Avec cette méthode, les résultats obtenus en termes de qualité d'image reconstruite et de charge utile sont meilleurs que ceux des méthodes de l'état-de-l'art décrites dans le chapitre 3, quelle que soit l'approche utilisée.

La suite de ce chapitre est organisée comme suit. La section 4.2 explique brièvement les méthodes de l'état-de-l'art précédentes. La méthode d'IDCDC proposée est développée en détail dans la section 4.3. Les résultats expérimentaux réalisés sont ensuite décrits en section 4.4. Enfin, la section 4.5 conclut ce chapitre et présente des travaux futurs.

4.2 IDCDC avant 2018

Comme décrit dans le chapitre 3, les méthodes d'IDCDC sont principalement utilisées pour l'enrichissement de données et l'authentification dans le domaine chiffré via l'insertion d'un message secret. Pendant la phase de décodage, l'extraction de ce message doit pouvoir être réalisée sans erreur et l'image originale doit pouvoir être parfaitement reconstruite. Les méthodes de l'état-de-l'art peuvent être classées selon deux groupes, suivant si l'espace pour réaliser l'IDC est libéré avant le chiffrement de l'image [81] ou après [173]. De plus, le chiffrement et l'IDC peuvent s'effectuer conjointement [100] ou séparément [173, 162]. Jusqu'en 2018, aucune méthode de l'état-de-l'art n'était à « haute capacité ». En effet, leur charge utile était généralement comprise entre $0,01$ *bpp* et $0,5$ *bpp* : elle n'était jamais supérieure ou égale à 1 *bpp*.

Wu et Sun ont décrit une méthode déclinée en deux approches [164]. Dans l'approche jointe, un sous-ensemble de pixels de l'image chiffrée est sélectionné et l'espace nécessaire pour l'IDC est libéré par décalage d'histogramme. Dans l'approche séparative, une substitution des MSB est réalisée. Pendant la phase de décodage, comme les valeurs des MSB sont perdues, un filtre médian est appliqué à l'image reconstruite pour supprimer les artéfacts visuels. Cao *et al.* ont proposé une technique d'IDCDC basée sur le codage éparse [8]. Ainsi, ils réussissent à atteindre une valeur intéressante de la charge utile (environ 1 *bpp*) en exploitant la corrélation entre les pixels. Zhang *et al.* ont utilisé la cryptographie à clé publique pour le chiffrement d'images [174]. L'IDC est réalisée par substitution des bits de poids faible (LSB) dans le domaine chiffré. Comme cette opération n'altère pas de façon significative l'image chiffrée, le message secret peut être extrait sans erreur et l'image originale est parfaitement reconstruite.

Aucune des méthodes présentées précédemment ne parvient à combiner une grande charge utile (supérieure ou égale à 1 *bpp*) et une haute qualité visuelle (supérieure à 50 *dB*). Dans la plupart des cas, les valeurs des LSB sont remplacées pour réaliser l'insertion des bits d'un message secret. Cependant, lorsqu'une image est chiffrée, il est

difficile de détecter si elle contient un message secret ou non. En effet, les valeurs des pixels d'une image chiffrée sont générées pseudo-aléatoirement. Ainsi, la corrélation entre un pixel et ses voisins est très basse. Pour cette raison, nous avons proposé d'utiliser les valeurs des MSB à la place des valeurs des LSB pour insérer le message secret. Avec cette méthode, les approches précédentes de l'état-de-l'art sont ainsi prises à contre-pied. Nous notons que, dans le domaine chiffré, la confidentialité reste la même et que, durant la phase de décodage, la prédiction des valeurs des MSB est plus simple que celle des valeurs des LSB.

4.3 Méthode proposée

Dans cette section, nous commençons par introduire le schéma général de la méthode d'IDCDC (section 4.3.1). Contrairement à ce qui est fait dans les méthodes précédentes, l'IDC est réalisée par substitution des valeurs des MSB par les bits d'un message secret. Comme les valeurs des MSB remplacés sont perdues durant la phase d'insertion du message, il est nécessaire de pouvoir les prédire sans erreur pendant la phase de décodage. Dans un second temps, nous présentons en détail deux approches possibles selon la contrainte désirée, à savoir une réversibilité stricte (PSNR $\rightarrow +\infty$) ou une charge utile maximale (1 bpp). La première approche, qui n'est pas parfaitement réversible (section 4.3.2), mais qui permet d'insérer un bit du message secret par pixel, est appelée IDCHC-CEP (insertion de données cachées haute capacité avec correction des erreurs de prédiction). La seconde approche, où l'image originale est parfaitement reconstruite (section 4.3.3), mais où les données cachées insérées comprennent le message secret mais aussi des informations nécessaires à la signalisation des erreurs de prédiction, est appelée IDCHC-SEP (insertion de données cachées haute capacité avec signalisation des erreurs de prédiction).

4.3.1 Description générale de la méthode

Encodage

La phase d'encodage comprend trois étapes : la détection des erreurs de prédiction (EP) des MSB, la prise en compte des EP combinée au chiffrement de l'image, et l'insertion du message par substitution des MSB. Un schéma général de la méthode d'encodage est proposé en fig. 4.1.

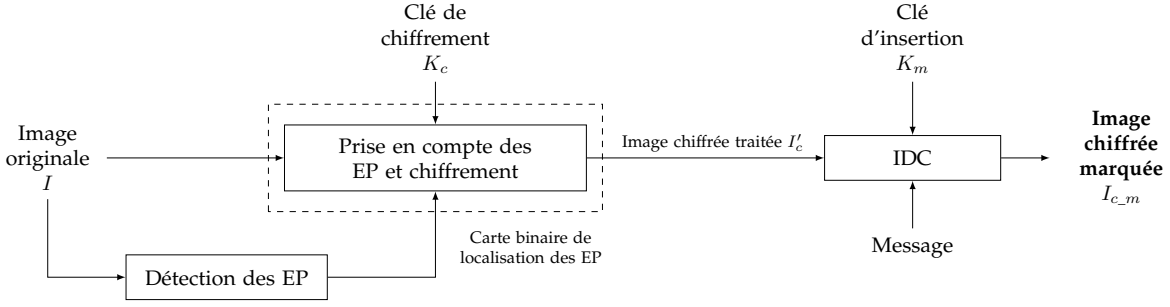


FIGURE 4.1 – Schéma général de la méthode d'encodage.

Dans cette méthode, puisque le message secret est inséré par substitution des MSB, les valeurs originales des MSB sont perdues après l'étape d'IDC. Ainsi, il est important de pouvoir prédire ces valeurs sans erreur durant la phase de décodage. En effet, dans le but de retrouver l'image originale, les pixels précédemment reconstruits sont utilisés pour prédire la valeur du pixel courant. De ce fait, la première étape consiste à analyser le contenu de l'image originale pour détecter toutes les EP :

— Nous considérons le pixel courant $p(i,j)$, avec $0 \leq i < m$ et $0 \leq j < n$, et sa valeur inverse, $inv(i,j) = (p(i,j)+128) \mod 256$. Comme il existe une différence de 128 entre ces deux valeurs, alors la valeur inverse correspond à la valeur originale de $p(i,j)$, mais dont le MSB a été inversé.

— En utilisant les valeurs des voisins de $p(i,j)$, nous calculons la valeur $pred(i,j)$, considérée comme prédicteur lors de la phase de décodage.

— Nous calculons alors la différence absolue entre $pred(i,j)$ et $p(i,j)$ et entre $pred(i,j)$ et $inv(i,j)$. Les valeurs ainsi calculées sont notées Δ et Δ^{inv} et sont telles que :

$$\begin{cases} \Delta = |pred(i,j) - p(i,j)| \\ \Delta^{inv} = |pred(i,j) - inv(i,j)| \,. \end{cases} \tag{4.1}$$

— Nous comparons ensuite les valeurs de Δ et Δ^{inv}. Si $\Delta < \Delta^{inv}$, alors il n'existe pas d'EP car la valeur originale de $p(i,j)$ est plus proche de son prédicteur que sa valeur inverse. Dans le cas contraire, une EP est identifiée et son emplacement est renseigné dans une carte binaire de localisation des EP, comme illustré en fig. 4.1.

Suivant l'approche utilisée, la carte binaire de localisation des EP est ensuite utilisée de deux façons différentes. Un pré-traitement de l'image originale pour corriger les EP peut être réalisé ; une image I' très similaire à l'image originale est alors obtenue. Par ailleurs, au lieu de corriger les EP, il est aussi possible de signaler leur emplacement dans le domaine chiffré après le chiffrement de l'image originale.

Quelle que soit l'approche utilisée, l'image en clair est chiffrée en utilisant une méthode de chiffrement par flot, telle que décrite dans la section 3.4.2. Ainsi, les pixels chiffrés $p_c(i,j)$ sont obtenus :

$$p_c(i,j) = s(i,j) \oplus p(i,j), \tag{4.2}$$

où $s(i,j)$ est l'octet associé à $p(i,j)$ dans une séquence binaire pseudo-aléatoire et $\oplus$ l'opération de ou-exclusif. Notons que si l'approche consiste à signaler l'emplacement des EP, l'image chiffrée ainsi obtenue est modifiée.

Durant la phase d'insertion du message, en utilisant la clé d'insertion K_m, le message secret est d'abord chiffré pour éviter sa détection dans l'image chiffrée marquée. Ensuite, les pixels de l'image chiffrée sont parcourus dans l'ordre des lignes de balayage (*scanline order*, de gauche à droite, de haut en bas), et le MSB de chaque pixel disponible est substitué par un bit b_k, avec $0 \leq k < m \times n$, du message secret pour obtenir le pixel $p_{c\_m}(i,j)$ associé :

$$p_{c\_m}(i,j) = b_k \times 128 + (p_c(i,j) \mod 128). \tag{4.3}$$

Nous notons que seul le premier pixel ne peut pas être marqué car sa valeur ne peut pas être prédite : il reste donc inchangé.

Décodage

Durant la phase de décodage, comme la méthode est séparable, le message secret peut être extrait et l'image originale $\tilde{I}$ en clair peut être reconstruite séparément. $\tilde{I}$ est exactement identique à l'image originale I ou à l'image pré-traitée I' très similaire à l'image originale, selon l'approche utilisée. Pour celui qui reçoit les données, trois scénarios sont alors possibles :

1. Il connaît seulement la clé d'insertion K_m,
2. Il connaît seulement la clé de chiffrement K_c,
3. Il connaît les deux clés.

Un schéma général de la phase de décodage est présenté en fig. 4.2.

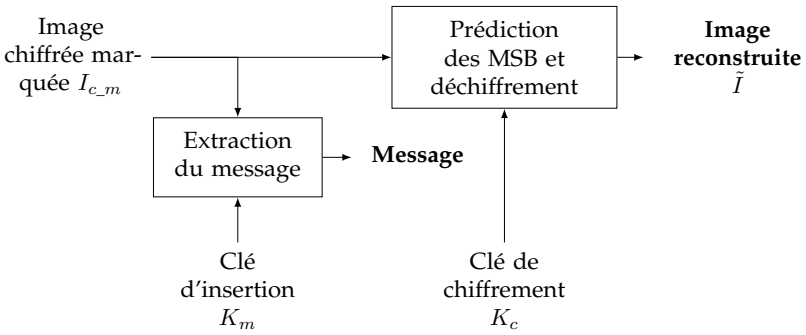


FIGURE 4.2 – Schéma général de la méthode de décodage.

Si celui qui reçoit les données connaît seulement la clé d'insertion K_m, les pixels de l'image chiffrée marquée sont parcourus dans l'ordre des lignes de balayage et le MSB de chaque pixel est extrait pour retrouver les bits du message secret chiffré :

$$b_k = p_{c\_m}(i,j)/128, \tag{4.4}$$

où $0 \leq k < m \times n$ et fait référence à l'indice du bit extrait du message.

Ensuite, en utilisant la clé d'insertion K_m, le message secret en clair correspondant est obtenu.

Dans le second scénario, si celui qui reçoit les données connaît seulement la clé de chiffrement K_c, l'image $\tilde{I}$ peut être reconstruite, telle qu'avant l'insertion du message et le chiffrement de l'image :

1. La clé de chiffrement K_c est utilisée pour générer la séquence binaire pseudo-aléatoire, composée de $m \times n$ octets $s(i,j)$.
2. Les pixels de l'image chiffrée marquée sont parcourus dans l'ordre des lignes de balayage, et pour chaque pixel, les sept LSB sont reconstruits en effectuant un ou-exclusif entre la valeur chiffrée marquée $p_{c\_m}(i,j)$ et l'octet $s(i,j)$ correspondant dans la séquence binaire pseudo-aléatoire :

$$\tilde{p}(i,j) = s(i,j) \oplus p_{c\_m}(i,j), \tag{4.5}$$

   où $\oplus$ représente l'opération de ou-exclusif.
3. La valeur du MSB est prédite comme suit :

— Avec les valeurs des pixels voisins précédemment reconstruites, la valeur du prédicteur $pred(i,j)$ est calculée.

— Les deux valeurs possibles du pixel original sont générées, avec MSB $= 0$ et MSB $= 1$. Les différences de ces deux valeurs avec $pred(i,j)$ sont calculées et notées Δ^0 et Δ^1 :

$$\begin{cases} \Delta^0 = \left| pred(i,j) - \tilde{p}(i,j)^{MSB=0} \right|, \\ \Delta^1 = \left| pred(i,j) - \tilde{p}(i,j)^{MSB=1} \right|. \end{cases} \tag{4.6}$$

— La plus petite valeur entre Δ^0 et Δ^1 renseigne sur la valeur recherchée du pixel original :

$$\tilde{p}(i,j) = \begin{cases} \tilde{p}(i,j)^{\text{MSB=0}}, & \text{si } \Delta^0 < \Delta^1, \\ \tilde{p}(i,j)^{\text{MSB=1}}, & \text{sinon.} \end{cases} \tag{4.7}$$

4.3.2 Approche IDCHC-CEP

Dans l'approche IDCHC-CEP (insertion de données cachées haute capacité avec correction des erreurs de prédiction), comme illustré en fig. 4.3, la première étape consiste à pré-traiter l'image originale pour éliminer toutes les EP et ainsi rendre possible la reconstruction haute qualité de l'image originale durant la phase de décodage. Après ce traitement, l'image pré-traitée est alors chiffrée. Pendant la phase d'IDC, chaque pixel de l'image chiffrée est marqué par un bit du message. En utilisant cette approche, une charge utile maximale d'1 bpp est atteinte.

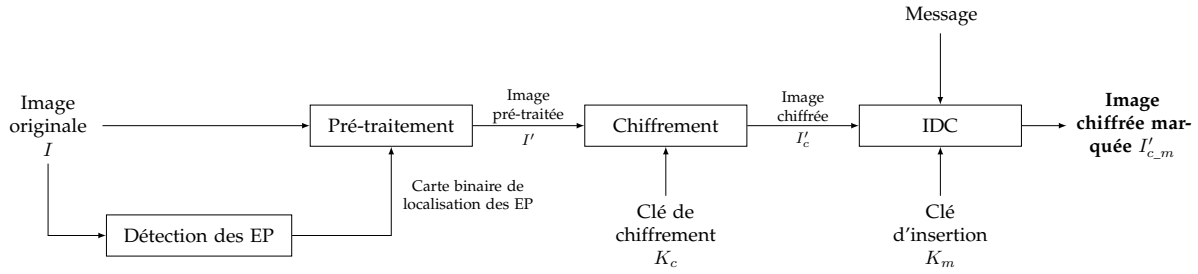


FIGURE 4.3 – Encodage avec l'approche IDCHC-CEP.

Prédicteur utilisé

Comme expliqué en section 4.3.1, les pixels précédents sont utilisés pour prédire la valeur du pixel courant. Dans cette approche, la moyenne entre le pixel de gauche et celui du dessus est considérée comme prédicteur $pred(i,j)$:

$$pred(i,j) = \frac{p(i-1,j) + p(i,j-1)}{2}. \tag{4.8}$$

Nous notons qu'un traitement spécifique est appliqué pour les pixels de la première ligne et de la première colonne.

En effet, utiliser la valeur moyenne comme prédicteur permet de minimiser la modification de la valeur du pixel courant en cas d'EP, en particulier lorsqu'il existe une grande différence entre la valeur du pixel courant et celle de l'un de ses voisins.

Pré-traitement de l'image

Après la détection des EP, l'image originale I est pré-traitée pour générer une image I' sans EP. Pour chaque pixel problématique, la valeur de l'EP est observée puis la modification minimale à apporter au pixel courant pour corriger cette EP est calculée. L'équation (4.9) indique la modification nécessaire pour éliminer toute erreur lors de la phase de décodage :

$$|pred(i,j) - p(i,j)| < 64. \tag{4.9}$$

Les étapes de pré-traitement de l'image originale pour corriger toutes les EP sont présentées dans l'algorithme 1.

Algorithme 1 : Algorithme de pré-traitement.

Données : Image originale I de taille $m \times n$ pixels
Résultat : Image pré-traitée I' de taille $m \times n$ pixels
pour $i \leftarrow 0$ **à** m **faire**
  pour $j \leftarrow 0$ **à** n **faire**
    $inv(i,j) \leftarrow (p(i,j) + 128) \bmod 256$;
    si $i = 0$ **ou** $j = 0$ **alors**
      Prédiction unidirectionnelle;
    sinon
      $pred(i,j) \leftarrow \frac{p(i-1,j)+p(i,j-1)}{2}$;
    $\Delta \leftarrow |pred(i,j) - p(i,j)|$;
    $\Delta^{inv} \leftarrow |pred(i,j) - inv(i,j)|$;
    si $\Delta \geq \Delta^{inv}$ **alors**
      si $p(i,j) < 128$ **alors**
        $p'(i,j) \leftarrow pred(i,j) - 63$;
      sinon
        $p'(i,j) \leftarrow pred(i,j) + 63$;
    sinon
      $p'(i,j) \leftarrow p(i,j)$;

Par exemple, si $p(i,j) = 50$, $p(i-1,j) = 78$ et $p(i,j-1) = 154$, alors :

$$inv(i,j) = (50 + 128) \mod 256 = 178,$$

$$pred(i,j) = \frac{78 + 154}{2} = 116.$$

Nous calculons Δ et Δ^{inv} :

$$\Delta = |116 - 50| = 66,\ \Delta^{inv} = |116 - 178| = 62.$$

Comme $\Delta \geq \Delta^{inv}$, il existe une EP et la valeur de $p(i,j)$ doit être modifiée. Pour corriger cette EP, l'inéquation suivante doit être vérifiée :

$$pred(i,j) - p(i,j) < p(i,j) + 128 - pred(i,j).$$

En développant cette expression, nous obtenons :

$$p(i,j) > pred(i,j) - 64.$$

Ainsi, la modification de $p(i,j)$ la plus faible pour minimiser la distorsion introduite est :

$$p'(i,j) = pred(i,j) - 63 = 116 - 63 = 53.$$

Ensuite, l'image pré-traitée I' est chiffrée. L'insertion du message secret est réalisée en remplaçant le MSB de chaque pixel de l'image chiffrée I'_c par un bit du message, en suivant l'équation (4.3). Finalement, l'image chiffrée marquée I'_{c_m} est obtenue avec une charge utile maximale de 1 bpp.

Extraction du message et reconstruction de l'image

Durant le décodage, pour extraire le message secret, l'image chiffrée marquée I'_{c_m} est parcourue et le MSB de chaque pixel est simplement extrait en utilisant l'équation (4.4). De plus, l'image originale pré-traitée I' peut être reconstruite sans perte. Pour cela, l'image chiffrée marquée I'_{c_m} est déchiffrée pour obtenir les sept LSB de chaque pixel (équation (4.5)) et la valeur du MSB est ensuite prédite en appliquant l'équation (4.6) et l'équation (4.7). Nous notons que l'image reconstruite est très similaire à l'image originale.

4.3.3 Approche IDCHC-SEP

Dans l'approche IDCHC-SEP (insertion de données cachées haute capacité avec signalisation des erreurs de prédiction), le but principal est d'être capable de reconstruire parfaitement l'image originale. Dans ce cas, la charge utile peut sensiblement diminuer à cause de la signalisation de l'emplacement des EP. Pour signaler les EP, le message à insérer est adapté de manière à tenir compte de la carte de localisation des EP, construite pendant la phase de détection des EP. L'image originale est alors chiffrée sans pré-traitement et ensuite, l'emplacement des EP est inséré dans l'image chiffrée. Pendant l'étape d'IDC, les bits du message secret peuvent seulement être insérés dans les pixels disponibles. A la fin du décodage, grâce à la signalisation des EP, l'image originale est reconstruite sans perte, ce qui est indiqué par une valeur du PSNR qui tend vers $+\infty$. Le schéma général de cette approche est présenté en fig. 4.4.

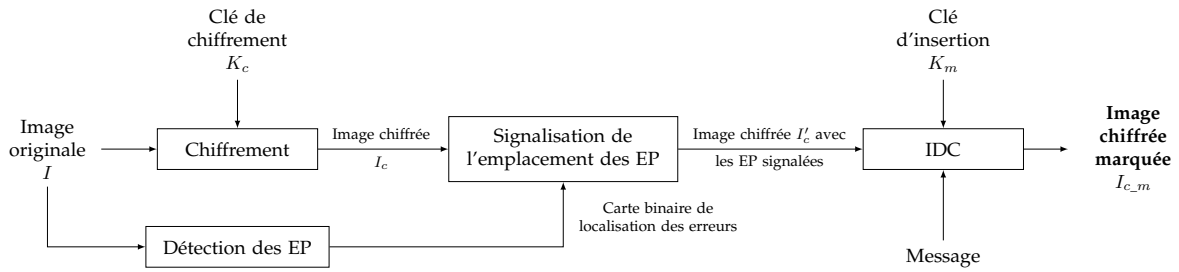


FIGURE 4.4 – Encodage avec l'approche IDCHC-SEP.

Prédicteur utilisé

Pour chaque pixel, deux pixels voisins peuvent être utilisés comme prédicteurs : celui de gauche $p(i, j-1)$ et celui du haut $p(i-1, j)$. Pour déterminer lequel des deux est considéré en tant que prédicteur, la valeur absolue de leur différence avec le pixel courant $p(i, j)$ est calculée et la valeur la plus proche est sélectionnée :

$$\begin{aligned} \text{Si} \quad & |p(i-1,j) - p(i,j)| < |p(i,j-1) - p(i,j)|, \\ \text{alors,} \quad & pred(i,j) = p(i-1,j), \\ \text{sinon,} \quad & pred(i,j) = p(i,j-1). \end{aligned} \tag{4.10}$$

Dans certains cas, l'autre valeur peut être utilisée comme prédicteur pour l'inverse du pixel $inv(i, j)$ pendant la détection des EP, mais le résultat reste identique. Nous notons que la moyenne entre les pixels à gauche et au dessus peut aussi être considérée comme prédicteur, comme pour l'approche IDCHC-CEP, mais il a été montré expérimentalement que les résultats obtenus étaient moins intéressants.

Signalisation de l'emplacement des EP

Pendant l'étape de détection des EP, l'emplacement des EP est renseigné dans une carte binaire de localisation des EP, comme expliqué en section 4.3.1. Ensuite, l'image originale I est chiffrée et, avant l'insertion du message, l'image chiffrée I_c est adaptée pour éliminer les EP. Elle est divisée en blocs de huit pixels et parcourue, bloc par bloc, dans l'ordre des lignes de balayage. Si au moins une EP est identifiée dans un bloc selon la carte binaire de localisation des EP, alors le bloc courant est encadré de deux drapeaux en remplaçant le MSB de chaque pixel des blocs précédents et suivant par des 1. Dans le bloc courant, le MSB du pixel est substitué par la valeur 1 s'il existe une EP et par la valeur 0 le cas échéant, comme illustré en fig. 4.5. Dans le cas où il n'y a pas d'EP dans le bloc courant et s'il ne sert pas de drapeau, alors les huit pixels de ce bloc sont utilisés pour l'IDC, comme décrit en section 4.3.1. Si des EP sont présentes dans deux blocs adjacents, le drapeau qui indique la fin de la séquence d'erreur est décalé jusqu'au prochain bloc sans EP. La perte en termes de charge utile est alors moins importante puisque les drapeaux sont utilisés pour plus d'une EP. Nous notons qu'il est possible de considérer des blocs de taille inférieure mais, statistiquement, le risque qu'une partie du message secret soit identifiée comme étant un drapeau augmente. Avec des blocs de huit pixels, le compromis entre la perte de charge utile et le taux de fausse alarme est raisonnable. En effet, peu de pixels ne peuvent pas être marqués par les bits du message secret et la probabilité qu'une partie du message ressemble à un drapeau est très faible ($\frac{1}{2^8}$)[1].

L'image chiffrée I'_c avec les EP signalées est alors obtenue. Grâce à ce traitement, pendant la phase d'insertion, le propriétaire du message caché peut extraire les valeurs des MSB de chaque pixel et utiliser l'information sur l'emplacement des EP pour détecter les pixels pouvant être marqués par les bits du message secret (*i.e.* dans tous les blocs où il n'y a pas d'EP et qui ne servent pas de drapeaux). Tous les pixels disponibles sont alors marqués pour obtenir l'image chiffrée marquée I_{c_m}, en utilisant l'équation (4.3).

1. Si huit bits consécutifs du message secret sont égaux à 1, ils peuvent être considérés comme un drapeau de fin de séquence. A ce jour, nous n'avons pas proposé de solution permettant de résoudre ce problème, toutefois rarement rencontré dans le domaine chiffré.

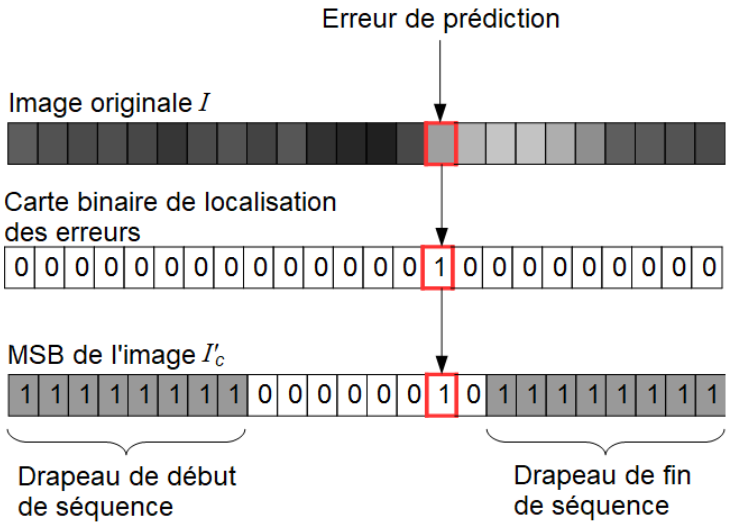


FIGURE 4.5 – Signalisation des EP.

Extraction du message et reconstruction de l'image

Pendant le décodage, le message secret peut être extrait en suivant les étapes suivantes :

— Les pixels de l'image chiffrée marquée I_{c_m} sont parcourus et, pour chaque pixel, la valeur du MSB est extraite, selon l'équation (4.4), et stockée. Les bits extraits jusqu'à la première séquence de huit bits égaux à 1 sont supposés comme étant ceux du message secret.

— Quand une telle séquence est rencontrée, cela indique le début d'une séquence comportant des EP. Puisque les pixels suivants n'ont pas été marqués pendant la phase d'insertion du message, les pixels sont parcourus jusqu'à la prochaine séquence de huit MSB égaux à 1, qui indique ainsi la fin de la séquence d'erreur.

— Ce processus est répété jusqu'à la fin de l'image.

D'autre part, comme cette méthode est totalement réversible, l'image originale I est parfaitement reconstruite. L'image chiffrée marquée I_{c_m} est d'abord déchiffrée pour retrouver les sept LSB de chaque pixel, en utilisant l'équation (4.5). Ensuite, les valeurs des MSB de chaque pixel sont prédites à l'aide de l'équation (4.6) et de l'équation (4.7).

4.4 Résultats expérimentaux

Dans cette section, nous présentons les résultats obtenus en appliquant notre méthode avec les approches IDCHC-CEP (insertion de données cachées haute capacité avec correction des erreurs de prédiction) et IDCHC-SEP (insertion de données cachées haute capacité avec signalisation des erreurs de prédiction). La section 4.4.1 détaille un exemple complet des deux approches. Ensuite, la section 4.4.2 développe les résultats obtenus sur 10000 images de la base BOWS-2 [3]. Dans la section 4.4.3, une analyse statistique est alors réalisée pour tester la confidentialité visuelle des images après l'application de notre méthode. Enfin, en section 4.4.4, nos deux approches sont comparées à deux méthodes simples, ainsi qu'aux méthodes récentes de l'état-de-l'art.

4.4.1 Exemple complet de la méthode proposée

Les deux approches ont été appliquées sur la même image originale de 512×512 pixels, illustrée en fig. 4.6 et issue de la base BOWS-2 [3].

FIGURE 4.6 – Image originale I issue de la base BOWS-2 [3].

La fig. 4.7 illustre les résultats obtenus avec l'approche IDCHC-CEP et la fig. 4.8, avec l'approche IDCHC-SEP. En fig. 4.7.a et fig. 4.8.a, l'emplacement des pixels concernés par une EP du MSB est signalé en blanc. Nous pouvons observer que, dans les deux approches, le nombre et l'emplacement des EP sont différents car les prédicteurs utilisés ne sont pas les mêmes, comme expliqué en section 4.3.2 et en section 4.3.3. Cependant, ils sont globalement du même ordre de grandeur.

Dans l'approche IDCHC-CEP (fig. 4.7.a), les MSB de certains pixels de l'image originale sont mal prédits si nous n'adaptons la valeur des pixels pendant la phase de pré-traitement. Dans l'approche IDCHC-SEP (fig. 4.8.a), les pixels concernés par une EP (en blanc) ne peuvent pas être marqués. De plus, les pixels en gris ne peuvent pas l'être non plus car ils sont utilisés comme drapeaux ou font partie d'une séquence avec une ou plusieurs EP. Nous notons que les EP sont souvent sur les contours. De plus, un même bloc peut parfois comporter plus d'une EP. Dans ce cas, la perte en terme de charge utile est moins importante. L'histogramme en fig. 4.7.b illustre la distribution des EP lorsque l'approche IDCHC-CEP est utilisée et les modifications nécessaires à effectuer sur les pixels pour supprimer ces EP. La fig. 4.7.c représente alors l'image pré-traitée en utilisant l'algorithme 1. Nous pouvons remarquer que l'image pré-traitée est très similaire à l'image originale, ce qui est indiqué par un PSNR égal à $46,87\ dB$ et un SSIM de $0,9997$. En fig. 4.7.d, nous pouvons voir l'image pré-traitée chiffrée par flot à l'aide de la clé de chiffrement. La fig. 4.8.b est l'image chiffrée obtenue avec l'approche IDCHC-SEP et la fig. 4.8.c correspond à cette image après la signalisation des EP. Le contenu de l'image originale et l'emplacement des EP restent visuellement confidentiels. La fig. 4.7.e et la fig. 4.8.d sont les images chiffrées marquées, obtenues à la fin de la phase d'encodage, après l'insertion d'un message secret. Avec l'approche IDCHC-CEP, chaque pixel de l'image pré-traitée est utilisé pour dissimuler un bit du message secret (charge utile = $1\ bpp$). Avec l'approche IDCHC-SEP, les pixels sont marqués pour signaler les EP et, même si la charge utile est moins importante, elle est tout de même haute et égale à $0,9220\ bpp$. La fig. 4.7.f et la fig. 4.8.e présentent les images reconstruites après extraction des données. La fig. 4.7.f est la même que l'image pré-traitée (PSNR = $46,87\ dB$) et, avec l'approche IDCHC-SEP, l'image originale est

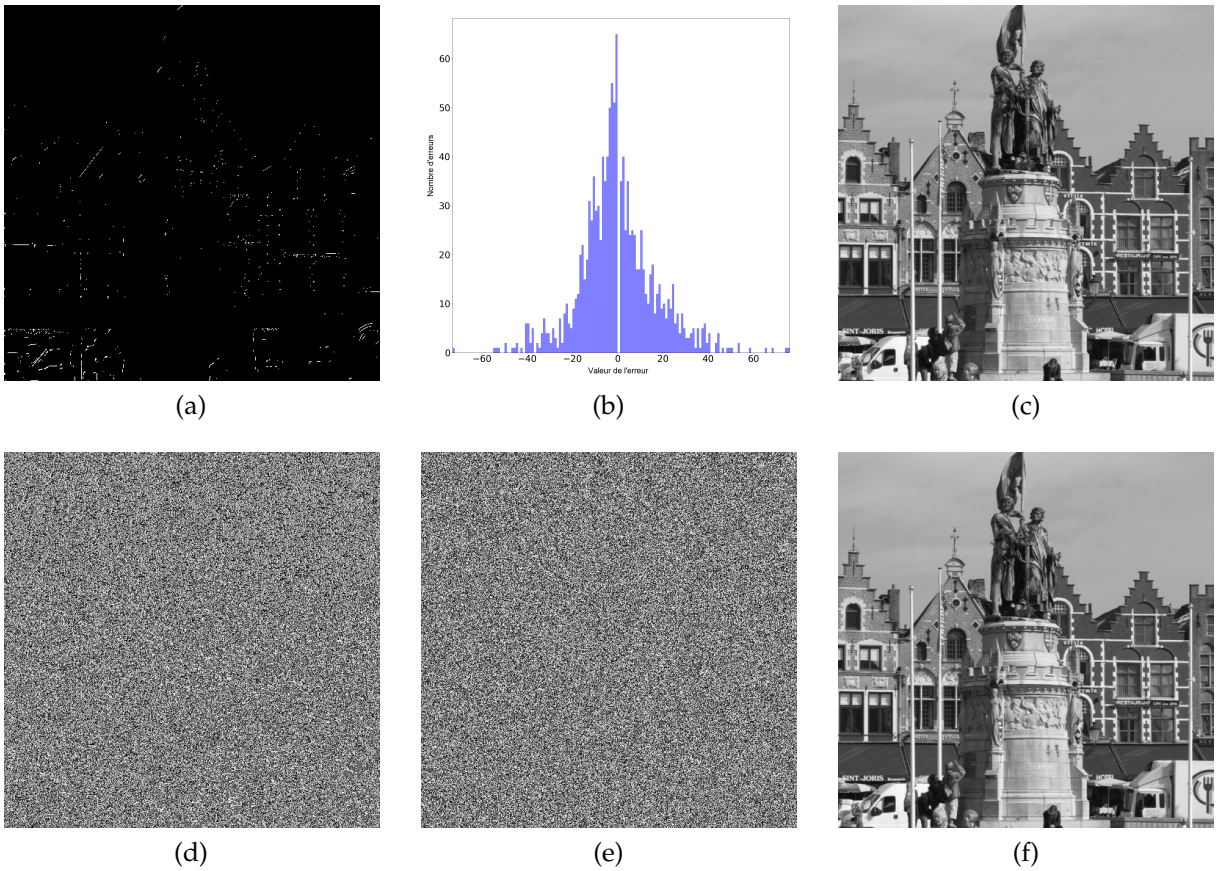
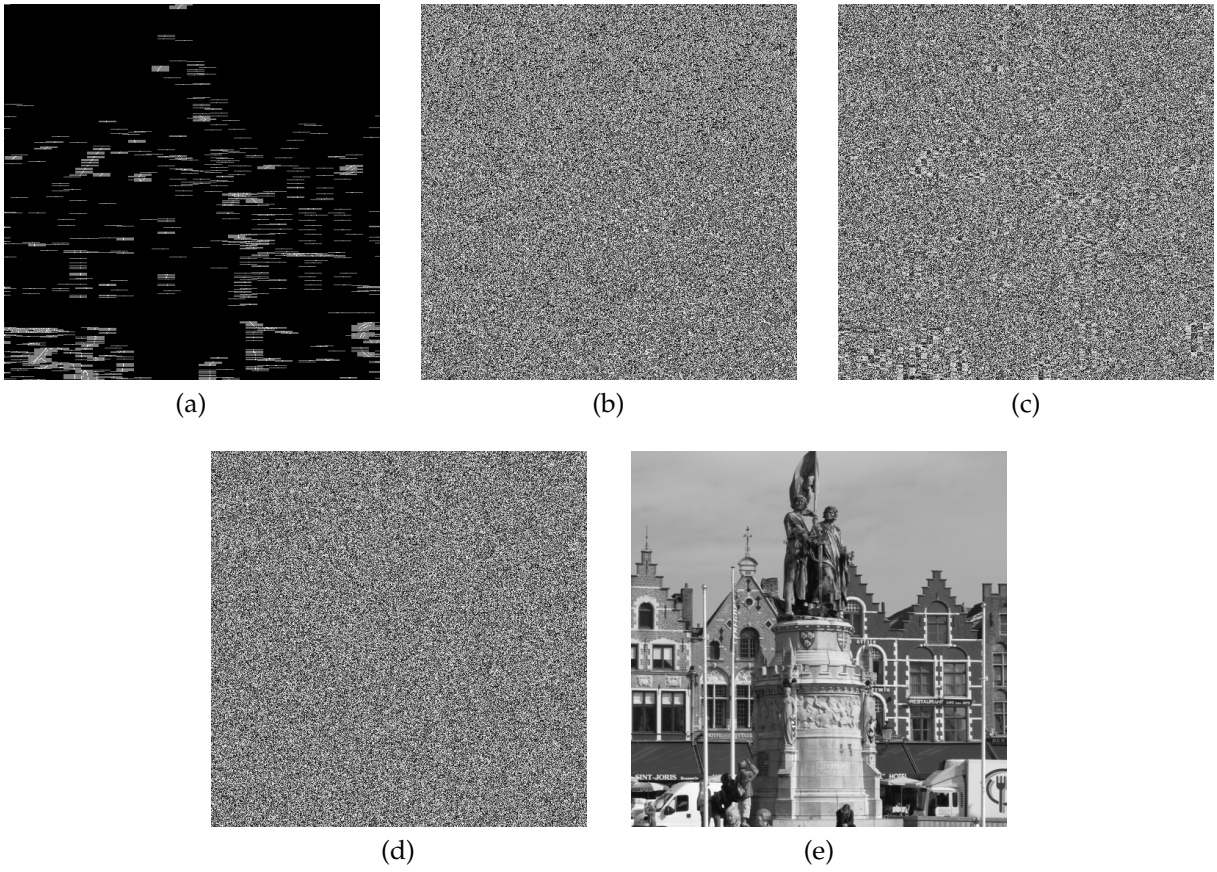


FIGURE 4.7 – Résultats expérimentaux obtenus avec l'approche IDCHC-CEP, avec une charge utile égale à $1\ bpp$: a) Emplacement des EP, nombre d'EP = $1242\ (0,47\%)$, b) Histogramme des EP, c) Image pré-traitée I', PSNR = $46,87\ dB$, d) Image chiffrée I'_c, e) Image chiffrée marquée I'_{c_m}, f) Image reconstruite I', PSNR = $46,87\ dB$, SSIM = $0,9997$.



(a) (b) (c) (d) (e)

FIGURE 4.8 – Résultats expérimentaux obtenus avec l'approche IDCHC-SEP : a) Emplacement des pixels non marqués (EP et drapeaux), nombre d'EP = 1225 $(0,46\%)$, b) Image chiffrée I_c, c) Image chiffrée I'_c avec les EP signalées, d) Image chiffrée marquée I_{c_m} avec une charge utile = $0,9220$ bpp, e) Image reconstruite I, PSNR $\rightarrow$ $+\infty$, SSIM = 1.

parfaitement reconstruite, comme l'attestent le PSNR qui tend vers $+\infty$ et le SSIM égal à 1 (fig. 4.8.e). Enfin, nous notons que le message secret est toujours extrait sans erreur quelle que soit l'approche utilisée.

4.4.2 Résultats obtenus sur une grande base de données

Les deux approches proposées ont également été appliquées à 10000 images en niveaux de gris de 512×512 pixels issues de la base BOWS-2 [3]. Nous notons que ces images présentent une grande variabilité statistique dans leur contenu. Le tableau 4.1 illustre les résultats obtenus sur cette base d'images. Dans $6,3\%$ des cas, quand il n'y a aucune EP (*i.e.* toutes les différences entre les pixels originaux et leurs prédicteurs sont inférieures ou égales à 64), les deux approches sont totalement réversibles. Dans ce cas, les images originales sont reconstruites sans erreur, comme indiqué par un PSNR tendant vers $+\infty$ et un SSIM égal à 1. De plus, il est possible de marquer tous les pixels des images et d'ainsi obtenir une charge utile maximale de 1 bpp. Dans les autres cas, pour l'approche IDCHC-CEP, la valeur de la charge utile est inchangée, mais les images originales ne peuvent pas être reconstruites sans perte car la correction des EP implique

la modification de certains pixels. Néanmoins, pour les images à faible contraste, la qualité des images reconstruites est haute. En effet, le PSNR est égal à $57,4\ dB$ et le SSIM est très proche de 1 $(0,9998)$ en moyenne. Par ailleurs, dans $98,64\%$ des cas, le PSNR est supérieur à $40\ dB$, ce qui indique une très haute qualité d'image. L'approche IDCHC-SEP, quant à elle, est totalement réversible pour toutes les images. Ainsi, le PSNR tend vers $+\infty$ et le SSIM est égal à 1. Même si tous les pixels ne sont pas marqués car quelques EP existent (en particulier dans le pire cas), la charge utile reste haute et en moyenne, sa valeur est de $0,9681\ bpp$. Notons que, dans $92,19\%$ des cas, elle est supérieure $0,9\ bpp$.

		Meilleur cas $(6,3\%)$	Pire cas	Moyenne
Approche IDCHC CEP	Pourcentage d'EP dans l'image originale	0%	$4,9\%$	$0,2\%$
	Charge utile (bpp)	1	1	1
	PSNR (dB)	$+\infty$	$29,0$	$57,4$
	SSIM	1	$0,9872$	$0,9998$
Approche IDCHC SEP	Pourcentage d'EP dans l'image originale	0%	$5,3\%$	$0,2\%$
	Charge utile (bpp)	1	$0,3805$	$0,9681$
	PSNR (dB)	$+\infty$	$+\infty$	$+\infty$
	SSIM	1	1	1

TABLE 4.1 – Performances des deux approches sur la base BOWS-2 (10000 images) [3].

Pour mieux visualiser la charge utile obtenue sur différentes images, en fig. 4.9, 500 images parmi les 10000 de la base BOWS-2 [3] ont été sélectionnées aléatoirement pour appliquer l'approche IDCHC-SEP.

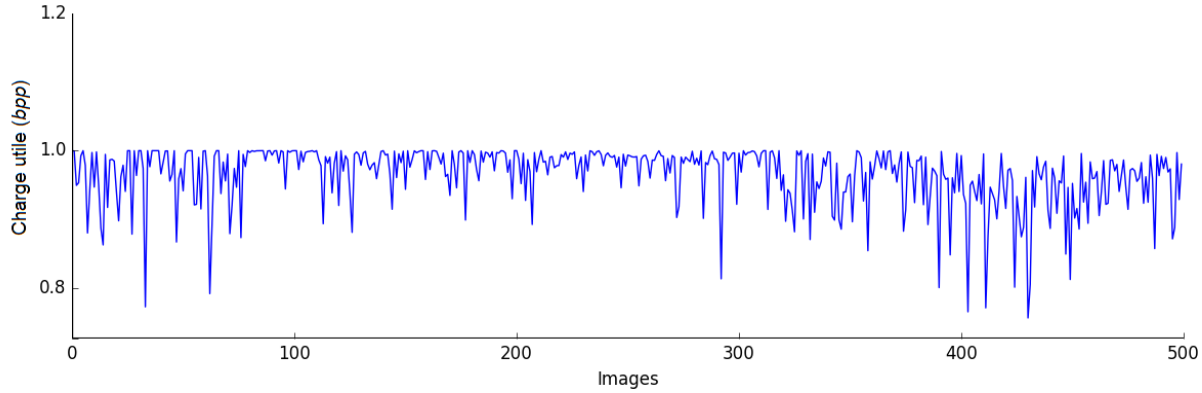


FIGURE 4.9 – Mesures de la charge utile, pour l'approche IDCHC-SEP, réalisées sur 500 images de la base BOWS-2 [3].

4.4.3 Analyse statistique

Nous avons également réalisé une analyse statistique sur les images obtenues avec nos deux approches afin d'évaluer le niveau de sécurité visuelle. Pour cela, comme expliqué en section 3.4.3, nous utilisons les métriques décrites dans la section 2.3.4.

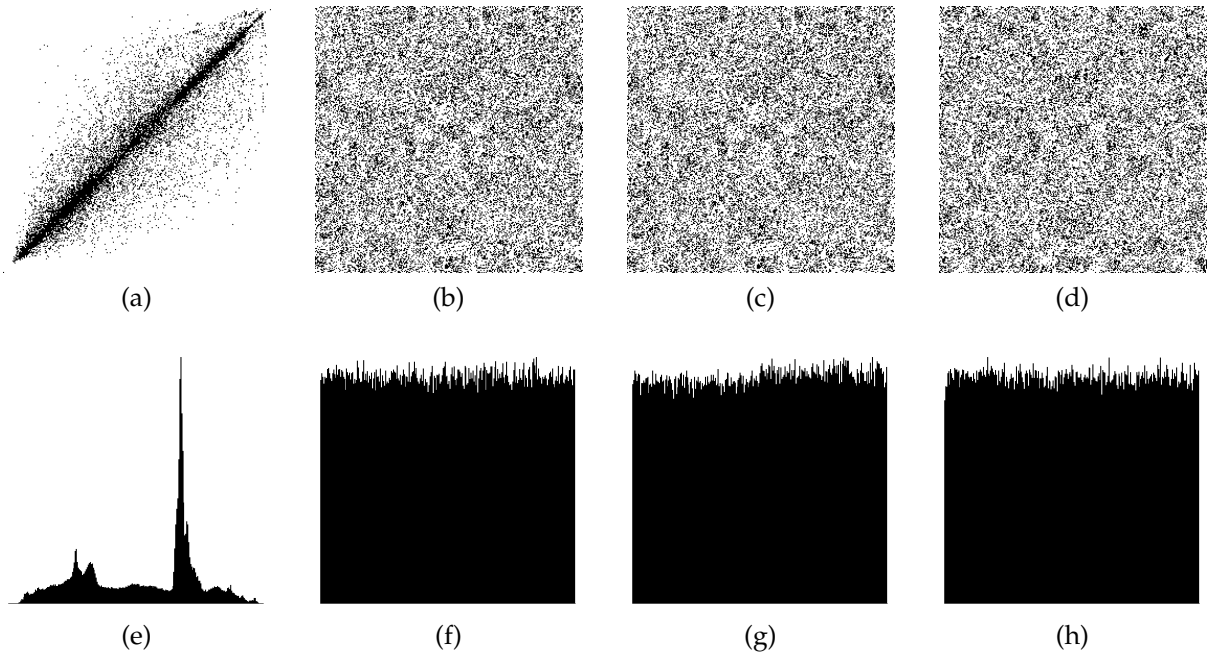


FIGURE 4.10 – Représentations statistiques : première ligne : Corrélation horizontale dans a) L'image originale (fig. 4.6), b) L'image chiffrée marquée avec l'approche IDCHC-CEP (fig. 4.7.e), c) L'image chiffrée avec les EP signalées avec l'approche IDCHC-SEP (fig. 4.8.c), d) L'image chiffrée marquée avec l'approche IDCHC-SEP (fig. 4.8.d) ; deuxième ligne : Histogrammes de e) L'image originale (fig. 4.6), f) L'image chiffrée marquée avec l'approche IDCHC-CEP (fig. 4.7.e), g) L'image chiffrée avec les EP signalées avec l'approche IDCHC-SEP (fig. 4.8.c), h) L'image chiffrée marquée avec l'approche IDCHC-SEP (fig. 4.8.d)

Image	Corrélation horizontale	Corrélation verticale	Entropie (bpp)	Test χ^2 (racine carrée)	NPCR (%)	UACI (%)	PSNR (dB)
Image originale (fig. 4.6)	$0,9388$	$0,9436$	$7,3227$	$668,628$	/	/	/
Image chiffrée, avec l'approche IDCHC-CEP (fig. 4.7.d)	$-0,0057$	$-0,0035$	$7,9994$	$14,8342$	$99,6143$	$30,1338$	$8,7081$
Image chiffrée marquée, avec l'approche IDCHC-CEP (fig. 4.7.e)	$-0,0062$	$-0,0015$	$7,9994$	$15,1188$	$99,6082$	$30,1521$	$8,7069$
Image chiffrée, avec l'approche IDCHC-SEP (fig. 4.8.b)	$0,0071$	$-0,0017$	$7,9994$	$14,8806$	$99,6136$	$30,1344$	$8,7081$
Image chiffrée avec les EP signalées, avec l'approche IDCHC-SEP (fig. 4.8.c)	$0,0362$	$0,0147$	$7,9991$	$18,2620$	$99,6071$	$30,1238$	$8,6834$
Image chiffrée marquée, avec l'approche IDCHC-SEP (fig. 4.8.d)	$-0,0016$	$0,0037$	$7,9994$	$14,8299$	$99,6059$	$30,1569$	$8,7039$

TABLE 4.2 – Évaluation de la sécurité des images obtenues avec nos deux approches.

Comme nous pouvons l'observer en fig. 4.10, la corrélation horizontale entre les pixels voisins de l'image originale est très forte (fig. 4.10.a) tandis qu'elle est très faible entre les pixels voisins dans les images chiffrées marquées (fig. 4.10.b et fig. 4.10.c) et

l'image chiffrée avec les EP signalées (fig. 4.10.d). De plus, l'histogramme de l'image chiffrée marquée obtenu avec l'approche IDCHC-CEP (fig. 4.10.f) et ceux de l'image chiffrée avec les EP signalées (fig. 4.10.g) et de l'image chiffrée marquée (fig. 4.10.h) obtenus avec l'approche IDCHC-SEP sont uniformément distribués en comparaison avec celui de l'image originale (fig. 4.10.e). Ainsi, ils ne peuvent pas être exploités pour obtenir des informations sur le contenu original de l'image en clair. En effet, la méthode de chiffrement utilisée permet de rendre pseudo-aléatoire la dépendance des propriétés statistiques entre les images chiffrées et l'image originale et cette caractéristique est conservée après l'insertion du message secret ou des informations de localisation d'EP, comme présenté dans le tableau 4.2. Dans l'image originale (fig. 4.6), il existe une très forte corrélation entre les pixels voisins, comme le montrent les valeurs proches de 1 ($0,9388$ et $0,9436$). Dans les images chiffrées ou chiffrées marquées, ces valeurs sont proches de zéro, ce qui signifie que la corrélation entre les pixels voisins est très faible. De plus, nous pouvons constater que la valeur de l'entropie est très haute pour les images chiffrées et chiffrées marquées ($\sim 7,9995\ bpp$) et proche de l'entropie maximale, ce qui indique que la distribution des pixels est proche de la distribution uniforme. En revanche, la valeur de l'entropie mesurée dans l'image originale est faible ($7,3227\ bpp$). Au regard des valeurs obtenues par le test du χ^2, nous observons qu'elle est très haute pour l'image originale ($668,628$) et bien plus faible pour les images chiffrées ou chiffrées marquées (~ 15). Cela signifie que la distribution des pixels des images chiffrées et chiffrées marquées peut être modélisée par une distribution uniforme est que leurs valeurs ne sont donc pas corrélées. Ainsi, nous pouvons attester que les images obtenues avec notre méthode d'IDCDC sont résistantes aux attaques statistiques. Nous avons aussi mesuré les valeurs du NPCR, de l'UACI et du PSNR entre l'image originale et les images chiffrées et chiffrées marquées. Les valeurs de NPCR sont très hautes et proches de la valeur maximale ($\sim 99,6\%$), l'UACI est proche de $30,15\%$ et le PSNR est très faible ($\sim 8,7\ dB$), ce qui indique que l'images originale et les images chiffrées et chiffrées marquées obtenues sont très différentes.

4.4.4 Comparaison avec l'état-de-l'art et discussion

En fig. 4.11, nous comparons notre méthode avec deux méthodes simples en considérant la même charge utile ($1\ bpp$) : les méthodes de substitution des LSB et celle de substitution naïve des MSB (*i.e.* sans détecter les EP). Pour cela, nous avons sélectionné aléatoirement 160 images parmi les 10000 testées de la base BOWS-2 [3] et représenté la valeur du PSNR associée pour chacune d'entre elles. Nous pouvons voir que, dans tous les cas, l'approche IDCHC-SEP permet d'atteindre de meilleurs résultats. La valeur du PNSR tend toujours vers $+\infty$ car l'image originale est parfaitement reconstruite. Par ailleurs, avec l'approche IDCHC-CEP, l'image reconstruite est généralement plus similaire à l'image originale qu'avec les deux méthodes simples. En effet, lorsqu'une substitution des LSB est effectuée, le PSNR est proche de $51\ dB$ dans tous les cas[2]. Avec

2. Si une substitution des LSB est réalisée, statistiquement environ la moitié des valeurs des pixels sont modifiées de ± 1. Ainsi, le calcul du PSNR s'effectue de la façon suivante :

$$\text{PSNR} = 10 \cdot \log_{10} \frac{255^2}{\frac{1}{m \times n} \cdot \frac{m \times n}{2}} = 10 \cdot \log_{10} \frac{255^2}{\frac{1}{2}} = 10 \cdot \log_{10}(130050) \approx 51\ dB.$$

la méthode naïve de substitution des MSB, l'image originale est presque toujours mal reconstruite car les EP n'ont pas été prises en compte. Il existe alors un phénomène de propagation des EP et le PSNR est inférieur à $20\ dB$. Il arrive qu'il soit un peu plus élevé mais des artefacts apparaissent dans l'image reconstruite à l'emplacement de tous les pixels avec une valeur erronée du MSB (*i.e.* une différence de 128 avec la valeur originale). Cependant, comme avec notre approche IDCHC-CEP, dans les images faiblement contrastées, il n'existe aucune EP et le PSNR tend alors vers $+\infty$.

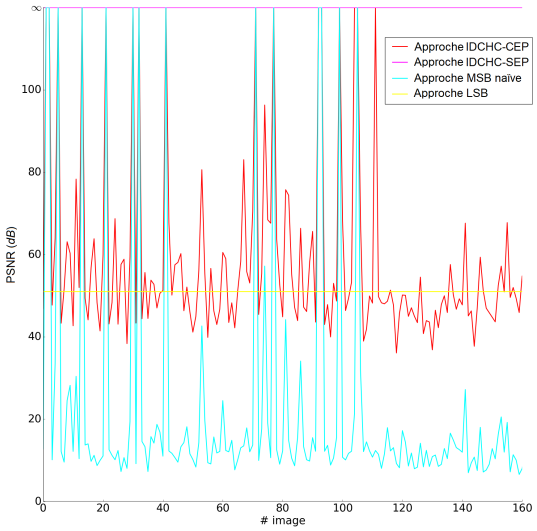
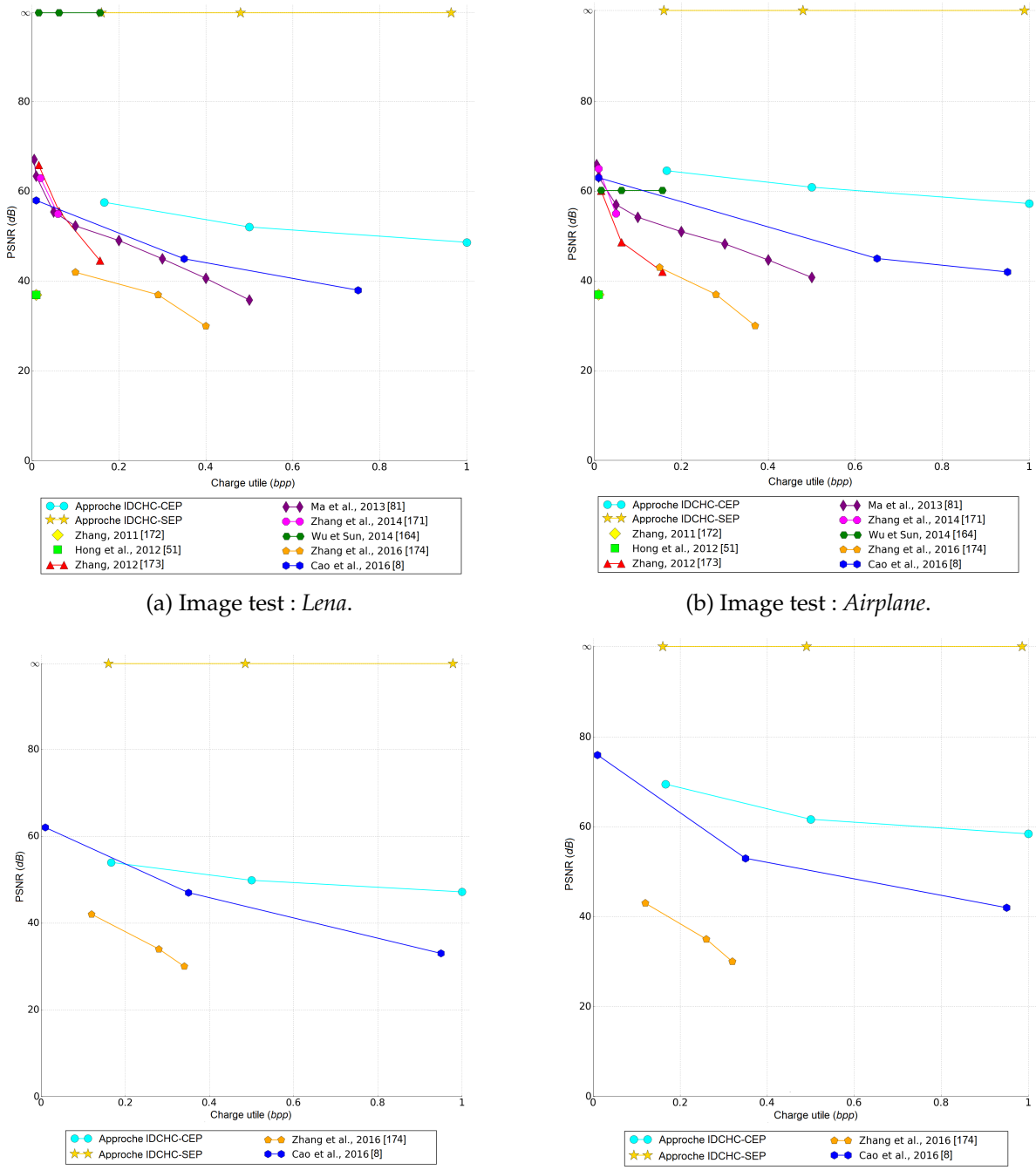


FIGURE 4.11 – Comparaison de la qualité des images reconstruites obtenues à l'aide de notre méthode et deux méthodes simples, en considérant la même charge utile ($1\ bpp$).

Nous avons également réalisé des comparaisons, en termes de charge utile et de qualité de l'image reconstruite, entre nos deux approches proposées et huit méthodes de l'état-de-l'art : les méthodes très récentes proposées par Zhang *et al.* [174] et Cao *et al.* [8] (fig. 4.12.a–d) et les méthodes de Zhang [172], Hong *et al.* [51], Zhang [173], Ma *et al.* [81], Zhang *et al.* [171] et Wu et Sun [164] (fig. 4.12.a–b).

Pour cela, nous avons utilisé les images de *Lena*, *Airplane*, *Man* et *Crowd*. Tout d'abord, nous pouvons voir que nos deux approches permettent d'obtenir une charge utile plus grande que les autres méthodes quelle que soit l'image utilisée. En effet, la valeur maximale de la charge utile des méthodes de l'état-de-l'art, obtenue par Cao *et al.* est $0,95\ bpp$. Avec notre approche IDCHC-CEP, nous pouvons insérer $1\ bpp$ et avec l'approche IDCHC-SEP, nous obtenons des résultats très proches de cette valeur. Puisque nous n'utilisons pas de fichier additionnel pour renseigner l'emplacement des EP dans l'approche IDCHC-SEP, nous devons diminuer la charge utile de $0,0359\ bpp$ pour *Lena*, $0,0111\ bpp$ pour *Airplane*, $0,0212\ bpp$ pour *Man* et $0,0145\ bpp$ pour *Crowd*. Par ailleurs, lorsque nous examinons la qualité de l'image reconstruite, notre approche IDCHC-SEP est la seule qui permet de reconstruire parfaitement l'image originale lorsque seule la clé de chiffrement est connue et sans avoir besoin de la clé d'insertion (PSNR $\rightarrow +\infty$). Aucune des autres méthodes ne permet d'obtenir de tels résultats pour toutes les images. Seule l'image de *Lena* est exactement la même que l'image originale en utilisant la méthode de Wu et Sun. De plus, pour toutes les autres images, nous pouvons voir que nos deux approches permettent d'obtenir de meilleurs résultats que



(a) Image test : *Lena*.

(b) Image test : *Airplane*.

(c) Image test : *Man*.

(d) Image test : *Crowd*.

FIGURE 4.12 – Comparaison des performances entre nos deux approches proposées et les méthodes récentes de l'état-de-l'art pour quatre images tests.

les autres méthodes, même lorsque la charge utile est importante. En outre, lorsque la valeur de la charge utile est fixée à $0,1667$ *bpp* (un pixel marqué sur six) ou à $0,5$ *bpp* (un pixel marqué sur deux), la qualité de l'image reconstruite est vraiment très haute avec nos deux approches.

En conclusion, en plus de permettre une extraction du message secret sans erreur dans tous les cas, notre méthode, quelle que soit l'approche utilisée, permet d'obtenir

un très bon compromis entre la charge utile et la qualité de l'image reconstruite en utilisant seulement la clé de chiffrement. D'un point de vue sécurité, notre analyse statistique a montré qu'aucune information sur l'image originale n'est visible sur l'image chiffrée ou chiffrée marquée. De plus, si une partie du message secret est modifiée par un attaquant, comme le message est chiffré avant son insertion, il ne peut pas être déchiffré et exploité pour de l'authentification. De plus, dans l'approche IDCHC-SEP, si le message est modifié ou supprimé, l'image en clair ne peut pas être reconstruite. Nous notons que même si, dans la méthode proposée, le message secret est utilisé à des fins d'authentification ou d'enrichissement des données, il peut être aussi utilisé comme une alternative aux codes correcteurs d'erreur pour un contrôle d'intégrité par exemple.

4.5 Conclusion

Dans ce chapitre, nous avons décrit une méthode efficace d'insertion de données cachées dans les images chiffrées basée sur la prédiction du MSB de chaque pixel, dont les performances dépassent celles des méthodes récentes de l'état-de-l'art. A notre connaissance, cette méthode d'IDCDC est l'une des premières à proposer d'utiliser les MSB à la place des LSB. En effet, la prédiction des valeurs des MSB est plus simple que celle des valeurs des LSB dans le domaine clair. De plus, comme altérer la qualité visuelle des images chiffrées n'est pas un problème, une très haute charge utile est atteinte. En analysant le contenu de l'image originale dans le domaine clair, l'emplacement des EP est détecté puis renseigné dans une carte binaire de localisation des EP. Dans l'approche IDCHC-CEP, l'image originale est pré-traitée pour éliminer toutes les EP. Le MSB de chaque pixel de l'image chiffrée est alors substitué par un bit du message secret. Ainsi, la charge utile est égale à $1\ bpp$ et la qualité de l'image reconstruite est très élevée, comme indiqué par une valeur du SSIM proche de 1 et un PSNR moyen de $57,4\ dB$. Dans l'approche IDCHC-SEP, l'emplacement des EP est signalé dans l'image chiffrée. Notons que, contrairement à de nombreuses méthodes d'IDCDC, cela n'implique pas le stockage de la carte de localisation. En effet, les MSB de certains pixels sont utilisés pour signaler l'emplacement des EP à la place d'être substitués par des bits du message secret. Bien que cela implique une diminution de la charge utile par rapport à l'approche IDCHC-SEP, la plupart des MSB sont marqués par le message secret et la charge utile est proche d'$1\ bpp$. De plus, pendant la phase de décodage, l'image originale peut être reconstruite parfaitement (PSNR $\rightarrow +\infty$). Enfin, nous avons vu que la méthode proposée offrait un bon niveau de sécurité et pouvait être utilisée pour préserver la confidentialité visuelle de l'image originale, tout en permettant de contrôler son intégrité et son authenticité.

Dans la suite de nos travaux de recherche, nous nous sommes intéressés à atteindre une charge utile supérieure à un $1\ bpp$. En effet, nous avons émis l'hypothèse qu'il était possible d'utiliser par exemple le deuxième MSB de chaque pixel pour augmenter la valeur de la charge utile. Dans le chapitre 5, nous montrons que l'ensemble des plans binaires d'une image peuvent en fait être exploités. Nous décrivons alors une extension de l'approche IDCHC-SEP et une nouvelle méthode d'IDCDC réversible, où les plans binaires d'une image sont parcourus récursivement.

Ces travaux ont fait l'objet de trois publications internationales. Tout d'abord, l'ap-

proche IDCHC-CEP a été présentée au cours de la conférence internationale IEEE IPTA 2016 [113] et l'approche IDCHC-SEP, au cours de la conférence internationale Electronic Imaging 2017 [105]. Par ailleurs, la méthode complète d'IDCDC basée sur la prédiction du MSB a été décrite en détail dans la revue internationale IEEE Transactions of Information Forensics and Security en 2018 [106].

CHAPITRE 5

Insertion de données cachées dans les images chiffrées : traitement récursif des plans binaires

Sommaire

5.1 Introduction

Dans ce chapitre, nous nous intéressons aux méthodes d'IDCDC réversibles à haute capacité, et particulièrement à celles basées sur le traitement de tous les plans binaires de l'image. Nous commençons par décrire les méthodes récentes de l'état-de-l'art publiées après 2018, suite à la parution de l'article décrivant la méthode détaillée dans le chapitre 4. Nous présentons alors une extension possible de l'approche IDCHC-SEP de cette dernière. En outre, le reste de ce chapitre est consacré à la description de notre méthode d'IDCDC réversible et à très haute capacité. Tous les plans binaires d'une image sont traités récursivement du plan MSB au plan LSB. Pour chaque plan binaire, différentes étapes sont réalisées : la prédiction des erreurs de prédiction (EP), l'adaptation réversible, le chiffrement et l'IDC. Pour prédire les valeurs des pixels, le prédicteur *Median Edge Detector* (MED), également nommé LOCO-I et connu pour son utilisation dans le standard de compression JPEG-LS, est utilisé. Contrairement à certaines méthodes de l'état-de-l'art, cette nouvelle méthode d'IDCDC ne nécessite pas de pré-traitement de l'image originale pour corriger les EP, ni d'utiliser des drapeaux pour signaler leur emplacement. En effet, une adaptation réversible des plans binaires est réalisée pour rendre possible la détection et la correction des EP pendant la phase de décodage. Grâce à la forte corrélation entre les pixels dans le domaine clair, une grande partie des bits de l'image peuvent être substitués par ceux d'un message secret. Nos résultats expérimentaux montrent que l'IDC peut généralement être réalisée jusqu'au quatrième plan binaire d'une image, ce qui permet d'obtenir expérimentalement une charge utile de $2,4586$ *bpp* en moyenne sur une base de 10000 images.

La section 5.2 présente les approches de l'état-de-l'art précédentes, celles-ci faisant suite à la publication de notre méthode décrite dans le chapitre 4. Une extension de l'approche IDCHC-SEP de cette méthode est détaillée en section 5.3. Notre nouvelle méthode d'IDCDC réversible proposée est alors développée en détail dans la section 5.4. Les résultats expérimentaux réalisés sont ensuite décrits en section 5.5. Enfin, la section 5.6 conclut ce chapitre et présente des travaux futurs.

5.2 IDCDC basée MSB (après 2018)

En 2018, comme décrit dans le chapitre 4, nous avons proposé d'utiliser les valeurs des MSB à la place des LSB pour réaliser l'IDC d'un message secret. En effet, nous avons montré que la substitution des MSB dans le domaine chiffré n'introduit pas d'artefacts et que leurs valeurs sont plus faciles à prédire que celles des LSB. Ainsi, nous avons proposé deux approches d'IDCDC à haute capacité : l'approche d'insertion de données cachées haute capacité avec correction des erreurs de prédiction (IDCHC-CEP) et l'approche d'insertion de données cachées haute capacité avec signalisation des erreurs de prédiction (IDCHC-SEP). Dans ces deux approches, tous les pixels de l'image en clair qui ne peuvent pas être prédits en utilisant les valeurs de leurs voisins sont identifiés. Dans l'approche IDCHC-CEP, l'image originale est pré-traitée pour éliminer toutes les EP. L'image pré-traitée est alors chiffrée et les MSB de tous les pixels sont aveuglément remplacés par les bits d'un message secret. Dans ce cas, la charge utile est d'1 *bpp* et l'image reconstruite correspond à l'image pré-traitée et est proche de l'image

originale (PSNR $> 50\ dB$). Dans l'approche IDCHC-SEP, l'image originale est chiffrée sans aucune modification. Après le chiffrement, l'information nécessaire pour signaler l'emplacement des EP est insérée par substitution des MSB de l'image chiffrée. L'IDC est alors réalisée en détectant les bits pouvant être marqués et en les substituant par les bits d'un message secret. Dans ce cas, la charge utile est légèrement inférieure à $1\ bpp$ mais une réversibilité parfaite est obtenue. Dans leur article [116], Puyang *et al.* proposent une première extension de l'approche IDCHC-SEP. Ils suggèrent d'utiliser un autre prédicteur, plus efficace, pour ainsi limiter le nombre d'EP. De plus, ils expliquent que le premier et le deuxième plans MSB peuvent être utilisés pour l'IDC. Grace à l'utilisation du deuxième plan binaire, une charge utile de $1,35\ bpp$ en moyenne est obtenue. Yi *et al.* ont proposé une méthode basée sur l'étiquetage d'un arbre binaire paramétrique, où la corrélation spatiale entre les pixels du domaine clair est conservée dans le domaine chiffré au sein de blocs de petite taille [170]. Certains pixels sont utilisés comme valeurs de référence pour calculer les EP qui sont ensuite mises en évidence grâce à l'étiquetage d'un arbre binaire paramétrique. Pour réaliser l'IDC, les bits d'un message secret sont insérés par substitution en exploitant la corrélation spatiale entre les pixels. Les auteurs atteignent ainsi une valeur de la charge utile de l'ordre de $2\ bpp$. Dans leur méthode [16], Chen et Chang réalisent un réarrangement des plans MSB par bloc. Cela leur permet de transformer les plans MSB de l'image originale en une séquence binaire pouvant être compressée en utilisant un codage par plages étendu. Grâce à ce réarrangement et à la compression, ils peuvent libérer un grand espace pour l'IDC. Pendant la phase de décodage, le message secret inséré peut être extrait directement dans le domaine chiffré avec l'aide de la clé d'insertion et l'image marquée en clair peut être obtenue avec la clé de chiffrement.

5.3 Extension de l'approche IDCHC-SEP

En tant qu'extension de l'approche IDCHC-SEP décrite dans le chapitre 4, nous avons proposé d'utiliser tous les plans binaires de l'image originale de manière itérative à la place d'exploiter seulement le plan MSB. En effet, le plan MSB n'est pas le seul plan binaire pouvant être facilement prédit dans le domaine clair. En effet, les MSB suivants peuvent également être exploités pour l'IDC. Dans l'approche proposée, en commençant par le plan MSB, chaque plan binaire est analysé itérativement pour mettre en évidence les EP, puis chiffré. Si la quantité d'information nécessaire pour signaler les EP est inférieure à la taille du plan binaire courant, les drapeaux sont alors insérés et les bits restants sont substitués par ceux d'un message secret. Les mêmes opérations sont ensuite réalisées sur les plans binaires suivants tant que l'IDC est possible. Après le décodage, l'image originale peut être parfaitement reconstruite en utilisant les informations sur la localisation des EP et la prédiction. Les résultats expérimentaux obtenus montrent que la charge utile peut être bien supérieure à $1\ bpp$. Sur l'ensemble des images de la base BOWS-2 [3], la valeur médiane est de $1,749\ bpp$ et la moyenne est de $1,836\ bpp$. Dans le meilleur des cas, elle atteint même $5,408\ bpp$.

Dans l'approche IDCHC-CEP présentée dans le chapitre 4, l'étape de pré-traitement altère l'image originale, qui ne peut pas être parfaitement reconstruite lors de la phase de décodage. En effet, les pixels adaptés ne peuvent pas être localisés et corrigés. De plus, dans l'approche IDCHC-SEP, également présentée dans le chapitre 4, des drapeaux

sont utilisés pour signaler l'emplacement des EP. En plus d'entraîner une diminution de la valeur de la charge utile, cela constitue également une faille de sécurité et, dans des cas particuliers, certains drapeaux peuvent être mal détectés. De plus, dans ces deux approches, la valeur maximale de la charge utile est égale à 1 *bpp* car seul le plan MSB est exploité pour l'IDC. Dans leur méthode, Puyang *et al.* ont proposé d'utiliser également le deuxième plan binaire [116]. La méthode décrite ci-dessus permet d'améliorer toutes ces approches. En effet, tous les plans binaires de l'image sont traités itérativement dans le but d'exploiter pleinement la redondance entre les pixels dans le domaine clair. Cependant, même si cette méthode est une extension intéressante de l'approche IDCHC-SEP, des drapeaux sont toujours présents et nécessaires pour signaler les EP.

5.4 Nouvelle méthode d'IDCDC proposée

Dans cette section, nous décrivons un nouvelle méthode d'IDCDC réversible avec une très haute valeur de la charge utile. Cette nouvelle approche est totalement réversible et permet d'obtenir une très haute charge utile car elle est pleinement récursive. Contrairement aux travaux précédents, elle n'est pas une extension de l'approche IDCHC-SEP décrite dans le chapitre 4 : les pixels mal prédits ne sont pas corrigés ou signalés à l'aide drapeaux. En effet, une adaptation des plans binaires est réalisée pour créer des configurations spécifiques permettant de détecter et de corriger tous les pixels mal prédits pendant la phase de décodage.

Tout d'abord, en section 5.4.1, nous décrivons les étapes générales de la phase d'encodage faisant intervenir le propriétaire de l'image et le propriétaire du message caché. L'image originale est traitée récursivement, en parcourant tous les plans binaires, du MSB au LSB. Pour chaque plan binaire, ce traitement implique trois différentes étapes : 1) la prise en compte des EP en utilisant le *Median Edge Detector* (MED) comme prédicteur [83] (section 5.4.2) ; 2) l'adaptation réversible des plans binaires (section 5.4.3) ; 3) le chiffrement du plan binaire courant et la phase d'IDC (section 5.4.4). Cette dernière comprend l'insertion de la liste des valeurs des EP et d'un message secret par substitution des bits des plans binaires pouvant être marqués. Ainsi, une image chiffrée marquée est obtenue. La phase de décodage est alors décrite en section 5.4.5. Puisque notre méthode est séparative, l'extraction du message secret et la reconstruction de l'image originale peuvent être réalisées séparément et sans erreur.

5.4.1 Description générale de la méthode

Dans la méthode proposée, une image originale I de $m \times n$ pixels codés sur 256 niveaux de gris, est considérée comme une pile de 8 plans binaires $I^{[k]}$, avec $0 \leq k \leq 7$. L'image originale I est ainsi renommée $I^{[0,7]}$. Nous notons $I_k^{[k,7]}$ l'image originale après k modifications. Pendant la phase d'encodage, les plans binaires sont parcourus récursivement, du plus significatif $I^{[0]}$ (aussi appelé plan MSB) au moins significatif $I^{[7]}$ (plan LSB). En effet, l'image $I_k^{[k+1,7]}$, composée des $7-k$ plans LSB, est nécessaire pour traiter le plan binaire courant $I_k^{[k]}$. Un schéma général de la phase d'encodage, impliquant le propriétaire de l'image et le propriétaire du message caché, est présenté en fig. 5.1.

Pour chaque plan binaire courant $I_k^{[k]}$, comme illustré en fig. 5.2, la première étape

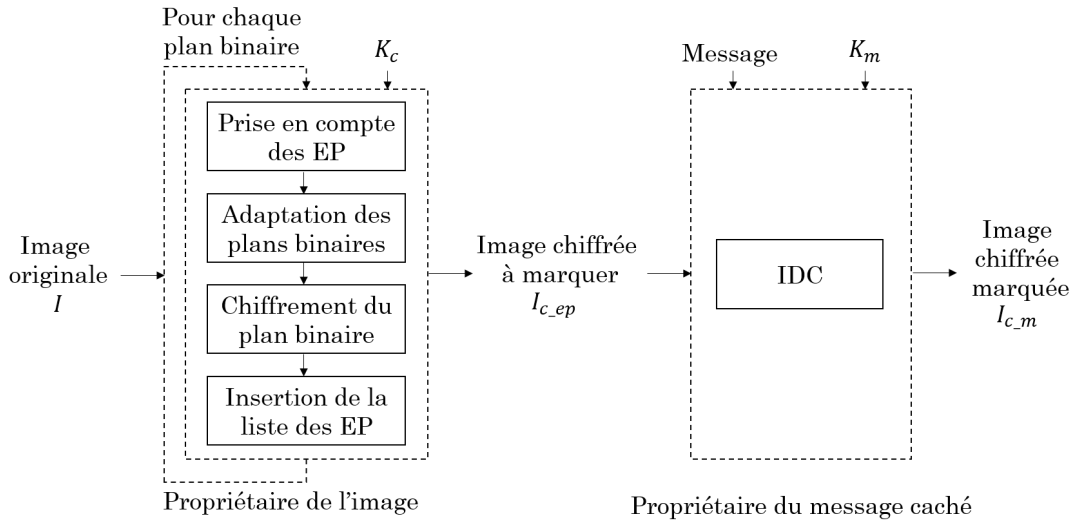


FIGURE 5.1 – Schéma général de la phase d'encodage de notre méthode d'IDCDC récursive.

du processus récursif consiste à prendre en compte les erreurs de prédiction (EP) dans l'image $I_k^{[k,7]}$, composée par les $8-k$ plans binaires LSB de l'image en cours de traitement. Une carte de localisation des EP et une liste des valeurs des EP sont construites pendant cette étape. A la fin, un test est réalisé pour savoir si la liste des valeurs des EP peut être insérée dans le plan binaire courant $I_k^{[k]}$, *i.e.* si la taille de la liste des valeurs des EP est inférieure à la taille du plan binaire. Si elle peut être insérée, l'image $I_k^{[k,7]}$ est adaptée pour rendre possible l'identification des EP, et finalement, l'image $I_{k+1}^{[k,7]}$ est obtenue. Notons que dans $I_{k+1}^{[k,7]}$, non seulement le plan binaire courant est adapté, mais aussi tous les autres plans binaires. Après cette étape, le plan binaire courant $I_{k+1}^{[k]}$ est chiffré avec la clé de chiffrement K_c. La liste des valeurs des EP est alors insérée dans le plan binaire chiffré en substituant les premiers bits autant que nécessaire pour obtenir le plan binaire $I_{c\_ep}^{[k]}$. Les étapes précédemment décrites sont alors récursivement répétées sur l'image $I_{k+1}^{[k+1,7]}$, obtenue après avoir adapté $I_k^{[k,7]}$ et traité séparément les plan MSB $I_{k+1}^{[k]}$. En revanche, si la liste des valeurs des EP ne peut pas être insérée dans le plan binaire courant, le processus récursif est interrompu. Dans ce cas, les $K = k$ premiers plans binaires (*i.e.* les plans binaires chiffrés après avoir inséré les valeurs des EP) sont ensuite empilés pour obtenir l'image $I_{c\_ep}^{[0,K-1]}$. Par ailleurs, le plan binaire courant et les $8 - K$ plans moins significatifs sont directement et simplement chiffrés en utilisant la clé de chiffrement K_c pour générer l'image chiffrée $I_c^{[K,7]}$. Enfin, l'image intégralement chiffrée et comportant les valeurs des EP I_{c_ep} consiste en un empilement de $I_{c\_ep}^{[0,K-1]}$ et $I_c^{[K,7]}$.

Après la transmission de l'image, le propriétaire des données à cacher commence par chiffrer le message secret à l'aide de la clé d'insertion K_m. Le message chiffré est alors inséré par substitution des bits dans chaque plan binaire pouvant être marqué $I_{c\_ep}^{[k]}$ avec $0 \leq k < K$, à la suite de la liste des valeurs des EP. De cette façon, le plan binaire chiffré marqué final $I_{c\_m}^{[k]}$ est obtenu. Comme illustré en fig. 5.1, l'image chiffrée

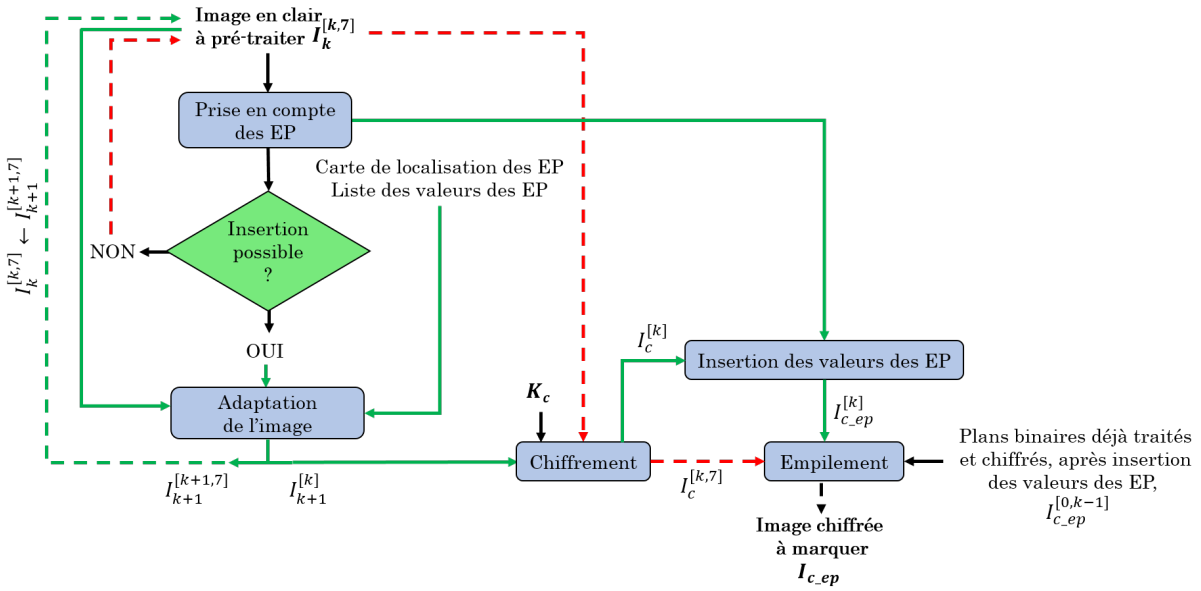


FIGURE 5.2 – Traitement de l'image $I_k^{[k,7]}$ par son propriétaire pendant la phase d'encodage.

marquée I_{c_m} consiste en huit plans binaires chiffrés, dont certains sont marqués. En effet, les plans binaires significatifs sont marqués par les bits de la liste des valeurs des EP et ceux d'un message secret, tandis que les moins significatifs sont uniquement chiffrés.

5.4.2 Calcul et analyse des erreurs de prédiction

Dans la méthode proposée, les bits d'un message secret sont insérés par substitution. Par conséquent, les valeurs originales des bits du plan binaire courant sont perdues pendant la phase d'IDC. Afin de pouvoir reconstruire parfaitement l'image originale, les valeurs originales des bits doivent être prédictibles. Ainsi, nous proposons de prédire la valeur de chaque pixel $p_k^{[k,7]}(i,j)$ de l'image $I_k^{[k,7]}$ en utilisant les $7-k$ plans LSB $I_k^{[k+1,7]}$ et les pixels précédemment parcourus. La première étape du traitement du plan binaire courant $I_k^{[k]}$ avec notre méthode récursive consiste à analyser le contenu de l'image $I_k^{[k,7]}$, composée des $8-k$ plans LSB de l'image en clair. Pendant ce processus, tous les pixels ne pouvant pas être prédits à l'aide de leurs voisins sont identifiés et les valeurs des EP associées sont évaluées. Notons que la valeur du premier bit $p_k^{[k]}(0,0)$ ne peut pas être prédite et reste donc inchangée. Elle est seulement chiffrée et sert à initialiser la prédiction.

Considérons un pixel $p_k^{[k,7]}(i,j)$ de $I_k^{[k,7]}$. Il est composé de $8-k$ bits et défini de la façon suivante :

$$p_k^{[k,7]}(i,j) = \sum_{l=k}^{7} p_k^{[l]}(i,j) \times 2^{7-l}, \tag{5.1}$$

où $p_k^{[l]}(i,j)$ est le bit d'indice l.

Pour la prédiction de ce pixel, son contexte, illustré en fig. 5.3, est observé. En tant

que prédicteur, le prédicteur *Median Edge Detector* (MED), aussi nommé LOCO-I [83], est utilisé. Ce prédicteur consiste à détecter la valeur maximale entre les contours horizontaux et verticaux avec l'algorithme LOCO-I. Il est reconnu pour son efficacité dans le standard de compression JPEG-LS compression. Ainsi, le prédicteur $pred(i,j)$ du pixel $p_k^{[k,7]}(i,j)$ est obtenu :

$$\begin{aligned} pred(i,j) &= \text{MED}(p_k^{[k,7]}(i,j)) \\ &= \begin{cases} \min(A,B) & \text{si } C \geq \max(A,B), \\ \max(A,B) & \text{si } C \leq \min(A,B), \\ A+B-C & \text{sinon.} \end{cases} \end{aligned} \tag{5.2}$$

L'algorithme de prise en compte des EP est présenté dans l'algorithme 2. Après le calcul du prédicteur, l'inverse de $p_k^{[k,7]}(i,j)$ est évalué : $inv(i,j) = (p_k^{[k,7]}(i,j) + 2^{7-k}) \mod 2^{8-k}$. Cette valeur est en fait obtenue en inversant la valeur du MSB $p_k^{[k]}(i,j)$ et en laissant inchangés les plans LSB $p_k^{[k+1,7]}(i,j)$. Par conséquent, il existe une différence de 2^{7-k} entre $p_k^{[k,7]}(i,j)$ et son inverse $inv(i,j)$. Les valeurs absolues des différences entre $p_k^{[k,7]}(i,j)$ et $pred(i,j)$, et entre $inv(i,j)$ et $pred(i,j)$ sont alors calculées et nommées Δ et Δ^{inv}. Si $\Delta < \Delta^{inv}$, alors la valeur originale du bit peut être prédite. En effet, cela signifie que la valeur correcte du pixel est plus proche de son prédicteur que la valeur inverse. En revanche, si $\Delta > \Delta^{inv}$, une EP existe lors de la prédiction du pixel courant et est alors renseignée dans la carte de localisation des EP $\mathbf{L_{loc}^k}$. L'amplitude des EP est ensuite calculée et stockée dans la liste des valeurs des EP $\mathbf{L_{val}^k}$.

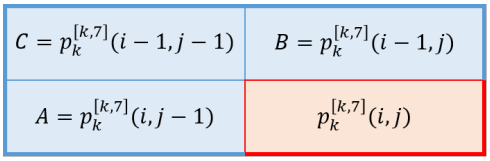


FIGURE 5.3 – Contexte pour la prédiction du pixel $p_k^{[k,7]}(i,j)$.

Notons que nous devons aussi vérifier que Δ ou Δ^{inv} sont différents de 2^{6-k} et $2^{6-k} + 2^{7-k}$. En effet, dans ces cas de figure (et en particulier quand $\Delta = \Delta^{inv} = 2^{6-k}$), la valeur correcte de $p_k^{[k,7]}(i,j)$ ne peut pas être déterminée. Pour cette raison, nous devons également indiquer une erreur dans $\mathbf{L_{loc}^k}$ et nous renseignons un code spécial dans $\mathbf{L_{val}^k}$: $-(2^{6-k}+1)$ si $p_k^{[k]}(i,j) = 0$, et $(2^{6-k}+1)$ si $p_k^{[k]}(i,j) = 1$.

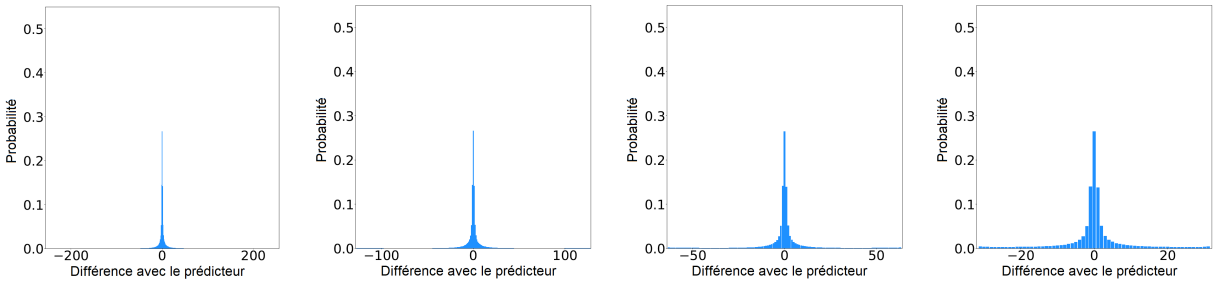
Après avoir parcouru entièrement le plan binaire $I_k^{[k]}$, nous calculons la taille de $\mathbf{L_{val}^k}$. Si cette taille, en bits, est inférieure à la taille du plan binaire courant, elle est stockée en substituant les premiers bits de ce plan après le chiffrement. Si ce n'est pas le cas, la détection des pixels ne pouvant pas être prédits et l'évaluation des EP sont interrompues et le plan courant et tous les plans LSB restants sont seulement chiffrés, comme décrit en section 5.4.4.

Algorithme 2 : Prise en compte des EP pour le $k^{\text{ème}}$ plan MSB $I_k^{[k]}$.

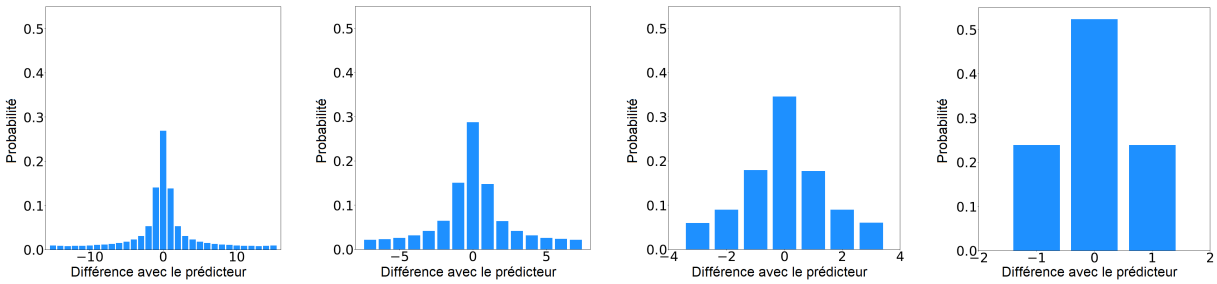
Données : Image en clair $I_k^{[k,7]}$ de $m \times n$ pixels $p_k^{[k,7]}(i,j)$ codés sur $8-k$ bits
Résultat : Carte de localisation des EP $\mathbf{L_{loc}^k}$ et liste des valeurs des EP $\mathbf{L_{val}^k}$

$\mathbf{L_{loc}^k} \leftarrow [\,]$;
$\mathbf{L_{val}^k} \leftarrow [\,]$;
pour $i \leftarrow 0$ **à** m **faire**
  pour $j \leftarrow 0$ **à** n **faire**
    si $i = 0$ **ou** $j = 0$ **et** $(i,j) \neq (0,0)$ **alors**
      Prédiction unidirectionnelle;
    sinon
      $pred(i,j) \leftarrow \text{MED}(p_k^{[k,7]}(i,j))$;
    $inv(i,j) \leftarrow (p_k^{[k,7]}(i,j) + 2^{7-k}) \mod 2^{8-k}$;
    $\Delta \leftarrow |pred(i,j) - p_k^{[k,7]}(i,j)|$;
    $\Delta^{inv} \leftarrow |pred(i,j) - inv(i,j)|$;
    si $(\Delta < \Delta^{inv})$ **et** $(\Delta \neq 2^{6-k}$ **ou** $\Delta^{inv} \neq 2^{6-k} + 2^{7-k})$ **alors**
      /* Il n'y a pas d'EP */
      $\mathbf{L_{loc}^k}$.ajouter(0);
    sinon si $(\Delta > \Delta^{inv})$ **et** $(\Delta \neq 2^{6-k} + 2^{7-k}$ **ou** $\Delta^{inv} \neq 2^{6-k})$ **alors**
      /* Il y a une EP */
      $\mathbf{L_{loc}^k}$.ajouter(1);
      si $p_k^{[k,7]}(i,j) < 2^{7-k}$ **alors**
        si $pred(i,j) \leq inv(i,j)$ **alors**
          $x \leftarrow pred(i,j) - p_k^{[k,7]}(i,j) - 2^{6-k}$;
        sinon
          $x \leftarrow pred(i,j) - inv(i,j) - 2^{6-k}$;
      sinon
        si $pred(i,j) \geq inv(i,j)$ **alors**
          $x \leftarrow pred(i,j) - p_k^{[k,7]}(i,j) + 2^{6-k}$;
        sinon
          $x \leftarrow pred(i,j) - inv(i,j) + 2^{6-k}$;
      $\mathbf{L_{val}^k}$.ajouter(x);
    sinon
      /* Il y a une EP */
      $\mathbf{L_{loc}^k}$.ajouter(1);
      si $p_k^{[k,7]}(i,j) < 2^{7-k}$ **alors**
        $x \leftarrow -(2^{6-k} + 1)$;
      sinon
        $x \leftarrow 2^{6-k} + 1$;
      $\mathbf{L_{val}^k}$.ajouter(x);
retourner $\mathbf{L_{loc}^k}$ et $\mathbf{L_{val}^k}$;

Dans la fig. 5.4, nous représentons les différences entre chaque pixel et leur prédicteur associé selon le plan binaire considéré. Ces expérimentations sont obtenues en moyennant l'ensemble des valeurs mesurées sur les images de la base BOWS-2 [3], soit 10000 images de 512×512 pixels avec différentes propriétés statistiques. Pour le plan binaire d'indice k ($0 \leq k \leq 8$), la valeur de la différence entre un pixel et son prédicteur



(a) $k = 0$. (b) $k = 1$. (c) $k = 2$. (d) $k = 3$.



(e) $k = 4$. (f) $k = 5$. (g) $k = 6$. (h) $k = 7$.

FIGURE 5.4 – Différences entre chaque pixel et son prédicteur associé selon le plan binaire d'indice k (valeurs moyennes en utilisant les 10000 images de la base BOWS-2 [3]).

est comprise entre $-2^{8-k}+1$ et $2^{8-k}-1$. Cependant, quel que soit le plan binaire, ces valeurs sont souvent proches de zéro. De ce fait, la distribution de ces différences peut être approchée par une distribution Laplacienne. Comme ces valeurs sont très faibles, cela signifie que la quantité de pixels mal prédits n'est pas significative, même lorsqu'un prédicteur simple comme le MED est utilisé. De plus, en cas d'EP, la valeur à stocker dans $\mathbf{L}^{\mathbf{k}}_{\mathbf{val}}$ est peu importante, en particulier dans les plans LSB.

En fig. 5.5, nous illustrons cette analyse sur l'image de *Lena*. Sur la première ligne, nous montrons les résultats obtenus sur l'image entière (pixels codés sur 8 bits) et sur la deuxième, ceux obtenus en utilisant les sept plans LSB seulement (pixels codés sur 7 bits). Nous pouvons d'abord observer l'image originale et son histogramme associé. La carte des différences entre ses pixels et leurs prédicteurs (MED) et l'histogramme associé sont ensuite présentés. Quel que soit le nombre de plans binaires, la distribution des différences peut être considérée comme une distribution Laplacienne (avec différentes valeurs de la variance) centrée en zéro. Les faibles valeurs des différences sont dues aux fortes similarités entre les valeurs des pixels et les prédicteurs associés, et par extension, à la haute corrélation entre les valeurs des pixels et les valeurs de leurs voisins.

5.4.3 Adaptation réversible des plans binaires

Après la prise en compte des EP, la carte de localisation des EP $\mathbf{L}^{\mathbf{k}}_{\mathbf{loc}}$ et la liste des valeurs des EP $\mathbf{L}^{\mathbf{k}}_{\mathbf{val}}$ sont obtenues. Si l'IDC est possible, l'étape suivante consiste à adapter l'image en clair $I_k^{[k,7]}$ selon les valeurs des EP, pour pouvoir détecter leur emplacement pendant la phase de décodage. Notons que cette étape est fondamentale pour pouvoir reconstruire parfaitement l'image originale.

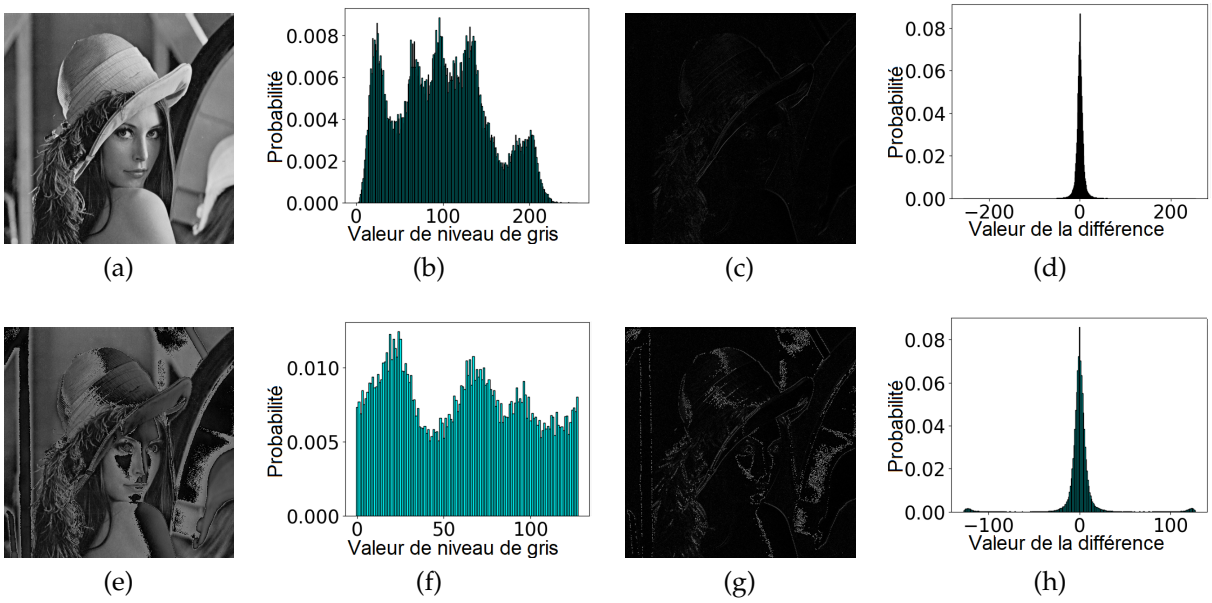


FIGURE 5.5 – Illustration de l'analyse des valeurs des différences : a) Image originale de *Lena* (pixels codés sur 8 bits), b) Histogramme associé aux valeurs de niveaux de gris de (a), c) Carte des différences entre chaque pixel et son prédicteur associé d'après (a), d) Histogramme des valeurs des différences (obtenu d'après (c)), e) Image composée des sept plans LSB de (a) (pixels codés sur 7 bits), f) Histogramme associé aux valeurs de niveaux de gris de (e), g) Carte des différences entre chaque pixel et son prédicteur associé d'après (e), h) Histogramme des valeurs des différences (obtenu d'après (g)).

L'algorithme 3 décrit les étapes nécessaires pour adapter l'image $I_k^{[k,7]}$. A l'aide de la carte de localisation des EP $\mathbf{L}_{\mathbf{loc}}^{\mathbf{k}}$, tous les pixels mal prédits $p_k^{[k,7]}(i,j)$ peuvent être identifiés. Ainsi, quand un pixel est identifié comme mal prédit, la première étape consiste à lire la valeur associée de l'EP dans $\mathbf{L}_{\mathbf{val}}^{\mathbf{k}}$. En ajoutant cette valeur de l'EP au pixel $p_k^{[k,7]}(i,j)$, le pixel adapté $p_{k+1}^{[k,7]}(i,j)$ est obtenu, comme décrit dans l'algorithme 3. Après cette modification, il existe une différence de 2^{6-k} ou $2^{6-k}+2^{7-k}$ entre la valeur du pixel adapté et son prédicteur, et de 2^{6-k} entre l'inverse de $p_{k+1}^{[k,7]}(i,j)$ et son prédicteur. Notons que, pendant le reconstruction de l'image originale, ces configurations spéciales peuvent être identifiées dans l'image adaptée $I_{k+1}^{[k,7]}$ et les valeurs originales peuvent être reconstruites à l'aide de $\mathbf{L}_{\mathbf{val}}^{\mathbf{k}}$ qui est insérée au début du plan binaire courant. Par ailleurs, $\mathbf{L}_{\mathbf{loc}}^{\mathbf{k}}$ n'est pas nécessaire à la reconstruction et ne doit pas être transmise comme document auxiliaire. Contrairement au pré-traitement de l'image dans l'approche IDCHC-CEP décrite dans le chapitre 4, l'adaptation des plans binaire est réalisée pour mettre en évidence l'emplacement des pixels mal prédits : son but n'est pas de les corriger. De plus, les valeurs des pixels originaux avant adaptation peuvent être retrouvées sans erreur et l'image reconstruite n'est pas une approximation, mais l'image originale elle-même.

Algorithme 3 : Adaptation réversible de l'image $I_k^{[k,7]}$.

Données : Image en clair $I_k^{[k,7]}$ de $m \times n$ pixels $p_k^{[k,7]}(i,j)$ codés sur $8-k$ bits, carte de localisation des EP $\mathbf{L_{loc}^k}$ et liste des valeurs des EP $\mathbf{L_{val}^k}$, pour le $k^{\text{ème}}$ plan binaire $I_k^{[k]}$
Résultat : Image adaptée en clair $I_{k+1}^{[k,7]}$ de $m \times n$ pixels $p_{k+1}^{[k,7]}(i,j)$ codés sur $8-k$ bits

```
index ← 0;
pour i ← 0 à m faire
    pour j ← 0 à n faire
        si L^k_loc[i × n + j] = 1 alors
            /* Il y a une EP                                   */
            si |L^k_val[index]| ≠ 2^(6−k) + 1 alors
                p^[k,7]_{k+1}(i,j) ← p^[k,7]_k(i,j) + L^k_val[index];
            index ← index + 1;
        sinon
            p^[k,7]_{k+1}(i,j) ← p^[k,7]_k(i,j);
retourner I^[k,7]_{k+1};
```

5.4.4 Chiffrement de l'image et IDC

La clé de chiffrement K_c est utilisée comme graine d'un GNPA pour obtenir une séquence pseudo-aléatoire de $m \times n$ bits $s(i,j)$. Dans le plan binaire courant $I_{k+1}^{[k]}$, *i.e.* le plan binaire MSB de l'image en clair $I_{k+1}^{[k,7]}$, un ou-exclusif entre chaque bit et le bit associé dans la séquence pseudo-aléatoire est réalisé pour générer un bit chiffré $p_c^{[k]}(i,j)$ du plan binaire chiffré $I_c^{[k]}$:

$$p_c^{[k]}(i,j) = s(i,j) \oplus p_{k+1}^{[k]}(i,j). \tag{5.3}$$

Après le chiffrement, si l'insertion du message est possible dans le plan binaire courant, la liste des valeurs des EP $\mathbf{L_{val}^k}$ est alors chiffrée avec la clé de chiffrement K_c et insérée par substitution des bits du plan binaire, à partir du troisième bit. En effet, comme indiqué précédemment, le premier bit est utilisé pour initialiser la prédiction et reste inchangé. Le deuxième bit sert à savoir si le plan suivant est marqué (1 si c'est le cas, 0 s'il est seulement chiffré). Il est également nécessaire d'insérer un drapeau FDL (Fin De Liste) pour indiquer au propriétaire du message caché la fin de $\mathbf{L_{val}^k}$. Ce drapeau peut être, par exemple, une séquence de huit bits consécutifs égaux à 1. A la fin du processus, le plan chiffré à marquer $I_{c\_ep}^{[k]}$ contenant $\mathbf{L_{val}^k}$ est obtenu.

Pour réaliser l'insertion du message, le propriétaire du message caché commence par repérer s'il peut être inséré dans le plan binaire courant $I_{c\_ep}^{[k]}$ en examinant la valeur du deuxième bit du plan binaire précédent (notons que nous supposons que le plan MSB est toujours marqué). La clé d'insertion K_m est alors utilisée pour chiffrer le message secret. De cette façon, il n'est pas possible de détecter sa présence après son insertion. En suivant l'ordre des lignes de balayage, le drapeau FDL est détecté et, à sa suite, tous les bits restants $p_{c\_ep}^{[k]}(i,j)$ sont utilisés pour l'IDC. Ainsi, les bits disponibles sont substitués par les bits b^l du message secret (avec $l < L$, le nombre de bits pouvant être marqués). Les bits $p_{c\_m}^{[k]}(i,j)$ de l'image chiffrée marquée $I_{c\_m}^{[k]}$ sont alors obtenus :

$$p_{c\_m}^{[k]}(i,j) = b^l. \tag{5.4}$$

La fig. 5.6 illustre la configuration d'un plan chiffré marqué $I_{c\_m}^{[k]}$ à la fin de la phase d'encodage. Le premier bit n'est pas marqué car sa valeur sert à initialiser le processus de prédiction. Le deuxième bit indique si le plan binaire suivant est marqué. A partir du troisième bit, la liste des valeurs des EP est stockée. Cette liste est suivie d'un drapeau FDL indiquant sa fin et donc le début du message secret inséré. Enfin, tous les bits restants sont ceux du message secret.

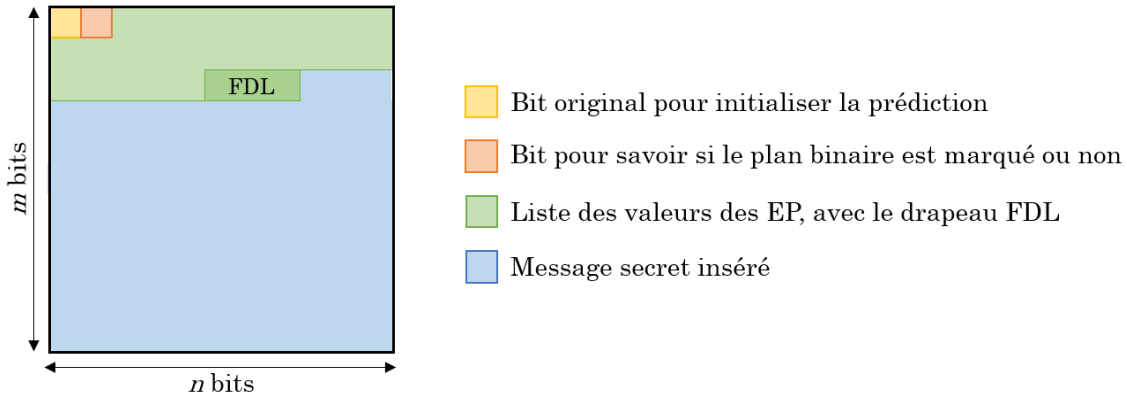


FIGURE 5.6 – Composition de chaque plan binaire chiffré marqué à la fin de la phase d'encodage.

5.4.5 Extraction du message secret et reconstruction de l'image

Lors de la phase de reconstruction, comme notre méthode est séparative, trois situations sont considérées, suivant la clé possédée.

Dans le cas où celui qui reçoit les données connaît seulement la clé d'insertion K_m, il doit parcourir l'ensemble des plans binaires de l'image chiffrée marquée I_{c_m}, en commençant par le plan MSB. Tout d'abord, il sait si le plan binaire courant $I_{c\_m}^{[k]}$ est marqué, selon la valeur du deuxième bit du plan binaire précédent. Notons que dans la section 5.4.4, nous avons supposé que le plan MSB était toujours marqué. Si tel est le cas, les premiers bits sont ceux de la liste des valeurs des EP $\mathbf{L}_{\mathbf{val}}^{\mathbf{k}}$ et ne doivent pas être extraits en tant que bits du message secret. Ainsi, la fin de $\mathbf{L}_{\mathbf{val}}^{\mathbf{k}}$, indiquée par un drapeau FDL, doit être identifiée. Après le drapeau FDL, les bits suivants du plan binaire peuvent être extraits aveuglément pour obtenir une partie du message secret. Dès qu'un plan binaire indique que le suivant n'est pas marqué, les plans binaires restants ne contiennent pas de bits du message secret. Par conséquent, l'extraction du message est terminée et les données extraites doivent être concaténées, depuis celles extraites dans le plan MSB jusqu'à celles du dernier plan marqué. Finalement, le message secret chiffré est obtenu et peut être déchiffré à l'aide de la clé d'insertion K_m. Notons que l'extraction du message s'effectue sans perte et sans erreur.

Si celui qui reçoit les données connaît seulement la clé de chiffrement, il peut reconstruire l'image originale sans altération grâce à l'adaptation réversible réalisée par

Algorithme 4 : Algorithme de reconstruction de l'image.

Données : Liste des valeurs des EP $\mathbf{L_{val}^k}$, image $I_{k+1}^{[k+1,7]}$ de $m \times n$ pixels $p_{k+1}^{[k+1,7]}(i,j)$ codés sur $7-k$ bits et plan binaire chiffré marqué $I_{c\_m}^{[k]}$

Résultat : Image en clair $I_k^{[k,7]}$ de $m \times n$ pixels $p_k^{[k,7]}(i,j)$ codés sur $8-k$ bits

```
/* Initialisation de la prédiction                                  */
/* D(·) est la fonction de déchiffrement                            */
```

$index \leftarrow 0$;
pour $i \leftarrow 0$ **à** m **faire**
  pour $j \leftarrow 0$ **à** n **faire**
    si $(i,j) = (0,0)$ **alors**
      $p_k^{[k,7]}(0,0) = \mathcal{D}(p_{c\_m}^{[k]}(0,0)) \times 2^{7-k} + p_{k+1}^{[k+1,7]}(0,0)$;
    sinon
      $p^0(i,j) \leftarrow p_{k+1}^{[k+1,7]}(i,j) + 0$;
      $p^1(i,j) \leftarrow p_{k+1}^{[k+1,7]}(i,j) + 2^{7-k}$;
      si $i = 0$ **ou** $j = 0$ **alors**
        Prédiction unidirectionnelle;
      sinon
        $pred(i,j) \leftarrow \text{MED}(p_k^{[k,7]}(i,j))$;
      $\Delta^0 \leftarrow |pred(i,j) - p^0(i,j)|$;
      $\Delta^1 \leftarrow \left|pred(i,j) - p^1(i,j)\right|$;
      si $(\Delta^0 \neq 2^{6-k}$ **ou** $\Delta^1 \neq 2^{6-k} + 2^{7-k})$
      et $(\Delta^0 \neq 2^{6-k} + 2^{7-k}$ **ou** $\Delta^1 \neq 2^{6-k})$
      et $(\Delta^0 \neq 2^{6-k}$ **ou** $\Delta^1 = 2^{6-k})$
      et $(\Delta^0 = 2^{6-k}$ **ou** $\Delta^1 \neq 2^{6-k})$ **alors**
        si $\Delta^0 < \Delta^1$ **alors**
          $p_k^{[k,7]}(i,j) \leftarrow p^0(i,j)$;
        sinon
          $p_k^{[k,7]}(i,j) \leftarrow p^1(i,j)$;
      sinon
        si $|\mathbf{L_{val}^k}[index]| \neq 2^{6-k} + 1$ **alors**
          $p^0(i,j) \leftarrow (p^0(i,j) - \mathbf{L_{val}^k}[index]) \mod 2^{8-k}$;
          $p^1(i,j) \leftarrow (p^1(i,j) - \mathbf{L_{val}^k}[index]) \mod 2^{8-k}$;
          $\Delta^0 \leftarrow |pred(i,j) - p^0(i,j)|$;
          $\Delta^1 \leftarrow \left|pred(i,j) - p^1(i,j)\right|$;
          si $\Delta^0 > \Delta^1$ **alors**
            $p_k^{[k,7]}(i,j) \leftarrow p^0(i,j)$;
          sinon
            $p_k^{[k,7]}(i,j) \leftarrow p^1(i,j)$;
        sinon
          si $\mathbf{L_{val}^k}[index] = -(2^{6-k} + 1)$ **alors**
            $p_k^{[k,7]}(i,j) \leftarrow p^0(i,j)$;
          sinon
            $p_k^{[k,7]}(i,j) \leftarrow p^1(i,j)$;
        $index \leftarrow index + 1$;

le propriétaire de l'image. Comme lors de la phase d'encodage, les huit plans binaires de l'image chiffrée marquée I_{c_m} sont parcourus récursivement, mais du plan LSB au plan MSB car les plans binaires d'indice $(k+1)$ à 7 en clair sont nécessaires pour la prédiction du $k^{\text{ème}}$ plan binaire. Tout d'abord, il est nécessaire de savoir si le plan binaire courant $I_{c\_m}^{[k]}$ est marqué ou bien simplement chiffré. Dans ce dernier cas, la séquence pseudo-aléatoire associée est générée en utilisant la clé de chiffrement K_c. Ainsi, le déchiffrement est réalisé avec l'opération ou-exclusif. Dans le cas contraire, si le plan courant est marqué, tous les bits chiffrés ont été remplacés par les bits de la liste des valeurs des EP $\mathbf{L_{val}^k}$ et ceux d'un message secret, à l'exception des deux premiers bits. Par conséquent, au lieu de réaliser un simple déchiffrement, une prédiction est nécessaire à partir du deuxième bit. En effet, seulement le premier bit est directement déchiffré et $\mathbf{L_{val}^k}$ est extraite du troisième bit jusqu'au drapeau FDL. Les valeurs des bits doivent alors être prédites, à l'aide du premier bit initialisant la prédiction et de $\mathbf{L_{val}^k}$. De plus, pendant le processus d'encodage décrit dans la section 5.4.3, notons que non seulement les valeurs du plan binaire courant sont modifiées pour rendre possible la prédiction, mais aussi celles des autres plans binaires moins significatifs. Ainsi, le processus récursif de prédiction pendant la phase de décodage prend en entrée : $\mathbf{L_{val}^k}$, l'image adaptée $I_{k+1}^{[k+1,7]}$ et le plan binaire chiffré marqué $I_{c\_m}^{[k]}$. En sortie, il renvoie l'image avant adaptation $I_k^{[k,7]}$. Ces données en entrée et en sortie ainsi que les différentes étapes du processus récursif sont présentées dans l'algorithme 4. L'image composée de $I_{c\_m}^{[k]}$ et de $I_{k+1}^{[k+1,7]}$ $(I_{c\_m}^{[k]} + I_{k+1}^{[k+1,7]})$ est parcourue dans l'ordre des lignes de balayage pour prédire les valeurs associées des pixels avant adaptation $p_k^{[k,7]}(i,j)$. Deux valeurs sont possibles pour chaque pixel courant adapté : $p^0(i,j) = p_{k+1}^{[k+1,7]}(i,j) + 0$, quand le MSB $p_{k+1}^{[k]}(i,j)$ est égal à 0 et $p^1(i,j) = p_{k+1}^{[k+1,7]}(i,j) + 2^{7-k}$, quand $p_{k+1}^{[k]}(i,j)$ est égal à 1. La valeur du MSB est la seule pouvant être erronée lors de cette étape. Le prédicteur $pred(i,j)$ est calculé comme étant le MED appliqué à $p_k^{[k,7]}(i,j)$, similairement au processus décrit dans la section 5.4.2. Notons que les pixels voisins de $p_k^{[k,7]}(i,j)$ avant l'adaptation servent à la prédiction et sont déjà reconstruits. Δ^0 et Δ^1 sont alors évaluées comme étant les différences absolues entre les deux valeurs possibles du pixel et leur prédicteur $pred(i,j)$. Selon les valeurs de Δ^0 et de Δ^1, différents cas doivent être considérés :

— Si aucune configuration spéciale n'est identifiée, il n'existe pas d'EP. La valeur du pixel avant adaptation $p_k^{[k,7]}(i,j)$, égale à la valeur du pixel après adaptation $p_{k+1}^{[k,7]}(i,j)$, est obtenue par :

$$\begin{aligned} p_k^{[k,7]}(i,j) &= p_{k+1}^{[k,7]}(i,j) \\ &= \begin{cases} p^0(i,j) & \text{si } \Delta^0 < \Delta^1, \\ p^1(i,j) & \text{sinon.} \end{cases} \end{aligned} \tag{5.5}$$

— L'emplacement courant correspond à une EP si une configuration spéciale est mise en évidence, *i.e.* $\Delta^0 = \Delta^1 = 2^{6-k}$ ou $(\Delta^0 = 2^{6-k}$ et $\Delta^1 = 2^{6-k} + 2^{7-k})$ ou $(\Delta^0 = 2^{6-k} + 2^{7-k}$ et $\Delta^1 = 2^{6-k})$. Dans ce cas, cela signifie que le pixel courant a été adapté pendant la phase d'encodage. Cependant, la valeur du pixel avant adaptation peut être reconstruite en utilisant $\mathbf{L_{val}^k}$ et en effectuant une prédiction

inverse. En effet, si une telle configuration est rencontrée, la valeur associée de l'EP dans $\mathbf{L}^{\mathbf{k}}_{\mathbf{val}}$ est extraite :

— Si la valeur absolue de l'EP est différente de $2^{6-k}+1$, les deux valeurs possibles pour le pixel avant adaptation $p_k^{[k,7]}(i,j)$ sont obtenues par soustraction de la valeur de l'EP $p^0(i,j)$ avec $p^1(i,j)$. Notons que ces nouvelles valeurs correspondent à $p_k^{[k,7]}(i,j)$ et $inv(i,j)$ pendant la phase de détection et d'évaluation des EP décrite en section 5.4.2. Dans le but de déterminer la valeur correcte, les différences absolues Δ^0 et Δ^1 sont évaluées une nouvelle fois, comme décrit dans l'algorithme 4. En se basant sur le fait que la valeur du pixel avant adaptation est concerné par une EP, la valeur de $p_k^{[k,7]}(i,j)$ est déterminée par la valeur la plus éloignée du prédicteur $pred(i,j)$:

$$p_k^{[k,7]}(i,j) = \begin{cases} p^0(i,j) & \text{si } \Delta^0 > \Delta^1, \\ p^1(i,j) & \text{sinon.} \end{cases} \tag{5.6}$$

— Le cas échéant, si la valeur absolue de l'EP est égale à $2^{6-k}+1$, une configuration spéciale est détectée. En effet, cela correspond au cas où la valeur correcte du pixel avant adaptation $p_k^{[k,7]}(i,j)$ ne peut pas être prédite. Le signe de la valeur de l'EP est donc utilisé pour connaître la valeur correcte :

$$p_k^{[k,7]}(i,j) = \begin{cases} p^0(i,j) & \text{si la valeur de l'EP est négative,} \\ p^1(i,j) & \text{sinon.} \end{cases} \tag{5.7}$$

Dans le cas où les clés d'insertion et de chiffrement sont connues, les bits du message secret sont d'abord extraits, puis l'image originale est parfaitement reconstruite. Dans cette situation, l'extraction du message et la reconstruction de l'image s'effectuent sans erreur.

5.5 Résultats expérimentaux

Dans cette section, nous présentons les résultats expérimentaux obtenus en appliquant notre méthode récursive d'IDCDC réversible et haute capacité. La section 5.5.1 illustre la méthode proposée, en présentant un exemple détaillé. Dans la section 5.5.2, une analyse des performances en termes de charge utile est réalisée. Ces tests sont réalisés sur l'intégralité de la base BOWS-2 composée de 10000 images en niveaux de gris [3]. Dans la section 5.5.3, une analyse statistique de la méthode proposée est présentée. Enfin, en section 5.5.4, notre méthode est comparée avec les algorithmes récents de l'état-de-l'art.

5.5.1 Exemple complet de la méthode proposée

Tout d'abord, nous illustrons la méthode proposée au travers d'un d'exemple détaillé. En fig. 5.7, nous avons appliqué notre méthode à l'image originale *Dolls I* de taille 666×1000 pixels codés sur 256 niveaux de gris (fig. 5.7.a). Pendant la phase d'encodage, les plans binaires sont traités récursivement du plus significatif ($I^{[0]}$) au moins significatif

($I^{[7]}$). Ainsi, la première étape consiste à prendre en compte les EP associées au premier plan binaire $I_0^{[0]}$ de l'image originale. Une carte de localisation des EP, illustrée en fig. 5.7.b, et une liste des valeurs des EP sont calculées. Notons que le nombre d'EP (1628) est peu important et représente seulement $0,2\%$ du nombre total de bits. La taille de la liste des values des erreurs de EP est alors analysée pour savoir si elle peut être insérée dans le plan binaire courant. Si l'insertion de cette liste est possible, l'image $I_0^{[0,7]}$ est traitée pour permettre la détection des erreurs de EP pendant la phase de reconstruction de l'image. En fig. 5.7.c, l'image adaptée $I_1^{[0,7]}$ est alors obtenue. Le plan MSB est alors traité séparément. Celui-ci est chiffré puis marqué par les valeurs des EP par le propriétaire de l'image. Une partie du message secret est alors insérée à la suite de la liste des valeurs des EP, par substitution des bits durant la phase d'IDC. La fig. 5.7.d représente l'image à la fin du traitement du premier plan binaire. Elle est composée du plan MSB chiffré marqué $I_{c\_m}^{[0]}$ et des 7 plans LSB de l'image adaptée $I_1^{[1,7]}$ toujours en clair. Après le traitement du premier plan binaire, la charge utile totale de l'image $I_{c\_m}^{[0]} + I_1^{[1,7]}$ est égale à $0,9804$ bpp, ce qui signifie que seulement $0,0196$ bpp sont utilisés pour stocker les valeurs des EP dans le premier plan binaire. Le deuxième plan binaire $I_1^{[1]}$ de l'image adaptée est alors analysé et traité. La fig. 5.7.e correspond à la carte de localisation des EP calculée pendant la phase de prise en compte des EP associée à $I_1^{[1]}$. Notons que le nombre d'EP est supérieur à celui du premier plan binaire (29324, soit $4,4\%$ du nombre total de bits). Cela s'explique par le fait que moins un plan binaire est significatif, moins ses bits sont corrélés. Cependant, la liste des valeurs des EP est de taille suffisamment faible pour réaliser son insertion dans le plan binaire courant. $I_1^{[1,7]}$ est alors adaptée pour rendre possible la détection des EP et $I_2^{[1,7]}$ est obtenue (fig. 5.7.f). La fig. 5.7.g correspond à l'image obtenue après les phases de chiffrement et d'IDC dans le plan binaire courant $I_2^{[1]}$. En effet, elle est composée de deux plans binaires chiffrés marqués $I_{c\_m}^{[0,1]}$ et des six plans moins significatifs $I_2^{[2,7]}$. La charge utile dans l'image $I_{c\_m}^{[0,1]} + I_2^{[2,7]}$ est égale à $1,6722$ bpp, ce qui indique un gain de $0,6918$ bpp en utilisant le second plan MSB. Les mêmes étapes sont répétées sur le troisième plan binaire $I_2^{[2]}$. La fig. 5.7.h est la carte de localisation des EP. Cette fois-ci, 77094 bits ($11,58\%$) sont associés à des EP, mais l'insertion de la liste des valeurs des EP est tout de même encore possible. L'image $I_2^{[2,7]}$ est adaptée, selon la phase de prise en compte des EP associée au troisième plan MSB. L'image adaptée $I_3^{[2,7]}$ est illustrée en fig. 5.7.i. De plus, comme présenté en fig. 5.7.j, la charge utile dans l'image $I_{c\_m}^{[0,2]} + I_3^{[3,7]}$ est égale à $1,9777$ bpp, ce qui signifie que l'utilisation du troisième plan binaire pour l'IDC permet une augmentation de $0,3055$ bpp. En fig. 5.7.k, la carte de localisation des EP associée au quatrième plan binaire $I_3^{[3]}$ est illustrée. A cause d'un nombre important d'EP (144014, soit $21,63\%$ des bits), la taille de la liste des valeurs des EP est trop grande pour que l'insertion soit réalisée dans le plan binaire courant. Le traitement récursif des plans binaires est alors interrompu et les plans binaires courant et restants, toujours en clair, sont seulement chiffrés. L'image obtenue à la fin de la phase d'encodage est représentée en fig. 5.7.l. La valeur finale de la charge utile est donc égale à $1,9777$ bpp, après une IDC dans les trois plans MSB. Pendant la phase de décodage, comme présenté en fig. 5.7.m, l'image originale I est parfaitement reconstruite, en traitant chaque plan binaire du moins significatif au plus significatif. En effet, les plans LSB sont nécessaires

pour prédire les plans MSB. Par ailleurs, la valeur du PSNR entre l'image originale et l'image reconstruite tend vers l'infini et la valeur du SSIM est égale à 1.

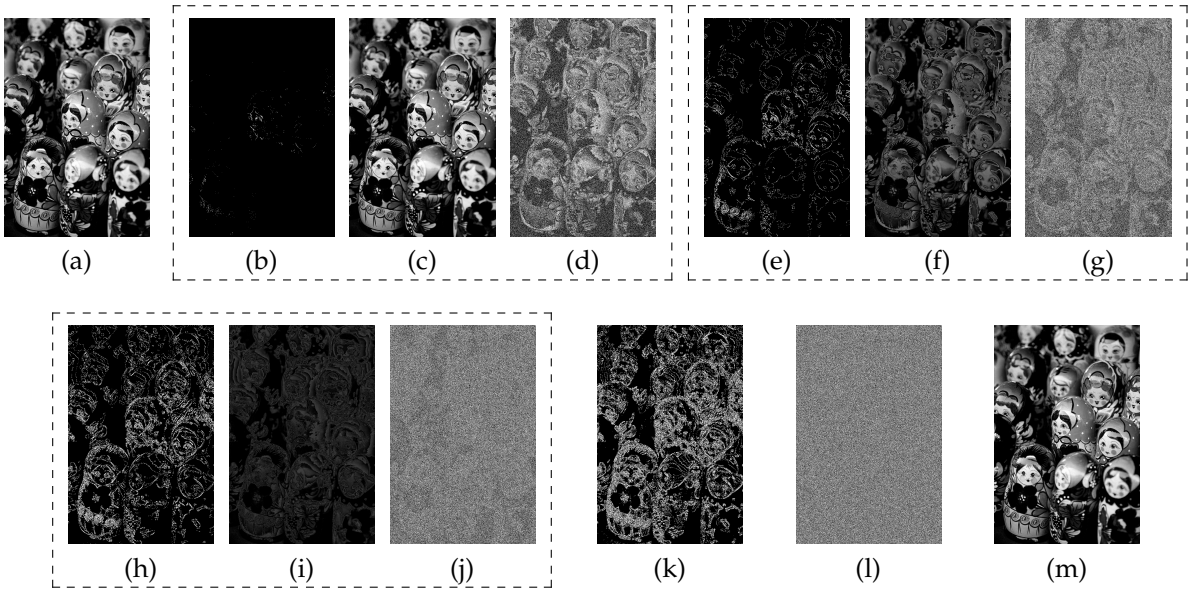


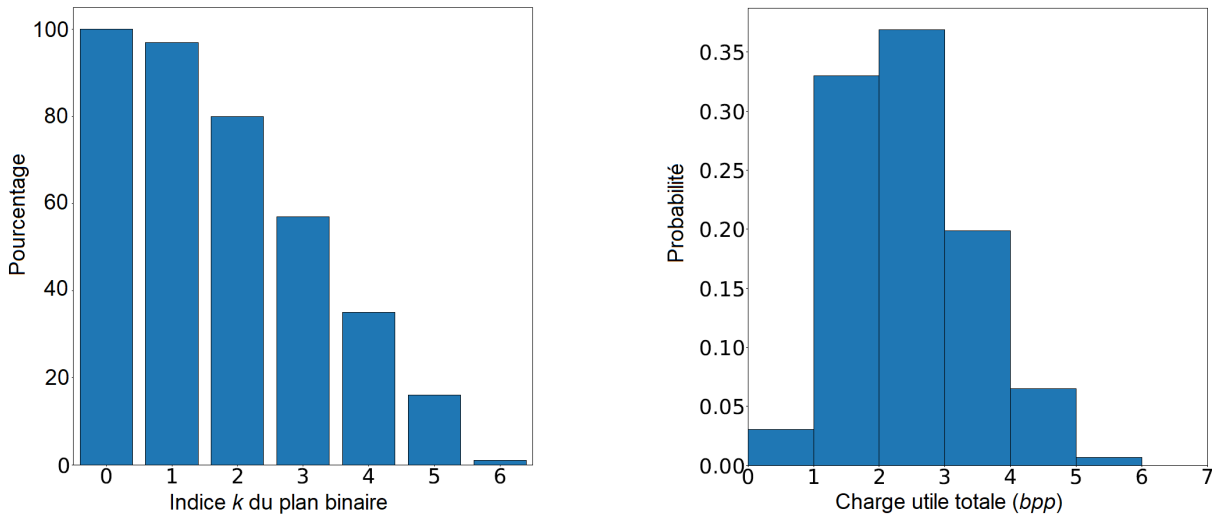
FIGURE 5.7 – Illustration de la méthode récursive proposée : a) Image originale *Dolls* I de 666×1000 pixels, b) Carte de localisation des EP associée au 1^{er} plan binaire $I_0^{[0]}$ (plan MSB), nombre d'EP = 1628 $(0,2\%)$, c) Image adaptée $I_1^{[0,7]}$ après prise en compte des EP, d) Image $I_{c\_m}^{[0]} + I_1^{[1,7]}$ composée du 1^{er} plan binaire chiffré marqué et des 7 plans LSB de (c), charge utile = $0,9804$ *bpp*, e) Carte de localisation des EP associée au $2^{\text{ème}}$ plan binaire $I_1^{[1]}$, nombre d'EP = 29324 $(4,4\%)$, f) Image adaptée $I_2^{[1,7]}$ après prise en compte des EP, g) Image $I_{c\_m}^{[0,1]} + I_2^{[2,7]}$ composée des 1^{er} et $2^{\text{ème}}$ plans binaires chiffrés marqués et des 6 plans LSB de (f), charge utile = $1,6722$ *bpp* (+$0,6918$ *bpp*), h) Carte de localisation des EP associée au $3^{\text{ème}}$ plan binaire $I_2^{[2]}$, nombre d'EP = 77094 $(11,58\%)$, i) Image adaptée $I_3^{[2,7]}$ après prise en compte des EP, j) Image $I_{c\_m}^{[0,2]} + I_3^{[3,7]}$ composée des 1^{er}, $2^{\text{ème}}$ et $3^{\text{ème}}$ plans binaires chiffrés marqués et des 5 plans binaires LSB de (i), charge utile = $1,9777$ *bpp* (+$0,3055$ *bpp*), k) Carte de localisation des EP associée au $4^{\text{ème}}$ plan binaire $I_3^{[3]}$, nombre d'EP = 144014 $(21,63\%)$, l) Image I_{c_m} après chiffrement de tous les plans binaires et IDC dans les 1^{er}, $2^{\text{ème}}$ et $3^{\text{ème}}$ plans binaires, charge utile au total = $1,9777$ *bpp*, m) Image originale reconstruite I, PSNR $\rightarrow +\infty$, SSIM = 1.

5.5.2 Résultats obtenus sur une grande base de données

Dans la fig. 5.8.a, nous présentons la répartition des images de la base BOWS-2 [3] suivant la possibilité de réaliser l'IDC dans chaque plan binaire. Tout d'abord, nous pouvons voir que, dans tous les cas, l'IDC est possible dans le premier plan binaire (plan MSB, $k = 0$). Cela s'explique par la très forte corrélation entre les MSB voisins dans chaque image. Jusqu'au troisième plan binaire, cette corrélation demeure relativement importante. Ainsi, l'IDC peut être réalisée dans le deuxième plan MSB ($k = 1$) dans 97% des images et dans le troisième plan MSB ($k = 2$) dans 80% des images. Après le troisième plan binaire, nous observons une diminution de la quantité d'images où les

plans binaires suivants peuvent être marqués. En effet, l'IDC peut être réalisée jusqu'au quatrième plan binaire ($k = 3$) dans 57% des images, jusqu'au cinquième plan binaire ($k = 4$) dans 35% des images, et jusqu'au sixième plan binaire ($k = 5$) pour 16% des images. Notons que très peu d'images ($0,1\%$) peuvent être marquées du plan MSB au dernier plan prédictible ($k = 6$, correspondant au deuxième plan LSB). Ces images ne sont pas ou très peu texturées et semblent presque homogènes, ce qui explique la prédictibilité de chaque plan binaire. Notons que le plan LSB ($k = 7$) n'est jamais marqué car ses valeurs ne peuvent pas être prédites avec l'aide des autres plans binaires.

La fig. 5.8.b représente la distribution de la base d'images selon la valeur de la charge utile obtenue. Les résultats sont cohérents avec ceux de la fig. 5.8.a. En effet, seulement 3% des images ont une charge utile inférieure à 1 *bpp*. Ces images sont les plus texturées de la base. Comme elles comportent de nombreux contours, la corrélation entre les pixels voisins est faible. Pour cette raison, seulement le plan MSB de ces images est généralement marqué et la charge utile maximale est alors d'1 *bpp*. Comme expliqué précédemment, la plupart des images sont marquées jusqu'au troisième plan binaire. Dans ce cas, la valeur maximale de la charge utile est de 3 *bpp*. Par conséquent, en fig. 5.8.b, nous constatons que la valeur de la charge utile est entre 1 *bpp* et 3 *bpp* pour 70% des images. Cela signifie également que, pour 27% des images de la base, la charge utile est très haute et supérieure à 3 *bpp*. Pour 20% des images, elle est entre 3 *bpp* et 4 *bpp*. Dans les 7% d'images restantes, elle est supérieure à 4 *bpp*, lorsque l'IDC peut être réalisée dans les plans LSB, parfois jusqu'au septième plan binaire. La valeur maximale de la charge utile est alors de 7 *bpp*. Néanmoins, notons que moins d'1% des images ont une charge utile entre 6 *bpp* et 7 *bpp*. Cela s'explique par le fait que, même si l'IDC est possible, la taille de la liste des valeurs des EP est grande et la quantité de bits insérés est alors relativement faible dans les plans LSB.



(a) (b)

FIGURE 5.8 – a) Répartition des images de la base BOWS-2 [3] suivant la possibilité d'insérer des données cachées dans le plan MSB d'indice k, b) Distribution des images de la base BOWS-2 [3] selon la charge utile.

De plus, les tableaux 5.1 et 5.2 présentent respectivement le pourcentage d'EP et la valeur de la charge utile mesurés dans chaque image de la base, selon la valeur k du plan binaire considéré. Notons que nous considérons seulement les images pour lesquelles l'IDC est possible dans le plan binaire précédent d'indice $k-1$ pour le calcul du nombre d'EP et le plan binaire courant d'indice k pour l'évaluation de la charge utile. Les résultats sont indiqués en termes de quartiles (Q1, médiane et Q3) et de valeur moyenne. Tout d'abord, nous pouvons voir que le nombre d'EP est très faible pour le plan MSB et proche de 0%. Par conséquent, la charge utile dans ce plan binaire est très haute et proche de la valeur maximale d'1 *bpp*. Le deuxième plan binaire est aussi hautement prédictible. En effet, la quantité d'EP est inférieure à 10% dans tous les cas. Ainsi, la valeur de la charge utile est très haute : elle est supérieure à $0,6105$ *bpp* pour 75% des images et même supérieure à $0,8881$ *bpp* pour 25% des images. Si le troisième plan binaire est considéré, le nombre d'EP est légèrement plus important et de l'ordre de 10%. De ce fait, la valeur de la charge utile diminue mais reste cependant assez haute : pour plus de 50% des images, elle est supérieure à $0,5$ *bpp*. Dans le quatrième plan binaire, le nombre d'EP est supérieur à 15% en moyenne. La valeur médiane associée de la charge utile est alors de $0,3574$ *bpp*. Cependant, pour 25% des images, elle est supérieure à $0,5$ *bpp*. Rappelons que moins de 50% des images de la base peuvent être marquées jusqu'au quatrième plan binaire. Dans le cinquième plan binaire, la quantité d'EP est supérieure à 20% dans la plupart des cas et la valeur associée de la charge utile est généralement inférieure à $0,4$ *bpp*. Notons que dans le sixième plan binaire, comme environ 30% des bits sont concernés par une EP, la charge utile est de $0,2$ *bpp* pour de nombreuses images. Dans le septième plan binaire, le nombre d'EP est très important (de l'ordre de 60%). Pour cette raison, ce plan binaire n'est généralement pas marqué. Néanmoins, $0,1\%$ des images de la base est très homogène. La liste des valeurs des EP est alors de petite taille, d'autant plus que seulement deux bits sont utilisés pour coder chaque EP (dans le cas où la liste n'est pas compressée). Ainsi, l'IDC est possible et dans ces cas particuliers, la charge utile peut être élevée, comme montré par une valeur médiane égale à $0,3875$ *bpp* et une moyenne de $0,4338$ *bpp*. Enfin, dans la dernière colonne du tableau 5.2, nous pouvons observer les valeurs finales de la charge utile, après l'IDC dans tous les plans binaires possibles. Les résultats attestent d'une très haute valeur de la charge utile pour toutes les images de la base. En effet, 75% des images ont une charge utile d'au moins $1,7224$ *bpp*. De plus, la valeur médiane est de $2,3209$ *bpp* et la moyenne est égale à $2,4586$ *bpp*. Par ailleurs, pour 25% des images, nous obtenons une très haute valeur de la charge utile puisqu'elle est supérieure à $3,0759$ *bpp*.

% d'EP	$k=0$	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$	$k=6$
Q1 (25%)	0,0092	1,6373	5,4136	10,6075	17,2287	28,7792	57,2205
Médiane (50%)	0,0469	3,2806	9,0668	15,6956	22,7856	33,9851	60,8387
Q3 (75%)	0,1545	5,8624	13,8999	20,9782	27,4303	38,2093	63,4789
Moyenne	0,1507	4,3247	10,3718	15,9634	22,1918	33,2427	59,4856

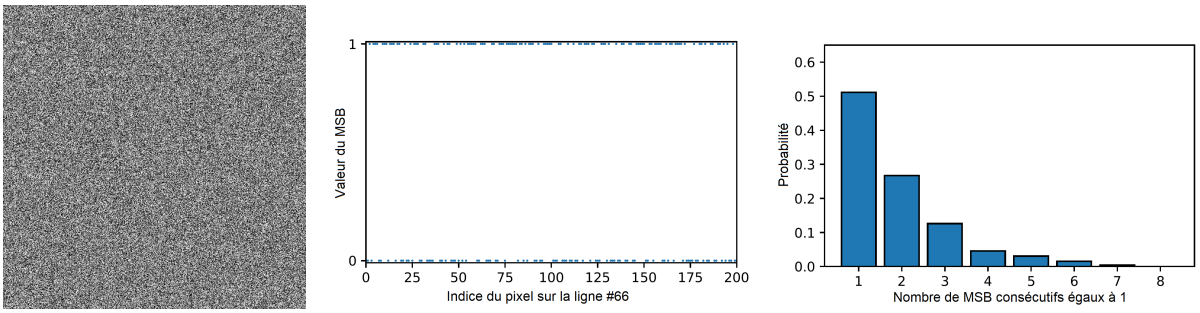
TABLE 5.1 – Pourcentage d'EP dans les images de la base BOWS-2 [3], selon le $k^{\text{ème}}$ plan binaire considéré, avec $0 \leq k \leq 6$.

Charge utile (*bpp*)	$k=0$	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$	$k=6$	$k=0$ à 6
Q1 (25%)	0,9876	0,6105	0,3315	0,1834	0,1181	0,0722	0,3711	1,7224
Médiane (50%)	0,9962	0,7770	0,5289	0,3574	0,2564	0,1518	0,3875	2,3209
Q3 (75%)	0,9993	0,8881	0,7089	0,5495	0,4233	0,2734	0,4536	3,0759
Moyenne	0,9879	0,7221	0,5146	0,3765	0,2867	0,1913	0,4338	2,4586

TABLE 5.2 – Mesures de la charge utile (en *bpp*) sur les images de la base BOWS-2 [3], selon le $k^{\text{ème}}$ plan binaire considéré, avec $0 \leq k \leq 6$.

5.5.3 Analyse statistique

Dans la fig. 5.9, nous évaluons le niveau de sécurité visuelle d'une image chiffrée marquée obtenue avec notre méthode d'IDCDC réversible. La fig. 5.9.a est l'image chiffrée marquée associée à l'image originale *Baboon*. Notons que nous avons choisi d'évaluer notre méthode sur cette image car, étant très texturée, elle est concernée par un grand nombre d'EP. De ce fait, il est intéressant d'observer si la prise en compte de ces EP a un impact visuel dans le domaine chiffré. Bien que la liste des valeurs des EP et le message secret sont insérés dans le domaine chiffré, aucun artefact visuel ne peut être distingué. Dans le but de réaliser notre analyse, nous considérons seulement une ligne de l'image chiffrée marquée. Par exemple, en fig. 5.9.b, les valeurs des MSB des 200 premiers pixels de la ligne #66 de l'image présentée en fig. 5.9.a sont illustrées. Nous pouvons constater qu'il existe peu de séquences de bits de même valeur. Ces résultats sont cohérents avec ceux présentés en fig. 5.9.c. Dans cette dernière, la distribution des longueurs des séquences de MSB consécutifs égaux à 1 dans toute l'image est représentée. Notons que les propriétés statistiques d'une image chiffrée marquée doivent suivre une loi géométrique de paramètre $p = 0,5$. Cela est le cas pour les résultats obtenus avec notre méthode, comme illustré en fig. 5.9.c.



(a) (b) (c)

FIGURE 5.9 – Évaluation du niveau de sécurité visuelle de la méthode proposée : a) Image chiffrée marquée associée à *Baboon* obtenue avec la méthode proposée, b) Valeurs des MSB sur les 200 premiers pixels de la ligne #66 de l'image présentée en (a), c) Distribution des séquences de MSB consécutifs égaux à 1 selon leur longueur sur l'intégralité de l'image (a).

Le tableau 5.3 et la fig. 5.10 présentent une analyse statistique comparant l'image originale *Baboon* et l'image chiffrée marquée associée illustrée en fig. 5.9.a. Les métriques statistiques utilisées sont celles classiquement utilisées pour l'évaluation des méthodes

d'IDCDC et décrites dans le chapitre 3. Comme présenté dans la fig. 5.10.a et dans la première ligne du tableau 5.3, la corrélation entre les pixels voisins est très importante dans l'image originale *Baboon*. Dans les directions horizontale et verticale, les valeurs sont proches de 1 ($0,8611$ et $0,7666$). En revanche, comme illustré dans la fig. 5.10.c et dans la deuxième ligne du tableau 5.3, cette corrélation est très faible dans l'image chiffrée marquée. En effet, les valeurs des pixels voisins sont très différentes : dans les directions horizontale et verticale, la corrélation est proche de 0 ($0,0005$ et $0,0007$). Les fig. 5.10.b et fig. 5.10.d correspondent respectivement aux histogrammes des pixels de l'image originale et de ceux de l'image chiffrée marquée. Contrairement à la distribution des pixels de l'image originale, celle qui est associée à l'image chiffrée marquée peut être approchée par une distribution uniforme. Cela signifie qu'aucune information statistique sur le contenu de l'image en clair ne peut être obtenue d'après l'histogramme de l'image chiffrée marquée. Ces deux histogrammes sont cohérents avec les valeurs de l'entropie mesurées. Pour l'image chiffrée marquée, l'entropie est proche de la valeur maximale de 8 *bpp* ($7,9994$ *bpp*), alors qu'elle est égale à $7,4744$ *bpp* dans l'image originale. Le caractère uniforme de la distribution des pixels de l'image chiffrée marquée est également attesté par le test χ^2. Pour ce test , nous avons $L = 256$ niveaux de gris possibles, soit $L - 1 = 255$ degrés de liberté et nous considérons un risque $\alpha = 0,05$. La table du χ^2 indique que $\chi^2(255, 0.05)$ est égal à $293,25$. De plus, le score χ^2 obtenu est très haut pour l'image originale ($142,81 \times 10^3$) et bien inférieur pour l'image chiffrée marquée ($228,65$). Les valeurs p associées sont respectivement 0 et $0,8810$. Comme attendu, d'après les résultats obtenus, la distribution de l'image originale n'est pas uniforme. En revanche, pour l'image chiffrée marquée, le score est inférieur à $\chi^2(255, 0,05)$ et la valeur-p est supérieure à $0,1$. Cela signifie qu'il n'y a pas de présomption contre l'hypothèse nulle : la distribution des pixels de l'image chiffrée marquée semble être approchée par une distribution uniforme. Ainsi, l'image chiffrée marquée n'est pas vulnérable aux attaques statistiques basées sur l'analyse d'histogramme. La valeur du NPCR est très haute et proche de la valeur maximale de 100% ($99,6037\%$), celle de l'UACI est égale à $28,6723\%$ et celle du PSNR est faible ($9,2245$ dB). Ces différentes mesures indiquent que le contenu de l'image originale et celui l'image chiffrée marquée sont très différents. Elles mettent aussi en évidence que l'insertion de la liste des valeurs des EP et du message secret n'a pas d'impact sur la sécurité de la méthode de chiffrement. Nous pouvons ainsi conclure que la méthode d'IDCDC réversible proposée est statistiquement sécurisée.

Image	Corrélation horizontale	Corrélation verticale	Entropie (*bpp*)	Score		NPCR (%)	UACI (%)	PSNR (dB)
				test χ^2	p-value			
Image originale	0,8611	0,7666	7,4744	$142,81 \times 10^3$	0	/	/	/
Image chiffrée marquée (fig. 5.9.a)	0,0005	0,0007	7,9994	228,65	0,8810	99,6037	28,6723	9,2245

TABLE 5.3 – Évaluation de la qualité de l'image chiffrée marquée associée à l'image originale de *Baboon*, obtenue en utilisant la méthode proposée.

5.5.4 Comparaison avec l'état-de-l'art et discussion

Dans cette section, nous présentons une comparaison des performances de notre approche avec celles obtenues par les méthodes récentes de l'état-de-l'art en termes de charge utile. Pour cela, nous utilisons tout d'abord les images *Lena* et *Man*, puis les

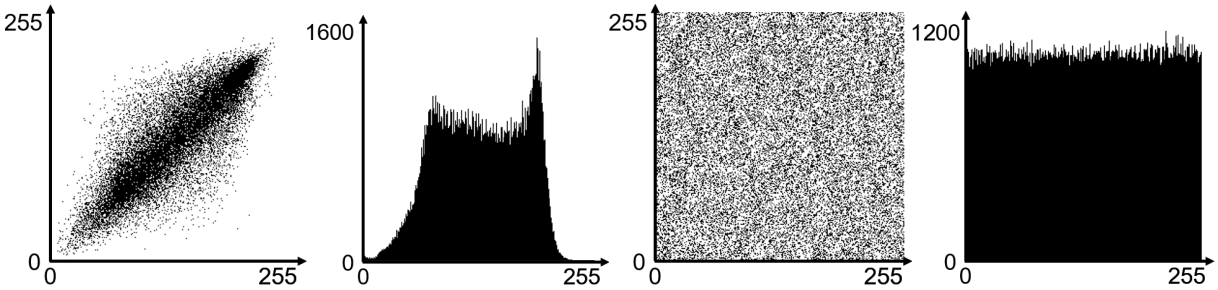
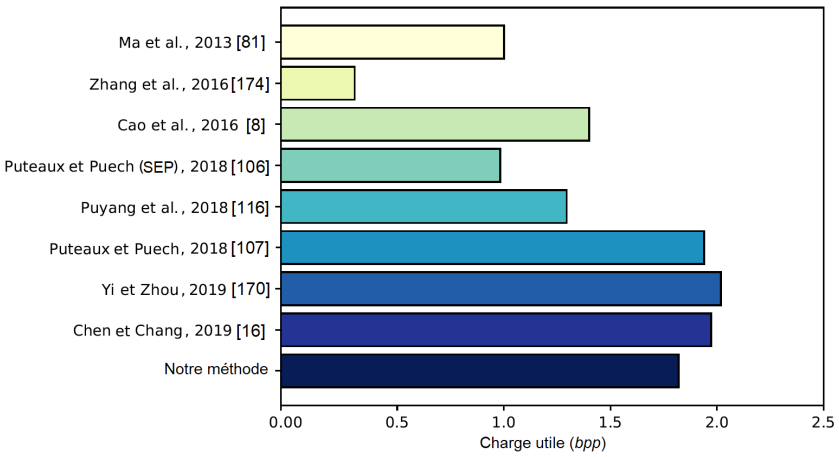


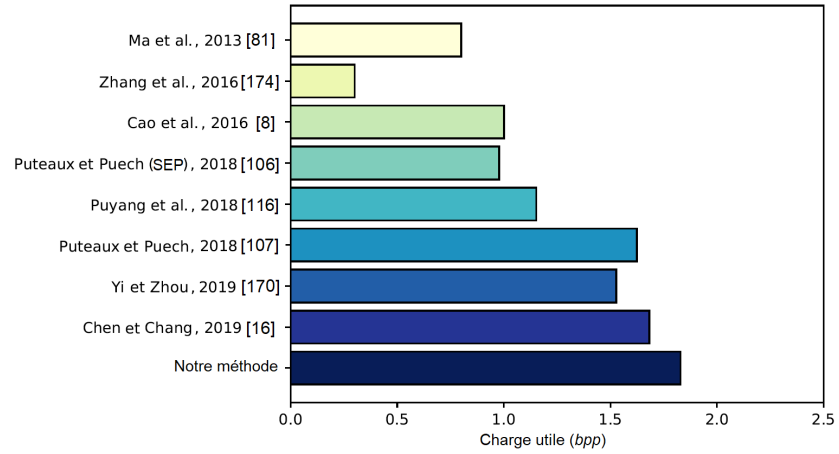
FIGURE 5.10 – Représentations statistiques (corrélation et histogramme) pour l'image originale *Baboon* et l'image chiffrée marquée associée obtenue avec notre méthode illustrée en fig. 5.9.a : a) Corrélation horizontale dans l'image originale, b) Histogramme de l'image originale, c) Corrélation horizontale dans l'image chiffrée marquée (fig. 5.9.a), d) Histogramme de l'image chiffrée marquée (fig. 5.9.a).

10000 images de la base BOWS-2 [3].

Dans la fig. 5.11, nous comparons la méthode proposée avec les algorithmes récents de l'état-de-l'art : les méthodes de Ma *et al.* [81], Zhang *et al.* [174], Cao *et al.* [8], Puyang *et al.* [116], Yi *et al.* [170], Chen et Chang [16], l'approche IDCHC-SEP [106] décrite dans le chapitre 4 et son extension à tous les plans binaires [107] décrite en section 5.3. La comparaison des performances est effectuée en termes de charge utile obtenue (exprimée en *bpp*). Notons que la charge utile calculée pour notre méthode correspond à sa valeur réelle, c'est-à-dire selon le nombre de bits du message secret insérés, en excluant les bits de la liste des valeurs des EP et le drapeau FDL de chaque plan binaire marqué. Pour ces comparaisons, nous avons utilisé les images *Lena* et *Man*. Nous ne comparons pas la qualité des images reconstruites. En effet, quelle que soit la méthode utilisée, l'image originale est parfaitement reconstruite durant la phase de décodage (PSNR$\rightarrow +\infty$ et SSIM $= 1$) à l'aide de la clé de chiffrement seulement [116, 170, 107] ou en utilisant les deux clés [81, 174, 8, 16]. Nous pouvons voir que la méthode proposée obtient de très bons résultats. Pour les deux images, la charge utile obtenue par l'approche de Zhang *et al.* [174] est inférieure à $0,5$ *bpp*. En utilisant les méthodes de Ma *et al.* [81], Cao *et al.* [8], Puyang *et al.* [116] ou l'approche IDCHC-SEP (chapitre 4), la charge utile est supérieure, mais proche d' 1 *bpp* et n'excède pas $1,5$ *bpp*. L'extension de l'approche IDCHC-SEP (section 5.3), ainsi que les méthodes de Yi *et al.* [170] et Chen et Chang [16] sont les trois méthodes les plus récentes atteignant une haute charge utile. De ce fait, elles permettent toutes d'insérer une grande quantité d'information dans les deux images. Pour l'image de *Lena*, la charge utile obtenue est proche de 2 *bpp* et, pour *Man*, elle est supérieure à $1,5$ *bpp*, mais reste inférieure à $1,75$ *bpp*. Avec la méthode proposée, il est possible d'insérer les bits d'un message secret dans les trois plans les plus significatifs des deux images. Ainsi, la charge utile est très élevée : $1,8100$ *bpp* pour l'image de *Lena* et $1,8289$ *bpp* pour celle de *Man*. Notons que ces valeurs sont cohérentes avec les taux de compression calculés entre les deux images originales et leurs versions compressées avec JPEG-LS ($1,7415$ pour *Lena* et $1,7067$ pour *Man*). Nos résultats sont donc comparables avec ceux obtenus par notre extension de l'approche IDCHC-SEP (section 5.3) et les méthodes de Yi *et al.* [170] et de Chen et Chang [16].



(a) Image test : *Lena*.



(b) Image test : *Man*.

FIGURE 5.11 – Comparaison des performances entre la méthode proposée et les méthodes récentes de l'état-de-l'art [81, 174, 8, 106, 107, 116, 170, 16].

En fig. 5.12, nous analysons les résultats sur l'ensemble des images de la base BOWS-2 [3] obtenus par la méthode proposée dans ce chapitre, nos deux approches décrites dans le chapitre 4 [106], l'extension de l'approche IDCHC-SEP [107] et les méthodes de Puyang *et al.* [116], Yi *et al.* [170] et Chen et Chang [16]. Similairement aux résultats présentés en fig. 5.11, la charge utile obtenue avec la méthode décrite dans le chapitre 4 [106] et l'approche de Puyang *et al.* [116] est inférieure à $1,5$ *bpp*. De plus, même si les méthodes [107], [170] et [16] permettent d'obtenir des valeurs de la charge utile plus élevées pour les images moins texturées, les valeurs moyennes sur l'ensemble de la base sont moins importantes qu'en utilisant la méthode proposée. Ainsi, la charge utile moyenne sur la base BOWS-2 avec la méthode décrite dans ce chapitre est égale à $2,4586$ *bpp*. Trois raisons peuvent expliquer cette augmentation de la charge utile par rapport aux autres méthodes basées sur la prédiction des MSB [116, 107]. Tout d'abord, dans les méthodes précédentes, quel que soit le plan binaire traité, près de 24 bits sont perdus en cas d'erreur de prédiction et ne peuvent pas être utilisés pour l'IDC. En effet, les plans binaires sont traités par séquences de 8 bits et deux drapeaux servent à mettre en évidence les séquence avec un ou plusieurs erreurs de prédiction. Avec

notre méthode, chaque plan binaire est traité bit par bit. Si une erreur de prédiction est détectée, sa valeur est stockée dans la liste des valeurs des erreurs de prédiction. Chaque valeur est alors codée avec $8-k$ bits (en considérant le $k^{\text{ème}}$ plan binaire comme le plan binaire courant). Cela signifie que même si plus d'erreurs de prédiction sont présentes dans les plans LSB que dans le plan MSB, un nombre plus faible de bits est utilisé pour coder chaque valeur des erreurs de prédiction. De ce fait, la charge utile est plus élevée que dans les approches décrites en [116] et en [107]. Elle est encore plus importante en utilisant un prédicteur plus efficace tel que le MED (*Median Edge Detector*). Cela a pour effet de diminuer le nombre d'erreurs de prédiction, ce qui augmente la valeur de la charge utile.

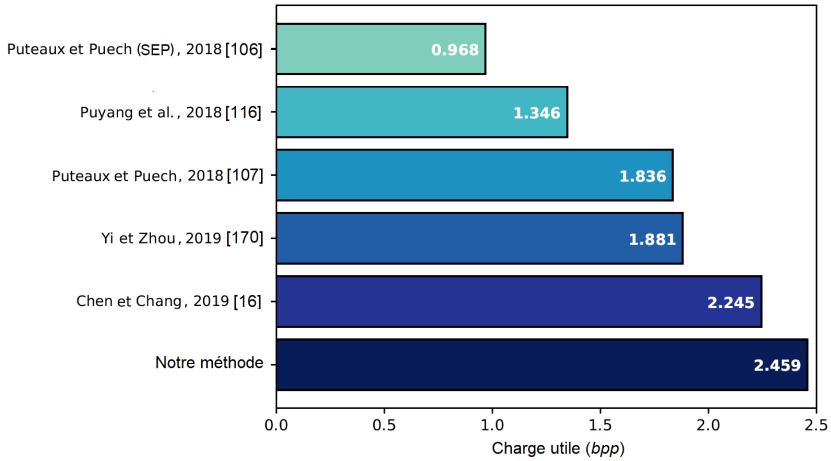


FIGURE 5.12 – Comparaison des performances entre la méthode proposée et les méthodes récentes de l'état-de-l'art d'IDCDC haute capacité [106, 107, 116, 170, 16] sur les images de la base BOWS-2 [3].

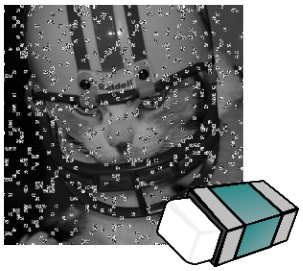
Pour conclure cette section, au travers de nombreuses expérimentations, nous avons montré que notre méthode d'IDCDC récursive est parfaitement réversible (PSNR$\rightarrow +\infty$, SSIM $= 1$) et permet d'obtenir une charge utile très élevée. De plus, l'extraction du message secret lors de la phase de reconstruction à lieu sans erreur. Les résultats obtenus sont supérieurs à ceux des méthodes récentes de l'état-de-l'art. La valeur médiane de la charge utile est égale à $2,3209$ bpp et la moyenne est de $2,4586$ bpp tandis que les autres méthodes basées sur la prédiction des MSB telles que [116] et [107] obtiennent respectivement des valeurs de la charge utile égales à $1,3460$ bpp et $1,8360$ bpp en moyenne. En effet, seulement les deux plans MSB sont utilisés pour insérer les bits d'un message secret dans la méthode [116] et c'est également souvent le cas avec l'approche décrite dans la section 5.3. En revanche, avec la méthode proposée, dans la plupart des cas, au moins les trois plans MSB d'une image sont prédictibles et peuvent alors être utilisés pour l'IDC. Notons également que seule la clé de chiffrement est nécessaire pour reconstruire l'image originale en clair depuis l'image chiffrée marquée, ce qui signifie que notre méthode est séparative. En outre, l'analyse statistique montre que notre approche offre un bon niveau de sécurité visuelle car aucune information sur l'image originale ne semble pouvoir être extraite de l'image chiffrée marquée. Enfin, la méthode proposée achève un très bon compromis entre la qualité de l'image reconstruite et la charge utile, tout en étant statistiquement sécurisée.

5.6 Conclusion

Dans ce chapitre, nous avons tout d'abord présenté les méthodes récentes d'IDCDC développées ces dernières années et permettant d'insérer une grande charge utile. Nous avons ensuite décrit brièvement l'extension de l'approche IDCHC-SEP que nous avons proposé. Nous avons ensuite présenté une nouvelle méthode d'IDCDC récursive, réversible et permettant d'obtenir une haute charge utile. Les plans binaires de l'image originale sont parcourus récursivement, du plus significatif au moins significatif. Pendant la phase d'encodage, pour chaque plan binaire, le contenu en clair est analysé et les EP sont prises en compte en construisant une carte renseignant leur emplacement et une liste de leurs valeurs. Un test est ensuite réalisé pour savoir si la liste des valeurs des EP peut être insérée dans le plan binaire courant. Si cela est possible, l'image est adaptée pour mettre en évidence l'emplacement des pixels mal prédits. Notons que cette phase d'adaptation est parfaitement réversible et qu'elle n'empêche pas la reconstruction sans perte de l'image originale. Le plan binaire courant est ensuite traité indépendamment des autres. Il est chiffré et marqué par les bits des valeurs des EP et du message secret. Si l'insertion de la liste des valeurs des EP n'est pas possible dans le plan binaire courant, alors le processus récursif prend fin. Dans ce cas, les plans binaires courant et restants sont seulement chiffrés (et non marqués). Pendant la phase de décodage, chaque plan binaire est reconstruit de manière récursive du plan LSB au plan MSB. En effet, les plans binaires les moins significatifs sont nécessaires pour reconstruire les plans binaires les plus significatifs. Ainsi, tous les plans binaires de l'image originale sont parfaitement reconstruits par prédiction (PSNR $\rightarrow +\infty$, SSIM = 1). Tous les pixels adaptés, correspondant aux pixels mal prédits, peuvent être localisés en identifiant les cas spéciaux. Ils sont ensuite corrigés en utilisant la liste des valeurs des EP. En plus de permettre un excellent compromis entre la qualité de l'image reconstruite et la charge utile, l'image chiffrée marquée est statistiquement sécurisée. Cette méthode d'IDCDC peut donc être utilisée pour assurer un niveau confidentiel de sécurité visuelle, mais aussi pour des contrôles d'authenticité ou d'intégrité avec l'aide du message secret inséré. Avec une charge utile moyenne de $2,4586$ bpp et une valeur médiane de $2,3209$ bpp (sur une base de 10000 images), les performances de notre approche sont meilleures que celles des méthodes récentes de l'état-de-l'art, où la charge utile est généralement inférieure à 1 bpp.

Dans de futurs travaux, nous sommes intéressés par trouver un moyen efficace pour conserver le message secret inséré dans l'image reconstruite après le déchiffrement de l'image chiffrée marquée. Dans ce contexte, nous cherchons à développer une méthode de chiffrement préservant le format et homomorphe à l'IDC. De plus, dans nos recherches en cours, nous portons un intérêt particulier à l'IDC réversible et haute capacité dans les images JPEG crypto-compressées.

Les travaux présentés dans ce chapitre ont fait l'objet de deux publications internationales. L'extension de l'approche IDCHC-SEP, décrite en section 5.3, a été présentée au cours de la conférence internationale IEEE WIFS 2018 [107]. Par ailleurs, la méthode d'IDCDC récursive, réversible et haute capacité développée en détail dans ce chapitre a été publiée dans le revue internationale IEEE Transactions on Multimedia en 2020 [112].



CHAPITRE 6

Analyse et correction d'images chiffrées bruitées

Sommaire

6.1 Introduction

Comme présenté dans le chapitre 2, les méthodes de chiffrement d'images permettent de préserver leur confidentialité visuelle et ainsi assurer leur sécurité durant leur transmission ou leur stockage. Cependant, ces méthodes sont très sensibles au bruit, ce qui constitue un problème important. En effet, si une image chiffrée est altérée par un bruit, son contenu original ne peut pas être directement reconstruit lors de la phase de déchiffrement. Dans ce chapitre, nous nous intéressons à l'exploitation des différences statistiques entre les blocs de pixels dans le domaine clair et ceux du domaine chiffré. Grâce à cette analyse, nous décrivons alors une approche efficace de correction des images chiffrées bruitées basée sur l'utilisation d'un chiffrement à l'aide d'un nouveau mode appelé CFB-puis-ECB.

La suite de ce chapitre est organisée comme suit. La section 6.2 explique comment l'entropie de Shannon peut être exploitée pour analyser des blocs de pixels de petite taille. La section 6.3 détaille les méthodes de l'état-de-l'art développées pour corriger les images chiffrées bruitées. Dans la section 6.4, nous décrivons alors la méthode proposée impliquant l'utilisation d'une approche de chiffrement à l'aide d'un nouveau mode appelé CFB-puis-ECB. Les résultats expérimentaux réalisés sont ensuite présentés en section 6.5. Enfin, la section 6.6 conclut ce chapitre et présente des pistes pouvant être envisagées dans des travaux futurs.

6.2 Analyse des blocs de petite taille avec l'entropie

Dans cette partie, nous étudions d'abord la mesure de l'entropie d'ordre zéro en fonction de la taille des blocs (section 6.2.1). Dans un second temps, nous exploitons la redondance entre les pixels, propriété très utile du domaine clair. Pour cela, nous construisons la carte des distances entre pixels voisins et effectuons une analyse de l'entropie de cette carte (section 6.2.2). En section 6.2.3, nous comparons alors l'efficacité de ces deux mesures basées sur l'entropie locale à discriminer les blocs de pixels en clair des blocs de pixels chiffrés.

6.2.1 Entropie d'ordre zéro

Soit I une image de taille $m \times n$ pixels avec 2^l niveaux de gris α_k $(0 \leq k < 2^l)$, de probabilité associée $p(\alpha_k)$. L'entropie d'ordre zéro $H(I)$ de cette image est calculée à l'aide de l'équation (2.16) décrite dans le chapitre 2. Dans le cas particulier où les 2^l niveaux de gris α_k ont la même probabilité, la valeur de l'entropie d'ordre zéro est maximale et est égale à :

$$H(I) = -\sum_{k=0}^{2^l-1} \frac{1}{2^l} \log_2 \left(\frac{1}{2^l}\right) = \log_2 (2^l) = l \ bpp. \tag{6.1}$$

Notons que si un algorithme de chiffrement est efficace alors les valeurs des pixels de l'image chiffrée sont générées pseudo-aléatoirement. De ce fait, la distribution des

niveaux de gris de l'image tend vers une distribution uniforme. La valeur de l'entropie d'une image chiffrée codée sur 2^l niveaux de gris (H_{chif}) est alors très proche de l'entropie maximale :

$$H_{chif} \approx l \ bpp. \tag{6.2}$$

Dans le domaine clair, la distribution des pixels peut être approchée par une distribution normale. Dans le domaine discret, l'équivalent d'une distribution normale est la distribution binomiale selon le théorème de Moivre-Laplace. De ce fait, dans le domaine clair, la valeur de l'entropie H_{clair} d'une image dont les pixels sont codés sur 2^l niveaux de gris est approchée par celle de la loi binomiale $\mathcal{B}(2^l, p)$:

$$H_{clair} \approx \frac{1}{2} \log_2 \left[2\pi e(2^l - 1)p(1-p)\right] \ bpp, \tag{6.3}$$

où e est la base de la fonction exponentielle et $0 \leq p \leq 1$.

Si nous comparons la valeur de l'entropie d'ordre zéro d'une image en clair et celle d'une image chiffrée, nous souhaitons que :

$$\begin{aligned} \frac{1}{2} \log_2 \left[2\pi e(2^l - 1)p(1-p)\right] &\leq l, \\ \log_2 \left[2\pi e(2^l - 1)p(1-p)\right] &\leq 2l, \\ c + \log_2(2^l - 1) &\leq 2l, \end{aligned} \tag{6.4}$$

où c est un entier de petite taille.

Cette inégalité est toujours vraie si l est un entier de grande taille. Ainsi, l'entropie d'ordre zéro d'une image dans le domaine clair est plus faible que celle mesurée dans le domaine chiffré :

$$H_{clair} < H_{chif}. \tag{6.5}$$

Nous proposons alors de considérer des blocs de 2^h pixels dans une image codée sur 2^l niveaux de gris pour définir le concept d'entropie locale. Soit b_i, un bloc de 2^h pixels dans une image avec 2^l niveaux de gris. L'entropie locale (*i.e.* à l'intérieur d'un bloc b_i) est majorée par la valeur minimale entre la taille du bloc 2^h et le nombre de niveaux de gris 2^l de l'image :

$$\begin{aligned} H_{(h,l)}(b_i) &= -\sum_{k=0}^{2^l-1} p(\alpha_k) \log_2(p(\alpha_k)), \\ &\leq -\min\left(2^h, 2^l\right) \cdot \frac{1}{\min\left(2^h, 2^l\right)} \log_2 \left(\frac{1}{\min\left(2^h, 2^l\right)} \right), \\ &\leq \log_2(\min\left(2^h, 2^l\right)), \\ &\leq \min(h, l). \end{aligned} \tag{6.6}$$

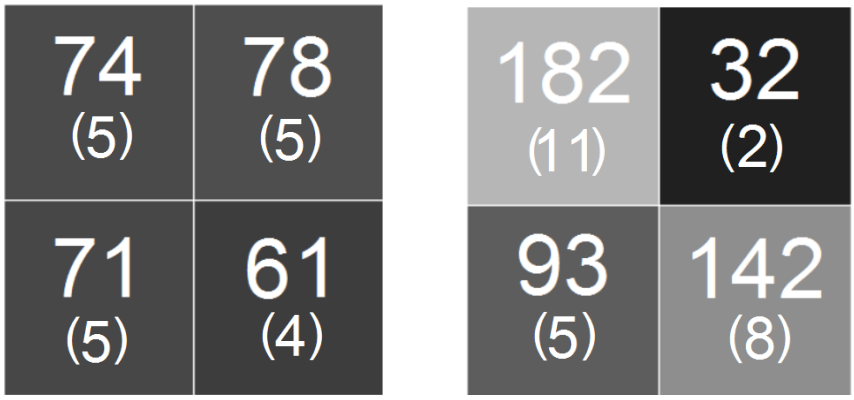
En effet, si la taille du bloc est plus grande que le nombre de niveaux de gris, l'entropie maximale correspond à l'équiprobabilité entre tous les niveaux de gris. Inversement, s'il y a plus de niveaux de gris que de pixels dans le bloc, la valeur de l'entropie maximale est atteinte lorsque tous les pixels ont des valeurs différentes. Dans ce cas, l'échantillon de pixels est épars car certaines valeurs de niveaux de gris ne sont pas

présentes dans le bloc b_i. Pour cette raison, la mesure de l'entropie peut être erronée et un bloc clair peut être considéré comme chiffré.

Le problème est illustré à l'aide d'un exemple présenté fig. 6.1 où nous considérons un bloc de taille $2^h = 2^2$ pixels avec $2^l = 2^8$ niveaux de gris et sa version chiffrée (caractères de grande taille). Dans le bloc de l'image en clair, même si les valeurs des pixels sont relativement proches, puisqu'elles sont toutes différentes, l'entropie est maximale, $H_{(2^2,2^8)_{clair}} = \min\left(2^2, 2^8\right)) = 2$ *bpp*.

Comme nous avons dans les deux cas une valeur maximale de l'entropie, nous ne pouvons pas distinguer un bloc de l'autre en utilisant classiquement l'entropie d'ordre zéro car le nombre de niveaux de gris est bien plus élevé que la taille du bloc. Pour résoudre ce problème, nous proposons de quantifier le nombre de niveaux de gris pour le calcul de l'entropie de façon à diminuer la valeur de 2^l. L'idée est de trouver le meilleur compromis entre la taille des blocs 2^h et le nombre de niveaux de gris 2^l dans l'image.

Si nous considérons à nouveau l'exemple de la fig. 6.1, nous montrons que si nous appliquons une quantification uniforme (valeurs entre parenthèses) à l'image pour se ramener à 2^4 niveaux de gris, nous levons l'ambiguïté à différencier un bloc en clair de sa version chiffrée. En effet, dans le bloc en clair, trois des quatre valeurs des pixels sont dans le même intervalle $[\![64, 79]\!]$ et sont donc codées avec le même niveau de gris 5. L'entropie correspondante est alors : $H_{(2^2,2^4)_{clair}} = -\frac{3}{4} \cdot \log_2\left(\frac{3}{4}\right) - \frac{1}{4} \cdot \log_2\left(\frac{1}{4}\right) = 0,81$ *bpp*.



(a) Bloc de 2^2 pixels dans l'image en clair $H_{(2^2,2^4)_{clair}} = 0,81$ *bpp*.

(b) Bloc de 2^2 pixels dans l'image chiffrée $H_{(2^2,2^4)_{chif}} = 2$ *bpp*.

FIGURE 6.1 – Mesure de l'entropie locale dans un bloc de 2^2 pixels et sa version chiffrée avec AES : mesure initiale de l'entropie d'ordre zéro (en caractères de grande taille) avec 2^8 niveaux de gris ; mesure après requantification de l'image (entre parenthèses) avec 2^4 niveaux de gris.

6.2.2 Entropie de la carte des distances

Lors du calcul de l'entropie d'ordre zéro, nous ne tenons pas compte de la corrélation entre les pixels voisins dans le domaine clair. En effet, les valeurs des pixels du même voisinage sont très proches. Cela n'est pas le cas dans le domaine chiffré : la corrélation entre les pixels est très faible puisque les pixels sont générés pseudo-aléatoirement.

Pour exploiter cette corrélation, nous construisons la carte des distances D pour l'image originale I. Les valeurs des distances correspondent aux valeurs absolues entre deux pixels voisins :

$$\forall d \in D,\ d = d(x, x') = |x - x'|\,, \tag{6.7}$$

avec x et x' deux pixels voisins dans une image I.

Comme l'image originale, la carte des distances est aussi codée sur 2^l niveaux de gris. A l'aide de l'équation (2.16) décrite dans le chapitre 2, comme chaque valeur de la distance d_k $(0 \leq k < 2^l)$ a la probabilité $p(d_k)$, l'entropie de la carte des distances est :

$$H(D) = -\sum_{k=0}^{2^l-1} p(d_k)\log_2(p(d_k)). \tag{6.8}$$

Dans le domaine chiffré, la probabilité théorique associée à la valeur de la distance d est :

$$P(D = d) = \begin{cases} \frac{2(2^l-d)}{(2^{2l}} & \text{si } 1 \leq d \leq 2^l - 1, \\ \frac{1}{2^l} & \text{si } d = 0. \end{cases} \tag{6.9}$$

En effet, la distribution des distances n'est pas uniforme, comme illustré en fig. 6.2.a : elle dépend de la valeur originale des pixels dans la paire de voisins. Par exemple, si un pixel x dans la paire est égal à 128, la valeur de la distance est entre 0 et 128, quelle que soit la valeur de x' :

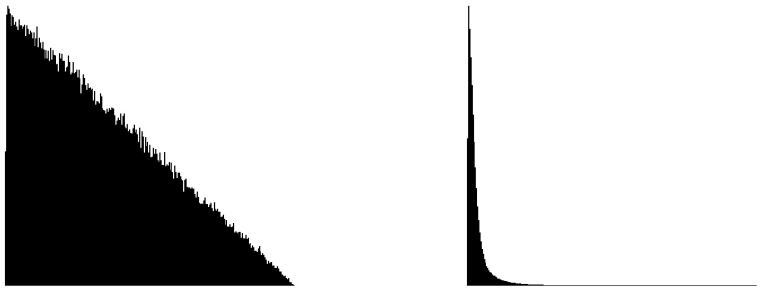
$$\forall x',\ d(x, x') \leq 128,\ P(D > 128 \mid X = 128) = 0. \tag{6.10}$$

En considérant cette valeur de la probabilité, l'entropie théorique de la carte des distances dans le domaine chiffré est :

$$\begin{aligned} H^D_{chif} &= \left[-\sum_{k=1}^{2^l-1} \frac{2k}{2^{2l}} \log_2\left(\frac{2k}{2^{2l}}\right)\right] - \frac{1}{2^l}\log_2\left(\frac{1}{2^l}\right), \\ &\geq \log_2\left(2^{l-1}\right)\ bpp. \end{aligned} \tag{6.11}$$

Dans une image en clair, la distribution des valeurs des distances est semblable à une distribution géométrique, comme illustré fig. 6.2.b. Si une variable aléatoire D suit une loi géométrique de paramètre p, sa probabilité d'être égale à d est :

$$P(D = d) = (1 - p)^{d-1}p. \tag{6.12}$$



(a) (b)

FIGURE 6.2 – Histogrammes de la carte des distances : a) Dans une image chiffrée, b) Dans une image en clair.

En conséquence, la valeur théorique de l'entropie pour la distribution des distances dans le domaine clair est :

$$\begin{aligned} H^D_{clair} &= -\sum_{k=1}^{2^l-1} \left((1-p)^{k-1}p\right) \log_2 \left((1-p)^{k-1}p\right), \\ &\leq \log_2 \left(2^{l-1}\right) bpp. \end{aligned} \tag{6.13}$$

D'après l'équation (6.11) et l'équation (6.13), nous avons alors :

$$H^D_{clair} < H^D_{chif}. \tag{6.14}$$

La valeur de l'entropie de la carte des distances dans le domaine clair est donc inférieure à celle mesurée dans le domaine chiffré.

6.2.3 Comparaison des mesures de l'entropie locale

Pour évaluer nos mesures de l'entropie locale, nous choisissons aléatoirement 1000 images de la base BOWS-2 [3] et les chiffrons à l'aide de l'algorithme AES en mode CBC. Notons que nous utilisons une clé de chiffrement différente pour chaque image. Nous considérons ensuite les blocs de 2^4 pixels (taille pré-requise pour l'AES 128 bits) des images en clair et de leur version chiffrée. Nous y mesurons l'entropie locale d'ordre zéro et celle de la carte des distances. Ainsi, pour comparer l'efficacité de ces deux mesures de l'entropie locale à discriminer les blocs de pixels en clair des blocs de pixels chiffrés, nous observons le nombre d'erreurs, *i.e.* quand un bloc de l'image en clair a une valeur de l'entropie plus élevée que sa version chiffrée.

Dans la fig. 6.3, nous considérons des blocs de 2^4 pixels et comparons le pourcentage d'erreurs obtenu en utilisant l'entropie d'ordre zéro (en bleu) avec celui associé à la mesure de l'entropie de la carte des distances (en rouge) en fonction du nombre 2^l de niveaux de gris considérés lors du calcul. Nous pouvons voir que, quelle que soit sa valeur, le nombre d'erreurs est toujours plus faible avec l'entropie de la carte des distances qu'avec l'entropie d'ordre zéro. En particulier, avec l'entropie de la carte des distances, le nombre d'erreurs est proche de zéro entre 2^2 et 2^6 niveaux de gris. Cela met en évidence l'importance de prendre en compte la corrélation entre les pixels dans le domaine clair.

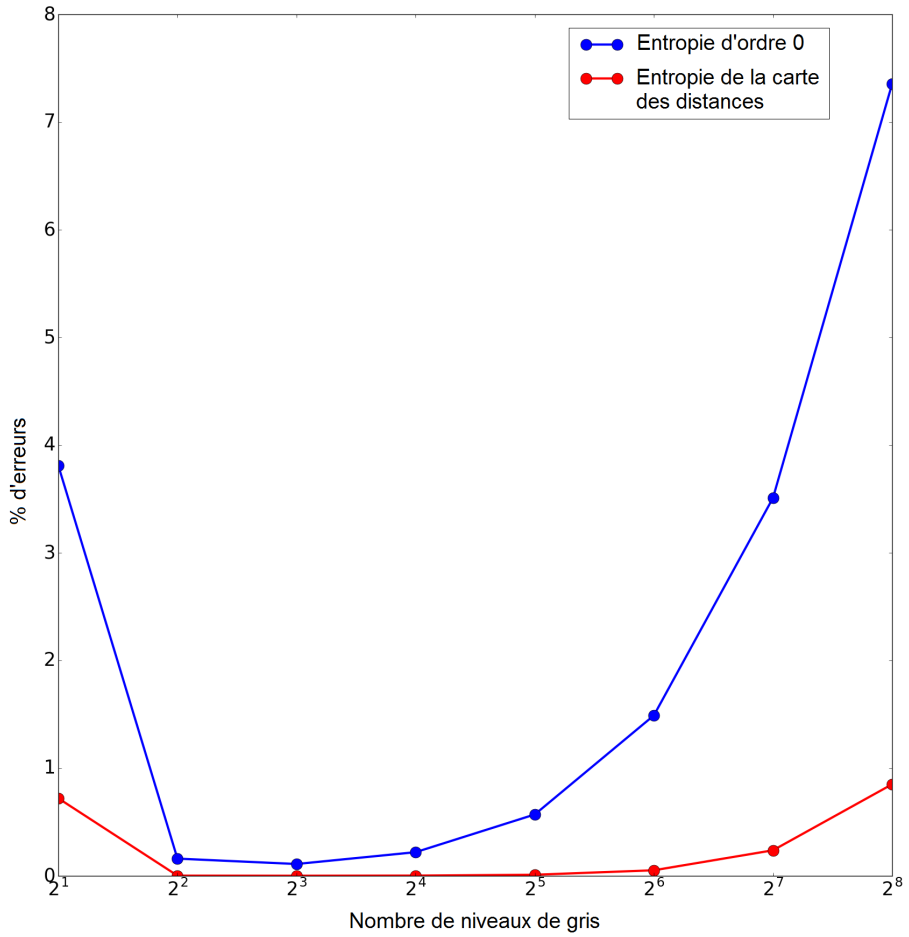


FIGURE 6.3 – Comparaison entre le pourcentage d'erreurs en utilisant l'entropie d'ordre zéro et l'entropie de la carte des distances pour des blocs de taille $2^h = 2^2$ en faisant varier le nombre 2^l de niveaux de gris considérés lors du calcul (moyenne sur les blocs de 1000 images choisies aléatoirement dans la base BOWS-2 [3]).

Le tableau 6.1 présente le nombre maximal de niveaux de gris 2^l à considérer lors du calcul de l'entropie en fonction de la taille de bloc 2^h considérée. Quand la taille du bloc est bien plus grande que le nombre de niveaux de gris de l'image (*i.e.* 256), il n'est pas nécessaire de quantifier l'image. Dans le cas contraire, quand il y a plus de niveaux de gris que de pixels dans le bloc, la distribution des niveaux de gris est éparse. Le meilleur compromis entre le nombre de niveaux de gris et la taille de bloc consiste à choisir un nombre de niveaux de gris inférieur à la taille du bloc. En effet, la meilleure quantification consiste à avoir des blocs dans le domaine clair relativement homogènes et des niveaux de gris uniformément distribués dans le domaine chiffré.

Ainsi, l'entropie locale peut être utilisée pour discriminer les blocs de pixels en clair des blocs de pixels chiffrés. En particulier, dans le cas des blocs de 2^4 pixels, le nombre maximum de niveaux gris devant être considéré pour le calcul est de 2^3. De plus, nos résultats expérimentaux montrent qu'utiliser l'entropie de la carte des distances permet d'obtenir de meilleurs résultats.

Dans la suite de ce chapitre, nous nous intéressons alors à la problématique de la correction des images dans le domaine chiffré, et en particulier aux méthodes basées

Taille des blocs 2^h (pixels)	2^2	2^4	2^6	2^8	$\geq 2^{10}$
Nombre max. de niveaux de gris 2^l	2^3	2^3	2^4	2^6	2^8
% des erreurs (ordre zéro)	4,7856	0,1066	0,0017	0	0
% des erreurs (distances)	4,0589	0,0012	0	0	0

TABLE 6.1 – Nombre maximal de niveaux de gris 2^l à considérer pour minimiser le nombre d'erreurs en fonction de la taille des blocs 2^h et pourcentage d'erreurs associé en utilisant l'entropie d'ordre zéro et l'entropie de la carte des distances (moyenne sur 1000 images de taille 512×512 choisies aléatoirement dans la base BOWS-2 [3]).

sur l'exploitation des différences statistiques entre les blocs de pixels en clair et les blocs de pixels chiffrés.

6.3 Correction d'images dans le domaine chiffré

La plupart des méthodes précédentes n'ont pas été développées spécifiquement pour les images. Ainsi, il est proposé supprimer le bruit des images chiffrées bruitées en utilisant des codes correcteurs d'erreur (CCE) [159, 85]. Les approches basées CCE consistent à introduire de la redondances dans les données. Des bits de contrôle, calculés à partir des données avec des algorithmes spécifiques, sont ajoutés au flux binaire original. A la réception, les bits de contrôle sont recalculés à partir des données reçues et comparés avec les valeurs stockées. Si les bits de contrôle calculés coïncident avec les valeurs stockées, il n'existe pas d'erreur de transmission. Le cas échéant, si une erreur est détectée alors celle-ci doit être corrigée. Cette correction d'erreur peut s'effectuer selon deux principes, à savoir la retransmission (*ARQ, Automatic Repeat reQuest*) ou la correction (*FEC, Forward Error Correction*). L'ARQ consiste à répéter une demande de retransmission des données corrompues jusqu'à ce que l'intégralité des données sont vérifiée et considérée comme correcte. La FEC est basée sur l'encodage des données en utilisant des CCE avant leur transmission.

Par ailleurs, certains travaux de recherche se sont intéressés à la correction des erreurs pendant l'exécution de l'algorithme de chiffrement AES en procédant à des calculs de parité et des modifications en entrée ou en sortie de chaque tour [163, 89]. Czapski et Nikodem ont aussi proposé une approche de correction des erreurs dans laquelle ils supposent que l'injection des erreurs affecte seulement un octet par bloc [24]. Leur algorithme est particulièrement efficace : il permet de détecter plus de 99% des erreurs. Notons que ces différents travaux se concentrent sur les spécifications des algorithmes de chiffrement. Ainsi, ils ne prennent pas en compte les propriétés statistiques des données chiffrées bruitées.

Les méthodes de correction des erreurs préservant la vie privée ont également été proposées. Hu *et al.* ont suggéré d'utiliser une méthode de chiffrement double pour réaliser un débruitage par patchs basé sur l'algorithme de réduction du bruit numérique *non-local means* [53]. Le premier chiffrement est généré en utilisant le cryptosystème de Paillier [91] et le deuxième est obtenu avec une transformation préservant la confidentialité et permettant une recherche non locale sur le *cloud*. SaghaianNejadEsfahani *et al.* ont

adopté une approche de partage de secret pour un débruitage par ondelettes [127]. Ils ont en effet utilisé la méthode de Shamir [132] pour développer des algorithmes permettant de gérer les opérations de traitement du signal. Ces dernières sont ensuite utilisées au sein d'un système de débruitage par ondelettes. Pedrouzo-Ulloa *et al.* ont présenté une méthode de correction des erreurs basée sur un chiffrement post-quantique. Ils montrent que les opérations homomorphes polynomiales et celles de seuillage peuvent être combinées directement dans le domaine chiffré pour traiter et corriger les images chiffrées [93].

D'autres méthodes consistent à réaliser une analyse statistiques de chaque bloc de pixels d'une image chiffrée pendant la phase de déchiffrement pour déterminer s'il est correctement déchiffré. Islam *et al.* ont expliqué comment corriger les images chiffrées avec l'AES bruitées en évaluant trois métriques statistiques : la variance globale, la somme des variances locales et la somme des différences au carré [56]. Les auteurs utilisent ces statistiques locales des données visuelles et les propriétés de confusion et de diffusion des algorithmes de chiffrement pour supprimer les erreurs.

6.4 Méthode proposée

Nous proposons une nouvelle méthode de correction d'images chiffrées bruitées exploitant les différences statistiques entre les blocs de pixels dans le domaine clair et ceux dans le domaine chiffré. En section 6.4.1, nous présentons un nouveau mode de chiffrement basé sur la combinaison du mode CFB et du mode ECB. En utilisant ce mode de chiffrement CFB-puis-ECB, une image originale est chiffrée puis transférée sur un réseau ou stockée sur une plateforme *cloud*. Si l'image chiffrée est bruitée pendant sa transmission ou son stockage, certains blocs de pixels ne peuvent pas être directement déchiffrés sans erreur. Avec un chiffrement en utilisant l'AES à l'aide du mode CFB-puis-ECB, en cas d'erreur, le bruit affecte le bloc de pixels courant mais aussi le bloc de pixels suivant par propagation. Ce phénomène peut alors être exploité pour les étapes d'analyse et de correction de l'image chiffrée bruitée. Ainsi, en section 6.4.2, nous décrivons notre méthode proposée d'analyse des blocs de pixels et de correction de l'image chiffrée bruitée. Cette méthode comporte deux étapes principales, à savoir une phase d'initialisation et une phase de correction. Dans chacune de ces étapes, un classifieur permet de discriminer les blocs de pixels en clair de ceux qui sont probablement mal déchiffrés.

6.4.1 Nouvelle approche de chiffrement d'images

Les modes CFB et ECB décrits dans le chapitre 2 sont combinés pour créer un nouveau mode de chiffrement appelé CFB-puis-ECB. Comme illustré en fig. 6.4, les blocs $b_{clair}(i)$ d'une image originale $I_{clair} = \{b_{clair}(i)\},\ 0 \leq i < |blocs|$ sont chiffrés avec la fonction de chiffrement AES $\mathcal{E}_K(\cdot)$ en utilisant la clé K à l'aide du mode CFB d'abord, puis du mode ECB. Un bloc de pixels chiffré $b_{chif}(i)$ est alors obtenu :

$$b_{chif}(i) = \mathcal{E}_K\left(\mathcal{E}_K\left(b_{chif}(i-1)\right) \oplus b_{clair}(i)\right). \tag{6.15}$$

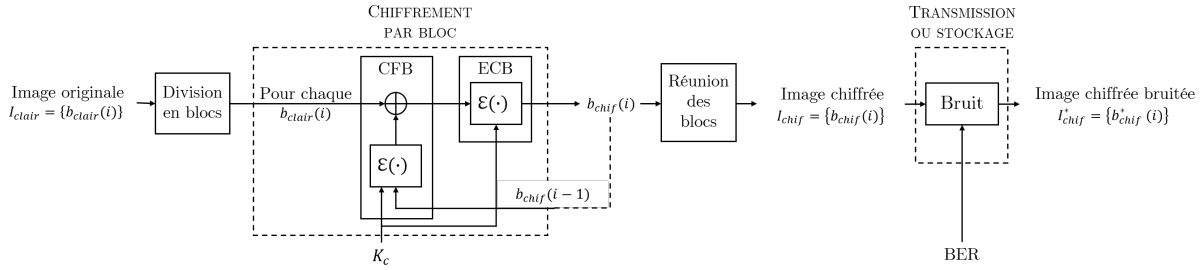


FIGURE 6.4 – Schéma général de la méthode proposée de chiffrement à l'aide du mode CFB-puis-ECB.

Pour déchiffrer le bloc de pixels $b_{chif}(i)$ et retrouver sa version associée dans le domaine clair $b_{clair}(i)$, comme avec le mode ECB, la fonction de déchiffrement AES $\mathcal{D}_K(\cdot)$ doit d'abord être appliquée. Après cela, comme avec le mode CFB, le bloc $b_{clair}(i)$ est obtenu en réalisant une opération ou-exclusif avec la version chiffrée en utilisant $\mathcal{E}_K(\cdot)$ du bloc de pixels précédent $b_{chif}(i-1)$:

$$b_{clair}(i) = \mathcal{D}_K(\mathcal{E}_K(b_{chif}(i-1)) \oplus b_{chif}(i)). \tag{6.16}$$

Après avoir chiffré tous les blocs de l'image I_{clair} en utilisant le mode de chiffrement CFB-puis-ECB, une image chiffrée $I_{chif} = \{b_{chif}(i)\},\ 0 \leq i < |blocs|$ est obtenue. Cette image est alors transmise sur un réseau et/ou stockée sur une plateforme *cloud*. Supposons que cette image est bruitée lors de ce processus à cause d'un bruit de canal : l'image $I^*_{chif} = \{b^*_{chif}(i)\},\ 0 \leq i < |blocs|$ est alors obtenue. Ainsi, durant la phase de déchiffrement avec le mode CFB-puis-ECB, même si la clé K utilisée lors du chiffrement est connue, il n'est pas possible de déchiffrer correctement I^*_{chif}. En effet, tous les blocs de pixels bruités sont entièrement mal déchiffrés car au moins un des bits est altéré. De plus, le bloc de pixels chiffré précédent est nécessaire pour déchiffrer le bloc de pixels chiffré courant. De ce fait, si le bloc chiffré précédent est bruité, même si le bloc courant ne l'est pas, il ne peut pas être correctement déchiffré.

Nous considérons un bloc de pixels $b_{chif}(i)$ de l'image chiffrée I_{chif}. Après avoir été corrompu par un bruit, sa version bruitée $b^*_{chif}(i)$ correspond à :

$$b^*_{chif}(i) = b_{chif}(i) + N(i), \tag{6.17}$$

où $N(i)$ est le bruit associé au bloc de pixels chiffré $b_{chif}(i)$. L'impact du bruit $N(i)$ peut être caractérisé par un taux d'erreur binaire (*BER, Bit-Error-Rate*), qui exprime le nombre de bits erronés divisé par le nombre total de bits transférés. Comme montré dans le tableau 6.2, ce taux est souvent bas quel que soit le type de transmission.

Sans fil	Câble torsadé	Câble coaxial	Fibre optique
10^{-4}	10^{-6}	10^{-9}	10^{-12}

TABLE 6.2 – Valeur du BER en fonction du type de transmission.

Selon la valeur du BER, nous sommes intéressés par avoir la plus petite taille possible de bloc de pixels dans le but de s'assurer qu'au plus un bit par bloc de pixels a été altéré.

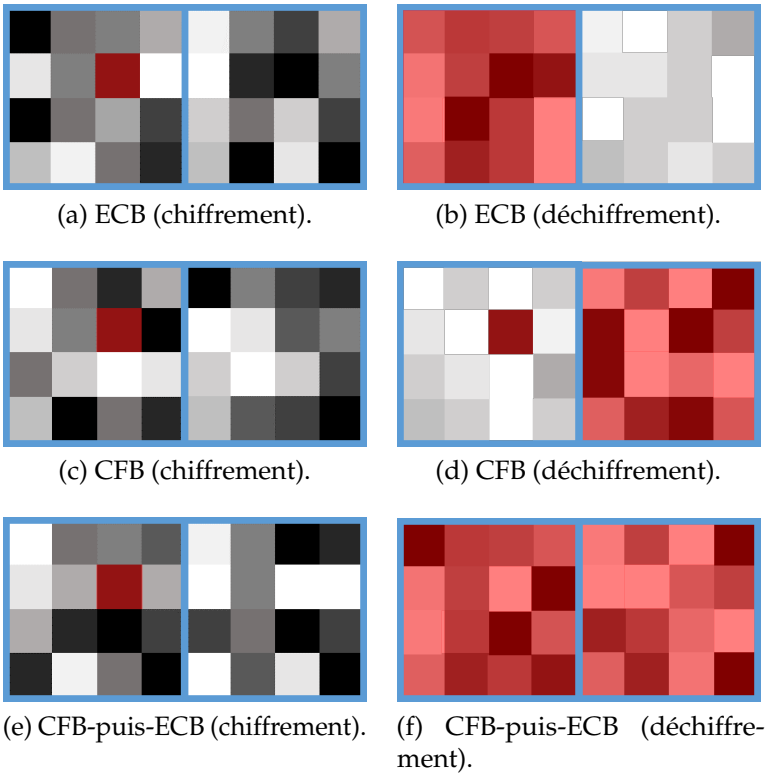
Par exemple, en utilisant notre méthode de chiffrement d'images, la taille des blocs est de 4×4 pixels. Ainsi, si une image chiffrée est bruitée avec un BER de 10^{-3}, un bit tous les six blocs de pixels en moyenne est corrompu.

Nous considérons un bloc de pixels $b^*_{chif}(i)$ de l'image chiffrée bruitée I^*_{chif}. En utilisant l'équation (6.16), sa valeur déchiffrée associée $b_{dec}(i)$ est obtenue :

$$b_{dec}(i) = \mathcal{D}_K(\mathcal{E}_K(b^*_{chif}(i-1)) \oplus b^*_{chif}(i)). \tag{6.18}$$

Si un bloc de pixels chiffré est altéré par le bruit, alors $b_{dec}(i)$ est un bloc de pixels mal déchiffré. En effet, différents cas sont possibles. Si le bloc de pixels chiffré courant ou le bloc de pixels chiffré précédent est bruité ($N(i-1) \neq 0$ ou $N(i) \neq 0$), alors $b_{dec}(i)$ est totalement différent de la valeur recherchée du bloc de pixels original ($b_{clair}(i)$). En revanche, si $N(i) = N(i-1) = 0$, alors $b_{dec}(i)$ correspond à la valeur originale du bloc de pixels dans le domaine clair $b_{clair}(i)$ car les blocs précédent et courant ne sont pas bruités :

$$\begin{aligned} &b_{dec}(i) = b_{clair}(i), \\ &\text{si et seulement si } \begin{cases} b^*_{chif}(i-1) = b_{chif}(i-1) \\ \text{et} \\ b^*_{chif}(i) = b_{chif}(i) \end{cases}. \end{aligned} \tag{6.19}$$



(a) ECB (chiffrement). (b) ECB (déchiffrement).

(c) CFB (chiffrement). (d) CFB (déchiffrement).

(e) CFB-puis-ECB (chiffrement). (f) CFB-puis-ECB (déchiffrement).

FIGURE 6.5 – Trois différents modes de chiffrement utilisant deux blocs de pixels voisins. Colonne de gauche : Blocs de pixels chiffrés bruités : un bit d'un pixel (en rouge) du premier bloc est corrompu à cause de l'altération par un bruit. Colonne de droite : Blocs déchiffrés (les parties mal déchiffrées sont colorées en rouge).

Faire la distinction entre un bloc bien déchiffré et un bloc erroné est un problème difficile, en particulier lorsque la taille de bloc est très petite. Cela constitue l'une des motivations principales à utiliser le mode de chiffrement CFB-puis-ECB à la place d'un mode standard pendant le chiffrement de l'image. En fig. 6.5, nous illustrons la différence entre ces trois modes de chiffrement (ECB, CFB et CFB-puis-ECB) en cas d'altération par un bruit. Nous considérons deux blocs de pixels chiffrés voisins. Pendant la transmission ou le stockage de ces deux blocs, si un bit du premier bloc est corrompu, alors le phénomène de propagation du bruit s'exprime différemment dans les blocs déchiffrés selon le mode utilisé (en rouge en fig. 6.5.a, fig. 6.5.c et fig. 6.5.e).

Si le mode de chiffrement ECB est utilisé, après le déchiffrement, le premier bloc de pixels (*i.e.* celui qui contient le bit corrompu) est entièrement mal déchiffré. En effet, la fonction de déchiffrement de l'AES est appliquée à la mauvaise configuration du bloc chiffré. Dans ce cas, le deuxième bloc de pixels est parfaitement reconstruit dans le domaine clair (fig. 6.5.b) car le premier bloc n'est pas impliqué dans l'opération de déchiffrement du deuxième.

Si le mode de chiffrement CFB est utilisé, tous les pixels du premier bloc sont correctement déchiffrés, à l'exception de celui qui est bruité dans le domaine chiffré. En effet, à cause de l'opération ou-exclusif, seulement la partie corrompue (le bit bruité) ne peut pas être reconstruite sans erreur. Par ailleurs, le deuxième bloc de pixels est intégralement mal déchiffré. En effet, la fonction de déchiffrement de l'AES est appliquée à la mauvaise configuration du premier bloc chiffré. Le mauvais flux binaire est alors obtenu et l'opération ou-exclusif avec le deuxième bloc chiffré ne permet de reconstruire aucune valeur des pixels du bloc original en clair (fig. 6.5.d).

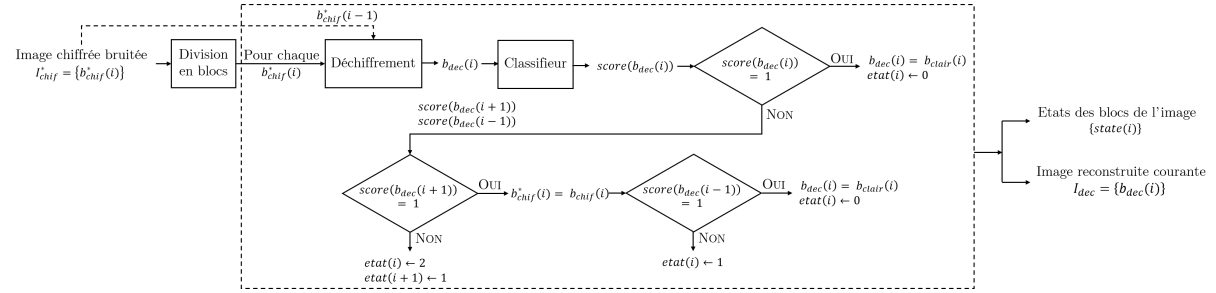
Si le mode de chiffrement CFB-puis-ECB est utilisé, alors le déchiffrement des deux blocs de pixels résulte en deux blocs de pixels mal reconstruits (fig. 6.5.f). Pour conclure, dans le but d'aider à la correction d'une image chiffrée bruitée, il est intéressant que le bruit soit diffusé le plus possible entre deux blocs voisins. Nous suggérons alors d'exploiter le phénomène de propagation du bruit en utilisant le mode de chiffrement CFB-puis-ECB. En effet, au lieu de détecter seulement un bloc de pixels mal déchiffré ou un bit mal déchiffré, nous investiguons un moyen efficace de mettre en évidence deux blocs de pixels voisins mal déchiffrés.

6.4.2 Analyse des blocs de pixels et correction

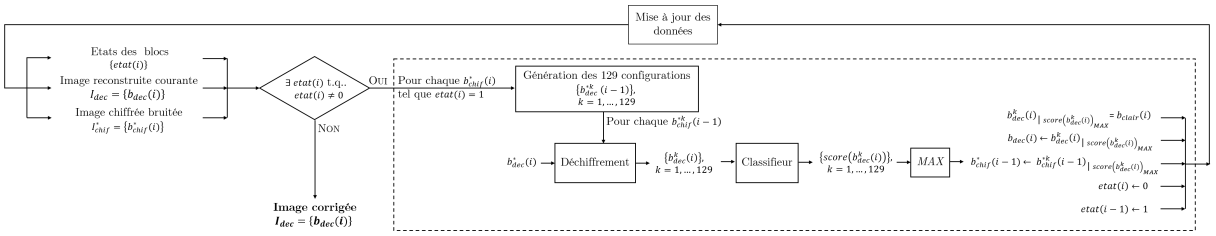
Dans cette section, nous proposons une nouvelle méthode pour corriger efficacement une image chiffrée bruitée durant la phase de déchiffrement. Notre algorithme, illustré en fig. 6.6, comprend deux étapes principales afin de discriminer les blocs de pixels en clair des blocs de pixels chiffrés. Premièrement, nous considérons les blocs de pixels directement déchiffrés (*i.e.* sans correction). Nous réalisons tout d'abord une étape d'analyse pour séparer les blocs de pixels classifiés comme correctement déchiffrés de ceux qui sont classifiés comme possiblement mal déchiffrés. Lors de la deuxième étape, pour chaque bloc de pixels possiblement mal déchiffré, nous analysons les différentes configurations possibles associées. Ainsi, cela nous permet d'effectuer leur correction et de reconstruire l'image originale en clair sans erreur.

Notons qu'un classifieur est utilisé pour discriminer les blocs de pixels en clair de ceux qui sont chiffrés lors des deux étapes. En effet, les blocs de pixels en clair et les

blocs de pixels chiffrés n'ont pas les mêmes propriétés. En particulier, les blocs en clair sont plus homogènes que les blocs chiffrés ou représentent un motif particulier. Compte tenu de ces caractéristiques, un classifieur peut être entraîné à délivrer un score pour faire leur distinction. Deux exemples de classifieurs sont décrits et utilisés en section 6.5, à savoir un classifieur basé sur l'entropie locale et un classifieur nommé CipherNet et basé sur un CNN.



(a) Initialisation des états des blocs de pixels et de l'image reconstruite.



(b) Étape de correction et mise à jour des états des blocs de pixels.

FIGURE 6.6 – Schéma général de la méthode proposée de correction d'images chiffrées bruitées comportant deux étapes principales.

La première étape de notre algorithme est l'étape d'initialisation, comme illustré en fig. 6.6.a. Lors de cette étape, l'image reconstruite I_{dec} est initialisée et un état $etat(i)$ est associé à chaque bloc de pixels. Trois différents états sont possibles et décrits dans le tableau 6.3. Ils sont utilisés pour connaître l'avancement de la correction durant tout le déroulement de l'algorithme. Si $etat(i) = 0$, cela signifie que le bloc de pixels déchiffré associé est classifié comme étant en clair, *i.e.* correctement déchiffré. Dans ce cas, le bloc de pixels n'a pas besoin d'être corrigé. Si $etat(i) = 1$ ou $etat(i) = 2$, cela signifie que le bloc de pixels déchiffré est classifié comme possiblement mal déchiffré. Dans le cas où $etat(i) = 2$, nous ne savons pas si cela est dû au phénomène de propagation du bruit lors du déchiffrement à l'aide du bloc de pixels chiffré précédent ou si le bloc chiffré courant est lui-même bruité. Dans le cas où $etat(i) = 1$, nous considérons que cela est dû au phénomène de propagation du bruit lors du déchiffrement à l'aide du bloc de pixels chiffré précédent. En effet, nous pouvons déduire que le bloc de pixels chiffré courant n'est pas bruité si le bloc suivant est tel que $etat(i + 1) = 0$. En pratique, notons que, dans une séquence de blocs de pixels voisins classifiés comme possiblement mal déchiffrés, le dernier bloc est toujours tel que $etat(i) = 1$ et pour les précédents, $etat(i) = 2$.

Nous considérons l'image chiffrée bruitée I^*_{chif}. Cette image a été chiffrée avec l'algorithme AES à l'aide du mode CFB-puis-ECB et bruitée lors de sa transmission sur un réseau ou son stockage sur une plateforme *cloud*. Tout d'abord, I^*_{chif} est divisée en

Valeur	Description	Correction
0	« bloc de pixels classifié comme clair »	terminée
1	« bloc de pixels classifié comme possiblement mal déchiffré à cause du phénomène de propagation du bruit lors du déchiffrement à l'aide du bloc de pixels chiffré précédent »	en cours
2	« bloc de pixels classifié comme possiblement mal déchiffré à cause d'une corruption par un bruit lors de la transmission/du stockage ou du phénomène de propagation du bruit lors du déchiffrement à l'aide du bloc de pixels chiffré précédent »	à traiter ultérieurement

TABLE 6.3 – Signification des états des blocs de pixels $etat(i)$.

blocs $b^*_{chif}(i)$, avec $0 \leq i < |blocs|$, de 4×4 pixels. Chaque bloc $b^*_{chif}(i)$ est déchiffré en utilisant le bloc voisin précédent $b^*_{chif}(i-1)$ à cause de l'utilisation du mode de chiffrement CFB-puis-ECB. Une version déchiffrée de chaque bloc $b_{dec}(i)$ est ainsi obtenue. Un classifieur est alors utilisé pour déterminer si $b_{dec}(i)$ correspond à un bloc de pixels de l'image originale en clair $b_{clair}(i)$ ou s'il correspond à un bloc de pixels chiffré, ce qui signifie qu'il est possiblement mal déchiffré. Le score de classification $score(b_{dec}(i))$ est compris entre 0 et 1, où 1 indique que le bloc de pixels est en clair en toute certitude. Par ailleurs, plus le score est proche de 0, plus le bloc de pixels est assimilé à un bloc de pixels chiffré. Si $score(b_{dec}(i)) = 1$, alors nous avons la certitude que $b_{dec}(i) = b_{clair}(i)$. Cela signifie que les deux conditions de l'équation (6.19) sont vérifiées : les blocs de pixels chiffrés précédent et courant ne sont pas bruités. Dans ce cas, nous avons $etat(i) = 0$. Si $score(b_{dec}(i)) < 1$, alors il est nécessaire d'observer les scores associés aux deux blocs de pixels voisins $b_{dec}(i-1)$ et $b_{dec}(i+1)$, comme illustré en fig. 6.6.a. Si $score(b_{dec}(i+1)) = 1$, cela signifie que le bloc chiffré courant $b^*_{chif}(i)$ n'est pas bruité. Nous avons alors $b^*_{enc}(i) = b_{enc}(i)$ et $etat(i)$ est initialisé à 1. Dans ce cas, $b_{dec}(i+1)$ est correctement déchiffré, ce qui ne serait pas le cas si $b^*_{chif}(i)$ avait été bruité à cause du phénomène de propagation du bruit en utilisant le mode de chiffrement CFB-puis-ECB. En revanche, si ce n'est pas le cas, $b^*_{chif}(i)$ peut être lui-même corrompu par le bruit et $etat(i)$ est initialisé à 2. Si $score(b_{dec}(i+1)) = 1$ et $score(b_{dec}(i-1)) = 1$, cela signifie que le bloc déchiffré courant est entouré de deux blocs de pixels en clair. De ce fait, ce bloc de pixels courant est forcément en clair et $etat(i)$ est égal à 0. En effet, les scores des blocs de pixels précédent et courant indiquent que $b_{dec}(i)$ n'est pas mal déchiffré à cause du phénomène de propagation du bruit venant de $b^*_{chif}(i-1)$ et $b^*_{chif}(i)$ n'a pas été altéré par un bruit.

A la fin de l'étape d'initialisation, les états de tous les blocs de pixels ($etat(i)$) sont initialisés et une première image reconstruite I_{dec} est obtenue, composée de tous les blocs de pixels qui ont été bien déchiffrés ($I_{dec} = \{b_{dec}(i),\ 0 \leq i < |blocs| \mid etat(i) = 0\}$). Les blocs de pixels avec $etat(i) = 0$ correspondent aux blocs de pixels de l'image originale en clair. Tous les blocs restants ont besoin d'être analysés en détail et corrigés.

L'étape de correction est réalisée en effectuant des itérations sur l'image chiffrée bruitée tant que tous les blocs de pixels ne sont pas tels que $etat(i) = 0$. Les états des blocs de pixels et l'image reconstruite courante sont également mis à jour durant l'intégralité des traitements. A chaque itération, nous nous concentrons sur les blocs de

pixels $b^*_{chif}(i)$ avec $etat(i) = 1$. En effet, comme précédemment mentionné, ces blocs de pixels sont ceux qui ne sont pas bruités mais dont les versions déchiffrées $b_{dec}(i)$ sont possiblement mal déchiffrées à cause du phénomène de propagation du bruit venant de $b^*_{chif}(i-1)$. Pour chacun de ces blocs, nous proposons d'analyser les 129 configurations possibles du bloc chiffré bruité précédent. En effet, comme aucune information sur la localisation du bit corrompu n'est connue, n'importe quel bit du bloc de pixels peut être faux. Notons que, dans notre approche, durant la correction de l'image chiffrée bruitée, nous considérons qu'un bit au plus a été altéré dans chaque bloc de pixels.

Comme illustré en fig. 6.6.b, les 129 configurations possibles $\{b^{*k}_{chif}(i-1),\ 1 \leq k \leq 129\}$ associées au bloc chiffré précédent $b^*_{chif}(i-1)$ sont générées. En effet, ces configurations correspondent à l'originale, plus les $8 \times (4 \times 4) = 128$ autres possibilités, obtenues en inversant la valeur d'un bit après l'autre. Le bloc de pixels chiffré courant $b^*_{chif}(i)$ est alors déchiffré en utilisant chacune des 129 configurations. Nous obtenons ainsi 129 versions déchiffrées possibles $\{b^k_{dec}(i),\ 1 \leq k \leq 129\}$ associées à $b^*_{chif}(i)$. Tous les blocs de pixels déchiffrés $b^k_{dec}(i)$ sont pris en entrée du classifieur. Les scores $\{score(b^k_{dec}(i)),\ 1 \leq k \leq 129\}$ associés sont calculés. Dans la plupart des cas, 128 scores sont faibles et un seul est égal (ou très proche) de 1. Le score maximum $score(b^k_{dec}(i))_{MAX}$ indique la configuration $b^k_{dec}(i)$ associée au bloc de pixels de l'image originale en clair $b_{clair}(i)$. Dans l'image reconstruite, $b_{dec}(i)$ est alors adapté en conséquence et l'état du bloc de pixels courant $etat(i)$ prend la valeur 0. De plus, $b^*_{chif}(i-1)$ est aussi mis à jour (en considérant $b^{*k}_{chif}(i-1)$ tel que $score(b^k_{dec}(i))_{MAX}$) et son état $etat(i-1)$ prend la valeur 1 (car $etat(i) = 0$).

Nous réalisons cette analyse et cette correction pour chaque bloc de pixels de l'image chiffrée bruitée tel que $etat(i) = 1$. A la fin de chaque itération du traitement de tous les blocs, si au moins un état n'est pas nul, alors le processus est réitéré sur l'intégralité de l'image. En effet, cela indique que certains blocs nécessitent toujours d'être corrigés. Quand tous les blocs de pixels ont été corrigés, l'image reconstruite I_{dec} correspond à l'image originale en clair.

6.5 Résultats expérimentaux

Dans cette section, nous présentons les résultats obtenus en appliquant notre méthode de correction d'images chiffrées bruitées. Dans la section 6.5.1, nous illustrons le mode de chiffrement CFB-puis-ECB et le comparons avec les modes classiques de chiffrement ECB et CFB. En section 6.5.2, nous décrivons les deux classifieurs pouvant être utilisés pour discriminer les blocs en clair des blocs chiffrés. La section 6.5.3 détaille un exemple complet de la méthode proposée de correction d'images chiffrées bruitées en utilisant les deux classifieurs. Ensuite, la section 6.5.4 développe les résultats obtenus sur 100 images de la base BOWS-2 [3]. Enfin, en section 6.5.5, nous comparons notre méthode avec les méthodes existantes de correction d'images chiffrées bruitées.

6.5.1 Illustration du chiffrement basé sur l'utilisation du mode CFB-puis-ECB

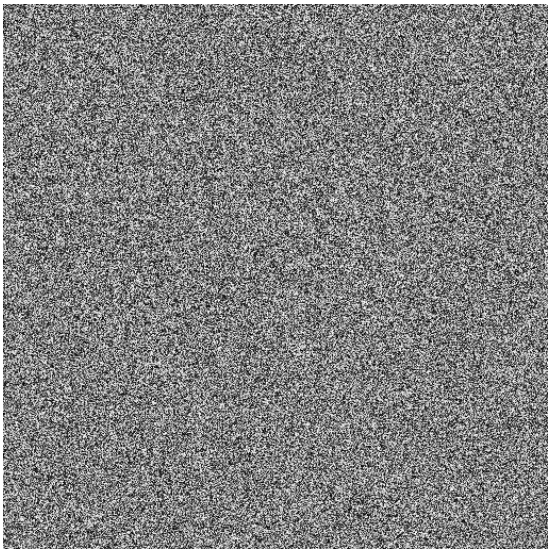
Dans la fig. 6.7, nous illustrons la méthode de chiffrement d'images basée sur l'utilisation du mode CFB-puis-ECB. A partir de l'image originale de la base BOWS-2 [3] (512×512 pixels codés avec 256 niveaux de gris) illustrée en fig. 6.7.a, nous appliquons l'algorithme de chiffrement AES en mode CFB-puis-ECB avec des blocs de 4×4 pixels pour obtenir l'image chiffrée illustrée en fig. 6.7.b. Notons qu'il n'existe aucune information visuelle sur le contenu de l'image originale (PSNR de $7,22\ dB$). La fig. 6.7.c présente l'image chiffrée bruitée associée à la fig. 6.7.b obtenue en introduisant un BER $= 2,6 \times 10^{-3}$. Cela signifie qu'un bit tous les trois blocs en moyenne est aléatoirement corrompu. Notons que cette valeur du BER est relativement haute en comparaison avec les valeurs réelles (présentées dans le tableau 6.2). Le PSNR entre l'image originale et l'image chiffrée bruitée reste bas ($7,21\ dB$) et le PSNR de $33,20\ dB$ entre l'image chiffrée et sa version bruitée indique que la quantité de bruit est relativement haute. La fig. 6.7.d illustre qu'un déchiffrement sans analyse n'est pas possible (PSNR de $14,97\ dB$), même si la clé secrète utilisée lors du chiffrement est connue. Cela est dû au grand nombre de blocs chiffrés bruités mal déchiffrés. De plus, sans analyse, il n'est pas possible de localiser les blocs mal déchiffrés et ainsi les discriminer des blocs correctement déchiffrés.

Dans la fig. 6.8, nous comparons les images directement déchiffrées obtenues d'après des images chiffrées en utilisant différents modes de chiffrement tels que ECB, CFB et le mode proposé CFB-puis-ECB. En utilisant le mode de chiffrement ECB, les blocs de pixels sont chiffrés indépendamment les uns des autres. De ce fait, après un déchiffrement sans correction, si un bloc de pixel est mal reconstruit, les blocs de pixels voisins ne sont pas impactés par le bruit (fig. 6.8.a). En utilisant le mode de chiffrement CFB, les blocs de pixels sont chiffrés en réalisant une opération ou-exclusif avec la version chiffrée du bloc précédent. Par conséquent, en fig. 6.8.b, nous pouvons voir que si un bloc de pixels chiffré est bruité, cela a deux conséquences sur l'image directement déchiffrée. Tout d'abord, le bit bruité dans le bloc de pixels courant est mal déchiffré. De plus, à cause du phénomène de propagation du bruit, tous les bits du bloc de pixels suivant sont mal déchiffrés. Notons qu'avec ce mode chiffrement, il n'est pas possible d'exploiter le fait qu'il existe des erreurs de déchiffrement dans deux blocs voisins pour les corriger. En effet, comme seulement un bit est mal reconstruit dans la version déchiffrée du bloc chiffré bruité, il est très difficile de l'identifier, en particulier lorsque c'est un bit peu significatif. La fig. 6.8.c illustre la version déchiffrée de l'image chiffrée marquée en utilisant la méthode de chiffrement d'images à l'aide du mode CFB-puis-ECB. Avec cette nouvelle méthode de chiffrement, si un bloc de pixels chiffré est bruité, alors le bloc de pixels courant et le suivant sont mal déchiffrés. Ainsi, pendant la phase de correction, ce phénomène de propagation est exploité. En effet, deux blocs de pixels voisins sont plus simples à identifier qu'un bloc de pixels isolé.

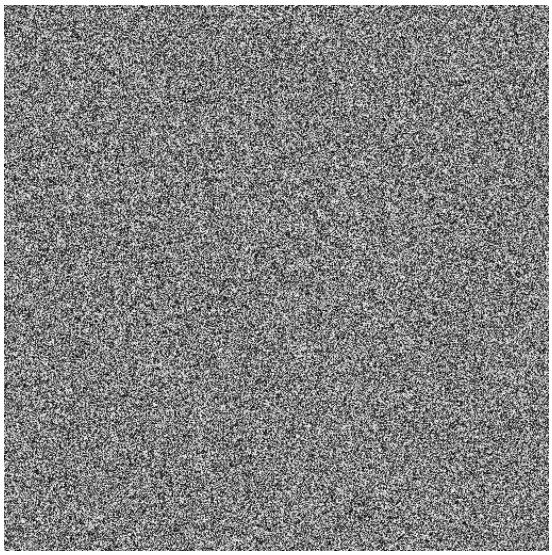
6.5.2 Classifieurs utilisés

Discriminer un bloc de pixels en clair d'un bloc de pixels chiffré est un problème difficile, en particulier lorsque la taille du bloc de pixels est très petite. Dans cette section,

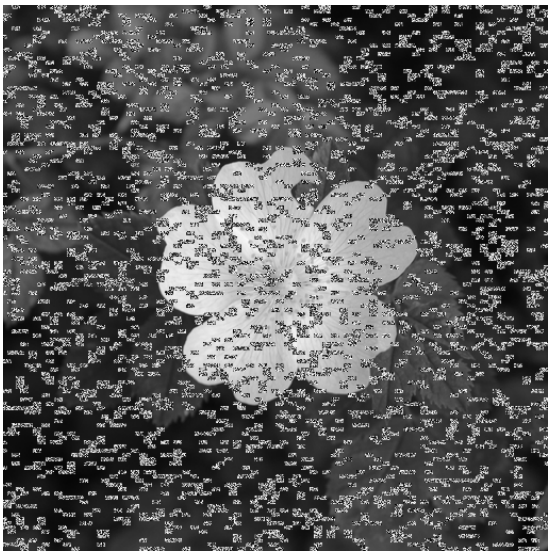
(a) Image originale issue de la base BOWS-2 [3].



(b) Image chiffrée en utilisant le mode CFB-puis-ECB.



(c) Image chiffrée bruitée.



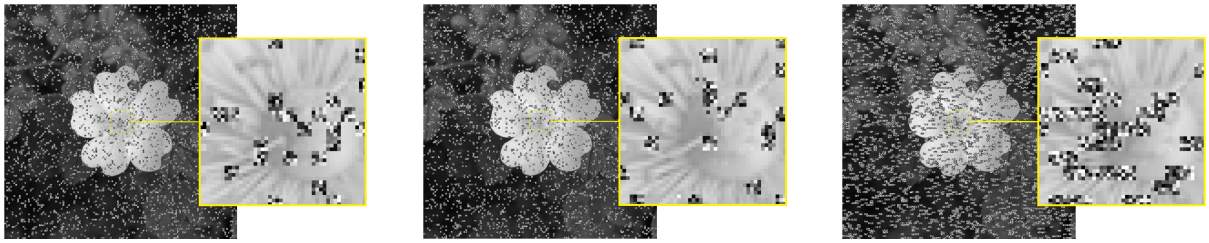
(d) Image directement déchiffrée.

FIGURE 6.7 – Problème du déchiffrement d'une image chiffrée bruitée sans correction.

nous décrivons deux classifieurs pouvant être intégrés dans notre méthode de correction d'images chiffrées bruitées pour déterminer si un bloc de 4×4 pixels est en clair ou chiffré.

Classifieur basé sur l'entropie locale

Dans la section 6.2, nous avons montré que l'entropie locale peut être utilisée pour différencier un bloc de pixels en clair d'un bloc de pixels chiffré. En effet, dans le domaine chiffré, la distribution des pixels est approchée par une distribution uniforme. Ainsi, la valeur de l'entropie est proche de l'entropie maximale et donc plus élevée que dans le domaine clair.



(a) Mode de chiffrement ECB. (b) Mode de chiffrement CFB. (c) Mode de chiffrement CFB-puis-ECB.

FIGURE 6.8 – Images directement déchiffrées obtenues d'après des images chiffrées en utilisant différents modes de chiffrement.

Nous avons vu que pour les blocs de 4×4 pixels, les meilleurs résultat sont obtenus en mesurant l'entropie avec 8 niveaux de gris et sur la carte des distances entre les pixels. Ainsi, le classifieur basé sur l'entropie locale utilise ces paramètres optimaux. Le score de classification associé à un bloc de pixels b_i est alors défini par :

$$score(b_i) = 1 - \frac{H^D_{(2^4,2^3)}(b_i)}{\min(4,3)} = 1 - \frac{H^D_{(16,8)}(b_i)}{3}. \tag{6.20}$$

Classifieur CipherNet

Dans le but de discriminer les blocs de pixels en clair des blocs de pixels chiffrés, nous avons proposé un nouveau réseau convolutionnel léger spécialisé appelé CipherNet dont l'architecture est présentée en fig. 6.9. Dans notre cas d'application, nous considérons un bloc b_i de 4×4 pixels en entrée de notre CNN. Dans la première couche, des filtres passe-haut issus du *Spatial Rich Model* [41] sont appliqués. Ces filtres passe-haut, illustrés en fig.6.10, sont utilisés pour extraire les hautes fréquences. Trois couches de convolution et deux couches de *pooling* permettent d'obtenir un vecteur caractéristique de dimension 1024. Une prédiction $pred(b_i)$ entre 0 et 1 est obtenue, où 0 correspond à la classe « Bloc de pixels en clair » et 1 à la classe « Bloc de pixels chiffré ». Ce score de classification associé à un bloc de pixels b_i est alors défini :

$$score(b_i) = 1 - pred(b_i). \tag{6.21}$$

Dans le but d'entraîner notre classifieur et d'évaluer son efficacité, nous avons considéré une base de 32 millions de blocs de pixels, dont 16 millions sont issus d'images en clair et 16 millions d'images chiffrées avec le mode CFB-puis-ECB. Cette base de données est alors divisée en trois sous-ensembles équilibrés : 17 millions de blocs de pixels pour la phase d'apprentissage, 4 millions pour la phase de validation et 11 millions pour la phase d'évaluation. Notons que seulement un *epoch* a été nécessaire pour que notre modèle converge.

Comparaison des performances

Quel que soit le classifieur, le score de classification $score(b_i)$ associé au bloc de pixels b_i est compris entre 0 et 1. Si $score(b_i) = 1$, nous considérons que le bloc b_i est en

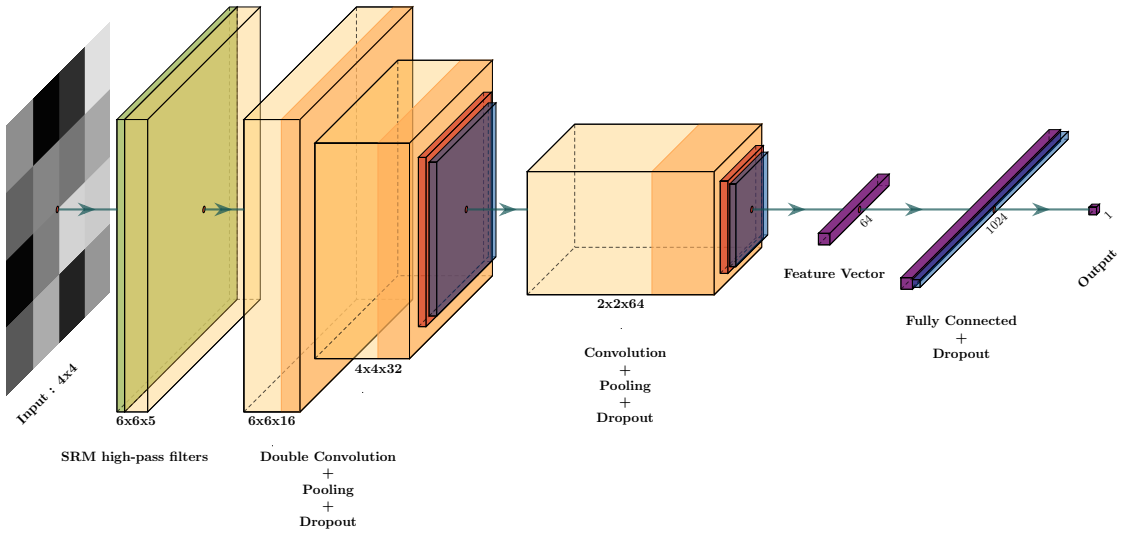


FIGURE 6.9 – Architecture de CipherNet.

$$\frac{1}{3}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -3 & 3 & -1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

(a) HSB.

$$\frac{1}{3}\begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -3 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

(b) VBH.

$$\frac{1}{4}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 2 & -4 & 2 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

(c) RF0.

$$\frac{1}{12}\begin{bmatrix} -1 & 2 & -2 & 2 & -1 \\ 2 & -6 & 8 & -6 & 2 \\ -2 & 8 & -12 & 8 & -2 \\ 2 & -6 & 8 & -6 & 2 \\ -1 & 2 & -2 & 2 & -1 \end{bmatrix}$$

(d) RF1.

$$\frac{1}{2}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

(e) RF2.

FIGURE 6.10 – Les cinq noyaux SRM utilisés pour le filtrage passe-haut dans la première couche de CipherNet.

clair. Dans le cas contraire, si $score(b_i) \neq 1$, cela signifie que le bloc b_i est possiblement un bloc de pixels chiffré.

Dans la fig. 6.11, nous illustrons les cartes de prédiction obtenues avec le classifieur basé sur l'entropie locale (fig. 6.11.a) et le classifieur CipherNet (fig. 6.11.b) sur une image en clair de la base BOWS-2 [3]. Les blocs de pixels qui sont correctement déchiffrés ($score(b_i) = 1$) sont représentés en clair. Cependant, nous pouvons voir que certains blocs texturés (en rouge) sont prédits comme étant chiffrés bien que cela soit faux. Ces blocs de pixels représentent $10,36\%$ et $1,56\%$ du nombre total de blocs de pixels avec le classifieur basé sur l'entropie locale et le classifieur CipherNet respectivement. Ces premiers résultats suggèrent que le classifieur CipherNet semble être plus efficace pour déterminer si un bloc de pixels est en clair ou chiffré.

Le tableau 6.4 indique les performances des deux classifieurs en renseignant les matrices de confusion et les mesures de la précision et du score-F1. Ces résultats sont obtenus en utilisant 1 638 400 blocs de pixels, dont 819 200 en clair et 819 200 chiffrés.

(a) Classifieur basé sur l'entropie locale.

(b) Classifieur CipherNet.

FIGURE 6.11 – Cartes de prédiction obtenues avec les deux classifieurs utilisés : les blocs de pixels prédits comme étant chiffrés sont colorisés en rouge, $score(b_i) \neq 1$.

Dans le tableau 6.4.a et le tableau 6.4.b, nous avons classé les blocs de pixels b_i comme suit :

- Positif ($score(b_i) = 1$) :
  - VP (Vrai Positif) : b_i est en clair et prédit comme étant en clair,
  - FP (Faux Positif) : b_i est en chiffré et prédit comme étant en clair,
- Négatif ($score(b_i) \neq 1$) :
  - VN (Vrai Négatif) : b_i est chiffré et prédit comme étant chiffré,
  - FN (Faux Négatif) : b_i est en clair et prédit comme étant chiffré.

Pour les deux classifieurs, nous pouvons voir qu'il n'existe aucun faux positif, ce qui signifie que si un bloc de pixels est chiffré, il n'est jamais prédit comme étant en clair. Cela est particulièrement important pour corriger les blocs de pixels chiffrés bruités. En effet, nous devons être en mesure d'identifier tous les blocs de pixels mal déchiffrés dans l'image directement déchiffrée. Nous pouvons aussi observer que la plupart des blocs de pixels en clair sont correctement prédits quel que soit le classifieur utilisé, en particulier avec le classifieur CipherNet (91%). Le tableau 6.4.c atteste également de l'efficacité des deux classifieurs et des meilleures performances obtenues par le classifieur CipherNet par rapport au classifieur basé sur l'entropie locale (précision de $0,8054$ *vs* $0,5784$ en moyenne). De plus, notons qu'aucun des deux classifieurs n'est capable de prédire parfaitement tous les blocs de pixels. Ainsi, utilisés seuls, ils ne permettent pas de corriger les images chiffrées bruitées : ils doivent être intégrés à notre algorithme de correction.

6.5.3 Exemple complet de la méthode proposée

En fig. 6.12, nous illustrons la méthode proposée sur l'image de *Lena* de taille 512×512 pixels codés sur 256 niveaux de gris (fig. 6.12.a). Tout d'abord, nous représentons les

Basé sur l'entropie locale	
VP = 80%	FP = 0%
FN = 20%	VN = 100%

(a)

CipherNet	
VP = 91%	FP = 0%
FN = 9%	VN = 100%

(b)

	Classifieur			
	Basé sur l'entropie locale		CipherNet	
	Précision	Score-F1	Précision	Score-F1
Min.	0,2235	0,3654	0,3020	0,4639
Max.	0,9996	0,9998	1	1
Moyenne	0,5784	0,7092	0,8054	0,8798
Q1	0,3996	0,5710	0,6823	0,8112
Médiane	0,5419	0,7029	0,8488	0,9182
Q3	0,7466	0,8549	0,9673	0,9834

(c)

TABLE 6.4 – Mesures de la performance des deux classifieurs : a) et b) Matrices de confusion, c) Précision et score-F1. Résultats obtenus en utilisant 1 638 400 blocs de pixels, 819 200 sont en clair et 819 200 sont chiffrés.

cartes de prédiction obtenues en utilisant les deux classifieurs présentés en section 6.5.2. La fig 6.12.b et la fig. 6.12.c sont les cartes de prédiction obtenues avec le classifieur basé sur l'entropie locale et le classifieur CipherNet respectivement. Dans les deux figures, nous représentons en rouge les blocs de pixels prédits comme étant chiffrés, *i.e.* mal prédits ($score(b_i) \neq 1$). Comme nous l'avons expliqué en section 6.5.2, ces blocs de pixels sont localisés dans les zones texturées et sur les contours. De plus, nous pouvons voir que davantage de blocs de pixels sont mal prédits en utilisant le classifieur basé sur l'entropie locale qu'avec le classifieur CipherNet (2595 blocs de pixels (16%) *vs* 539 blocs de pixels (3%)). La fig. 6.12.d est l'image chiffrée associée à la fig. 6.12.a en utilisant le mode de chiffrement CFB-puis-ECB. Notons que le contenu de l'image originale n'est pas visible, comme indiqué par une très faible valeur du PSNR ($8,55\ dB$). Pendant sa transmission, cette image chiffrée est bruitée aléatoirement avec un BER de $2,6 \times 10^{-3}$, qui altère aléatoirement un bit tous les trois blocs de pixels en moyenne (fig. 6.12.e). Cette introduction de bruit n'a pas d'impact sur la confidentialité du contenu de l'image originale (PSNR $= 8,54\ dB$). De plus, l'image chiffrée représentée en fig. 6.12.d et sa version bruitée en fig. 6.12.e sont sensiblement différentes, comme indiqué par un PSNR de $33,54\ dB$. Si l'image chiffrée bruitée de la fig. 6.12.e est directement déchiffrée, comme présenté en fig. 6.12.f, de nombreux blocs de pixels sont mal déchiffrés (5028 blocs de pixels (31%), encadrés en rouge). Ainsi, même lorsque la clé secrète utilisée lors du chiffrement est connue, le contenu de l'image originale ne peut pas être reconstruit sans erreur à cause du bruit. Un PSNR de $13,73\ dB$ entre la fig. 6.12.a et la fig. 6.12.f montre également la nécessité d'appliquer notre méthode de correction d'images chiffrées bruitées pendant la phase de décodage. Notons que les blocs de pixels mal déchiffrés sont toujours par paire du fait de l'utilisation du mode de chiffrement CFB-puis-ECB.

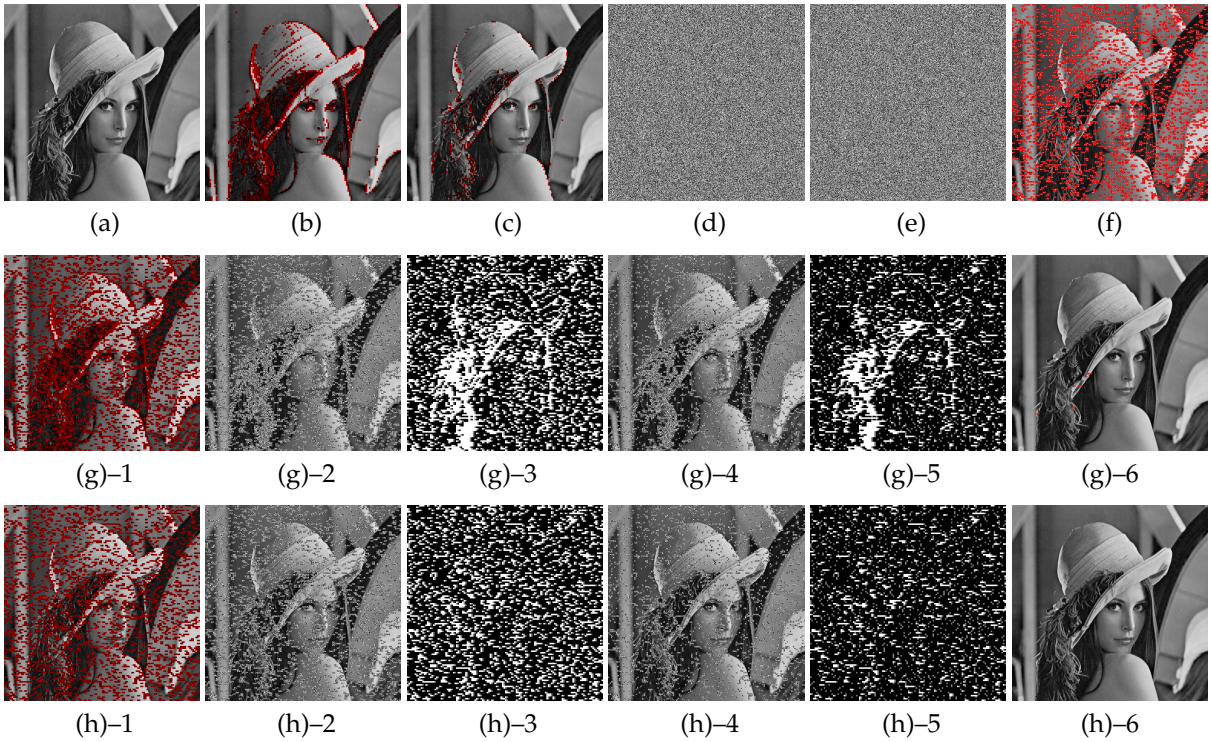


FIGURE 6.12 – Illustration de la méthode proposée de correction des images chiffrées bruitées : a) Image originale de *Lena*. Cartes de prédiction obtenues d'après (a) : b) avec le classifieur basée sur l'entropie locale (en rouge : blocs de pixels prédits comme étant chiffrés, $score(b_i) \neq 1$), c) avec le classifieur CipherNet, d) Image chiffrée associée à (a) en utilisant le mode de chiffrement CFB-puis-ECB, e) Image chiffrée bruitée obtenue d'après (d), f) Image directement déchiffrée sans correction obtenue d'après (e) (encadrés en rouge : blocs de pixels mal déchiffrés).
Correction en utilisant : g) le classifieur basé sur l'entropie locale, h) le classifieur CipherNet : 1) Carte de prédiction obtenue d'après (f) (en rouge : blocs de pixels prédits comme étant chiffrés, $score(b_i) \neq 1$), 2) Image reconstruite d'après (e) après la phase d'initialisation, 3) Carte des états des blocs de pixels obtenue d'après (g)-2 (en noir : $etat(i) = 0$, en gris : $etat(i) = 1$, en blanc : $etat(i) = 2$), 4) Image reconstruite d'après (e) après la phase d'initialisation et une itération de la phase de correction, 5) Carte des états des blocs de pixels obtenue d'après (g)-4 (en noir : $etat(i) = 0$, en gris : $etat(i) = 1$, en blanc : $etat(i) = 2$), 6) Image reconstruite finale d'après (e) après la phase d'initialisation et la phase de correction (27 itérations avec le classifieur basé sur l'entropie locale et 12 itérations avec le classifieur CipherNet) (encadrés en rouge : blocs de pixels mal déchiffrés).

En effet, quand un bloc de pixels est mal déchiffré à cause de l'altération par le bruit, son voisin est aussi mal déchiffré à cause du phénomène de propagation du bruit.

La fig. 6.12.g et la fig. 6.12.h illustrent les résultats obtenus en utilisant notre algorithme de correction avec le classifieur basé sur l'entropie locale et le classifieur CipherNet respectivement. La fig. 6.12.g-1 et la fig. 6.12.h-1 montrent les cartes de prédiction obtenues en utilisant les deux classifieurs sur l'image directement déchiffrée présentée en fig. 6.12.f. Nous pouvons voir que tous les blocs de pixels mal déchiffrés

sont correctement identifiés comme étant encore chiffrés. Cependant, notons que certains blocs de pixels en clair sont prédits comme étant chiffrés, en particulier lorsque le classifieur basé sur l'entropie locale est utilisé. En effet, ces blocs de pixels font partie de ceux identifiés en fig. 6.12.b et en fig. 6.12.c. La fig. 6.12.g-2 et la fig. 6.12.h-2 montrent les images reconstruites obtenues à partir de l'image chiffrée bruitée (fig. 6.12.e) après la phase d'initialisation. La fig. 6.12.g-3 et la fig. 6.12.h-3 illustrent alors les cartes des états des blocs de pixels associées. En fig. 6.12.g-2 et en fig. 6.12.h-2, tous les blocs de pixels identifiés comme étant en clair sont représentés en clair et les blocs de pixels possiblement mal déchiffrés restent dans le domaine chiffré. Notons que certains des blocs de pixels mal prédits (voir fig. 6.12.g-1 et fig. 6.12.h-1) sont identifiés comme étant en clair en utilisant l'information que les blocs de pixels précédent et suivant sont en clair. Si nous comparons la longueur des séquences de blocs de pixels possiblement mal prédits entre la fig. 6.12.g-3 et la fig. 6.12.h-3, nous pouvons observer que le classifieur CipherNet est plus efficace que le classifieur basé sur l'entropie locale. De plus, la longueur de la plus grande séquence indique le nombre nécessaire d'itérations pour corriger l'intégralité de l'image chiffrée bruitée pendant la deuxième phase de notre algorithme (phase de correction). La fig. 6.12.g-4 et la fig. 6.12.h-4 sont les images reconstruites obtenues d'après l'image chiffrée bruitée (fig. 6.12.e) après une itération de la phase de correction. La fig. 6.12.g-5 et la fig. 6.12.h-5 sont les cartes des états des blocs de pixels associées. Nous pouvons voir que tous les blocs de pixels de la fig. 6.12.g-2 et de la fig. 6.12.h-2 tels que $etat(i) = 1$ (*i.e.* représentés en gris sur les cartes des états des blocs de pixels) sont alors correctement déchiffrés et représentés en clair dans la fig. 6.12.g-4 et la fig. 6.12.h-4. De plus, dans la fig. 6.12.g-5 et la fig. 6.12.h-5, leur état est mis à zéro ($etat(i) = 0$) et l'état du bloc précédent est mis à 1 ($etat(i-1) = 1$). La fig. 6.12.g-6 et la fig. 6.12.h-6 sont les images reconstruites finales à la fin de la phase de correction. La fig. 6.12.g-6 est obtenue en utilisant le classifieur basé sur l'entropie locale après 27 itérations pendant la phase de correction. Nous pouvons voir que la plupart des blocs de pixels ($99,93\%$) sont correctement reconstruits mais 12 autres (6 paires de blocs de pixels), très texturés en clair, restent mal déchiffrés. Le PSNR entre l'image reconstruite et l'image originale de la fig. 6.12.a est égal à $40,43\ dB$. D'autre part, la fig. 6.12.h-6 est obtenue en utilisant le classifieur CipherNet après seulement 12 itérations lors de la phase de correction. Dans ce cas, tous les blocs de pixels sont correctement reconstruits et l'image finale correspond parfaitement à l'image originale en fig. 6.12.a (PSNR $\rightarrow +\infty$). Ainsi, cela montre qu'utiliser le classifieur CipherNet dans notre algorithme est la meilleure option pour corriger les images chiffrées bruitées.

6.5.4 Résultats obtenus sur une grande base de données

Nous avons appliqué notre algorithme de correction des images chiffrées bruitées sur 100 images en niveaux de gris (512×512 pixels) tirées aléatoirement dans la base BOWS-2 [3]. Ces images ont de fortes variabilités statistiques en termes de contenu. Le tableau 6.5 montre les résultats obtenus avec les deux classifieurs. Quel que soit le classifieur utilisé, nous pouvons voir que la plupart des blocs de pixels des images chiffrées bruitées sont correctement reconstruits. En effet, en moyenne, $99,53\%$ des blocs de pixels en utilisant le classifieur basé sur l'entropie locale et $99,85\%$ des blocs de pixels en utilisant le classifieur CipherNet sont correctement reconstruits. De plus, dans le pire

scénario (images très texturées), plus de 92% des blocs de pixels sont tout de même bien reconstruits ($92,96\%$ en utilisant le classifieur basé sur l'entropie locale et $95,80\%$ en utilisant le classifieur CipherNet). Nous pouvons aussi remarquer que 26 images parmi 100 en utilisant le classifieur basé sur l'entropie locale et 51 images parmi 100 en utilisant le classifieur CipherNet sont parfaitement reconstruites, ce qui signifie qu'il n'existe aucune erreur. Enfin, nous pouvons conclure que le classifieur CipherNet est plus efficace dans notre cas d'application.

	Classifieur	
	Basé sur l'entropie locale	CipherNet
Min.	92,96	95,80
Average	99,53	99,85
Max.	100	100
Q1	99,51	99,94
Median	99,91	100
Q3	100	100
Nombre d'images parfaitement reconstruites	26	51

TABLE 6.5 – Pourcentage de blocs de pixels correctement reconstruits par image en utilisant notre méthode avec les deux classifieurs (resultats obtenus sur 100 images (512×512 pixels, *i.e.* 16384 blocs) choisis aléatoirement dans la base BOWS-2 [3]).

La fig. 6.13 illustre des exemples de zones d'images difficiles à reconstruire quel que soit le classifieur utilisé. Notons que, dans ces zones, les blocs de pixels des images originales n'obtiennent pas un score de classification égal à 1. Ainsi, les blocs de pixels reconstruits associés sont ceux dont le score est maximal parmi toutes les configurations possibles. Comme illustré sur les première et deuxième lignes, le classifieur CipherNet est plus efficace que le classifieur basé sur l'entropie locale pour discriminer les blocs avec un motif et les blocs chiffrés. Cependant, sur la troisième ligne, nous pouvons voir que lorsqu'une zone comporte du texte, les résultats obtenus avec le classifieur basé sur l'entropie locale sont plus intéressants que ceux avec le classifieur CipherNet. En effet, l'étape de quantification avant le calcul de l'entropie locale permet d'améliorer les résultats obtenus pour ce genre de motifs. Enfin, la dernière ligne présente une zone très texturée (branches d'arbres) de l'image originale. Nous pouvons voir que les deux classifieurs ne permettent pas de reconstruire parfaitement tous les blocs de pixels. De plus, nous remarquons également que les blocs de pixels mal reconstruits ne sont pas les mêmes avec les deux classifieurs. Ainsi, il pourrait être intéressant d'essayer de combiner leurs performances pour améliorer la qualité de la reconstruction.

6.5.5 Comparaison des performances avec d'autres méthodes de correction

Nous avons comparé la méthode proposée avec d'autres approches de correction d'erreur préservant le format : une approche standard basée sur l'utilisation des codes de Reed-Solomon (RS) [85] sans expansion de la taille, la méthode d'Islam *et al.* [56] et une méthode basée sur une utilisation simple de l'entropie locale décrite dans la section 6.2 [108]. Les codes de Reed-Solomon sont notés RS($\mathbf{n}$, $\mathbf{k}$) avec des symboles de

$\mathbf{m}$ bits (pour des images sur 256 niveaux de gris, $\mathbf{m} = 8$ bits). L'encodeur RS prend en entrée $\mathbf{k}$ symboles de $\mathbf{m}$ bits et ajoute des symboles de parité pour construire un code de $\mathbf{n}$ symboles de $\mathbf{m}$ bits. Pendant la phase de décodage, près de $\mathbf{t} = \frac{\mathbf{n}-\mathbf{k}}{2}$ symboles erronés peuvent être corrigés, sans information sur l'emplacement des erreurs. Pour une image en niveaux de gris de 512×512 pixels, pour pouvoir corriger 1 bit par bloc de 4×4 pixels, des codes RS(255, 251) avec des symboles de 8 bits sont utilisés.

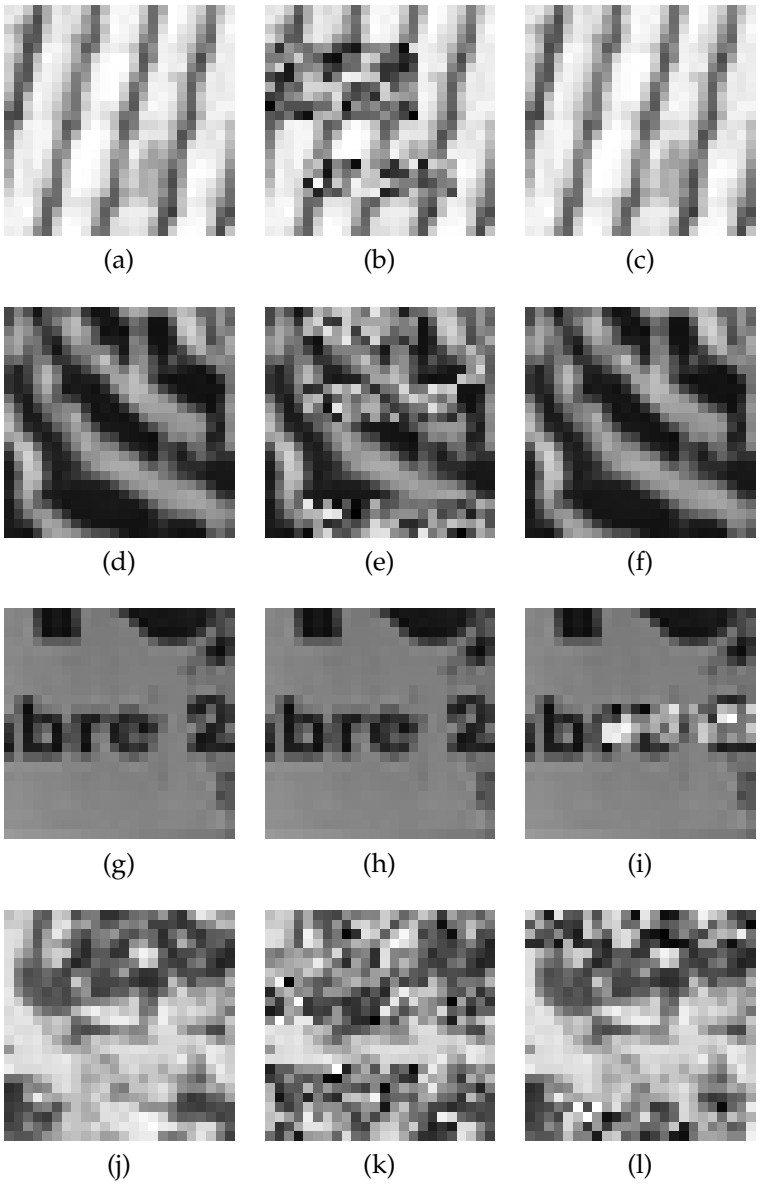


FIGURE 6.13 – Comparaisons des résultats obtenus en utilisant notre méthode avec les deux classifieurs : lignes) Différentes zones de 24×24 pixels générées à partir d'images de la base BOWS-2 [3] ; zones issues de : première colonne) Images originales, deuxième colonne) Images reconstruites en utilisant le classifieur basé sur l'entropie locale, troisième colonne) Images reconstruites en utilisant le classifieur CipherNet.

En effet, avec ces paramètres, près de $\mathbf{t} = 2$ octets (16 bits) parmi 16 blocs de 4×4 pixels ($\mathbf{n} = 251 \simeq 16 \times (4 \times 4)$) peuvent être corrigés. Pour éviter une expansion de la

taille de 4 kB, des bits de l'image chiffrée sont remplacés par les codes correcteurs d'erreurs calculés. Ainsi, $98,47\%$ des blocs de pixels sont bien reconstruits. Cependant, avec la méthode proposée, comme présenté dans le tableau 6.5, nous obtenons de meilleurs résultats. En effet, $99,53\%$ des blocs de pixels sont bien reconstruits en utilisant notre méthode avec le classifieur basé sur l'entropie locale et $99,85\%$ avec le classifieur CipherNet.

L'approche d'Islam *et al.* [56], celle décrite dans [108] et la méthode proposée maintiennent les données dans le format d'origine et préservent la taille originale pour la même quantité théorique de bits corrigés par bloc. Par ailleurs, notons que le mode de chiffrement CFB-puis-ECB est plus sécurisé que le mode ECB utilisé dans l'approche [108]. Ainsi, la méthode proposée de correction d'images chiffrées bruitées est plus adaptée aux applications pratiques que celle décrite dans [108]. De plus, un plus grand nombre d'images parfaitement reconstruites est obtenu avec la méthode proposée qu'avec l'approche d'Islam *et al.* [56] appliquée aux images en niveau de gris. En effet, d'après leurs expérimentations faites sur 100 images, 3 images sont parfaitement reconstruites avec la variance globale, 20 avec la somme des variances locales et 37 avec la somme des différences au carré [56], tandis qu'avec l'approche proposée, selon le tableau 6.5, 26 images sont parfaitement reconstruites avec le classifieur basé sur l'entropie locale et 51 avec le classifieur CipherNet.

6.6 Conclusion

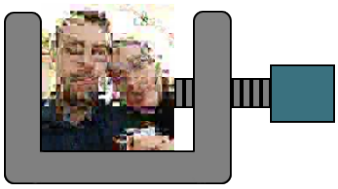
Dans ce chapitre, nous avons décrit la problématique d'altération par un bruit des images chiffrées pendant leur transmission ou leur stockage. Nous avons alors proposé une méthode efficace de correction d'images chiffrées bruitées impliquant l'utilisation d'une nouvelle approche de chiffrement à l'aide d'un nouveau mode appelé CFB-puis-ECB. Le mode de chiffrement CFB-puis-ECB combine les modes classiques CFB et ECB pour un chiffrement d'images avec l'algorithme AES. En utilisant ce nouveau mode, nous avons montré que si un bloc de pixels de l'image chiffrée est bruité, alors les blocs de pixels courant et suivant sont mal reconstruits durant la phase de déchiffrement. Ce phénomène de propagation du bruit est exploité pour la correction de l'image chiffrée bruitée. En effet, deux blocs de pixels mal reconstruits qui se suivent sont plus simples à détecter qu'un bloc isolé. Notre algorithme de correction des images chiffrées bruitées comprend deux étapes principales, à savoir : l'étape d'initialisation et d'analyse des blocs de pixels et l'étape de correction. Un classifieur permettant de discriminer les blocs en clair des blocs chiffrés est utilisé dans ces deux étapes. Avec en moyenne plus de $99,5\%$ de blocs de pixels correctement reconstruits quel que soit le classifieur utilisé, notre méthode est efficace pour corriger les images chiffrées bruitées.

Selon nos résultats expérimentaux, il pourrait être intéressant de combiner les résultats obtenus avec les deux classifieurs dans le but d'améliorer l'efficacité de la correction. En effet, il n'est pas toujours possible de reconstruire parfaitement le contenu de l'image originale après le déchiffrement. De plus, les blocs de pixels mal reconstruits ne sont pas nécessairement les mêmes suivant le classifieur utilisé. Dans des travaux futurs, nous pourrons également étendre l'algorithme de correction aux images couleur chiffrées. Dans ce cas, la corrélation entre les composantes RGB peut aussi être exploitée lors

des deux étapes de l'algorithme pour améliorer ses performances. Par ailleurs, dans la méthode présentée, nous considérons que seul un bit (au maximum) a été corrompu par le bruit dans chaque bloc de pixels de l'image chiffrée. Néanmoins, si la quantité de bruit est plus importante ou si le bruit n'est pas uniformément distribué, plus d'un bit par bloc de pixels peut être altéré. Ainsi, nous sommes intéressés par l'investigation de ce dernier point dans nos travaux de recherche à venir.

Ces travaux ont fait l'objet de quatre publications internationales (dont un article soumis et un article en révisions majeures) et une publication nationale. Tout d'abord, l'adaptation du calcul de l'entropie de Shannon pour effectuer une mesure significative dans des petits blocs de pixels a été présentée lors de la conférence internationale IPTA 2017 dans une application à l'IDCDC [103], puis lors de la conférence nationale CORESA 2017 et de la conférence internationale EUSIPCO 2018 dans une application à la correction d'images chiffrées bruitées [104, 108]. De plus, un premier classifieur pour discriminer les blocs de pixels en clairs des blocs de pixels chiffrés basé CNN et proche de CipherNet a été décrit dans un article présenté à la conférence internationale IPTA 2020 [110]. Par ailleurs, la méthode complète de correction d'images chiffrées bruitées impliquant l'utilisation d'une approche de chiffrement à l'aide du mode CFB-puis-ECB a été détaillée dans la revue internationale IEEE Transactions of Circuits and Systems for Video Technology en 2020 [111].

CHAPITRE 7



Recompression d'images JPEG crypto-compressées

Sommaire

7.1 Introduction

Dans le chapitre 2, nous avons vu que le chiffrement d'images est une méthode efficace pour assurer la confidentialité visuelle de leur contenu, tout en préservant leur format et leur taille. En introduisant de l'aléa, il empêche une personne non autorisée d'accéder au contenu original d'une image en clair. Selon l'application et le niveau de sécurité désiré, le chiffrement peut être total lorsque l'intégralité des données originales est chiffrée ou sélectif lorsque seulement une partie des données est sélectionnée pour être protégée.

La croissance rapide de l'utilisation des réseaux impose un besoin de plus en plus grand en bande passante. Cependant, limiter l'utilisation de bande passante est un enjeu de productivité numérique et écologique actuel. Depuis plus de 25 ans, la compression JPEG [156] (*Joint Photographic Experts Group*) est le format de compression le plus utilisé pour le stockage et le partage d'images numériques depuis sa dernière norme ISO/CEI 10918-1 UIT-T Recommendation T.81. publiée en 1993. Actuellement, son utilisation élevée est surtout dûe à son historique et reste privilégiée dans de nombreux cas d'application. En effet, la vaste majorité des afficheurs, notamment les navigateurs internet, sont compatibles avec les nombreuses versions et extensions de JPEG (XR, XT,...). En outre, l'IJG écrit et distribue une librairie gratuite (libjpeg), performante et régulièrement mise à jour, qui est largement utilisée [55]. L'efficacité JPEG résulte de la compression avec perte et l'utilisation de codes à redondance minimale. La dégradation de la qualité liée à la compression JPEG peut cependant limiter son utilisation dans les domaines de haute précision comme en imagerie médicale, satellite ou militaire. Les techniques de codage hiérarchique et sans perte sont moins couramment utilisées et nous ne les considérons pas ici.

Afin d'exploiter à la fois l'efficacité de la compression et du chiffrement, des méthodes de chiffrement conformes au format sont conçues pour produire un contenu chiffré compatible aux spécifications du format. Ces méthodes sont appelées méthodes de crypto-compression. Les premières méthodes de crypto-compression JPEG sont apparues au début des années 2000. Les approches proposées s'inspirent des méthodes de chiffrement d'images classiques. Cependant, l'application directe au cas d'images compressées JPEG n'est pas triviale. En effet, le format JPEG ne permet pas la manipulation simple des pixels de l'image. Il faut tenir compte des spécificités du format tout en appliquant les bases de la cryptographie (décrites dans le chapitre 2).

En section 7.2, nous introduisons le format JPEG, ses propriétés ainsi que les notations utilisées dans ce chapitre. En section 7.3, une étude de l'état-de-l'art de la crypto-compression JPEG est présentée. Ensuite, dans la section 7.4, nous décrivons notre méthode de recompression d'images crypto-compressées. Les résultats expérimentaux associés à cette méthode sont détaillés en section 7.5. Enfin, dans la section 7.6, nous concluons ce chapitre.

7.2 Compression JPEG

La norme JPEG spécifie les phases de compression et décompression JPEG dont les étapes sont illustrées en fig. 7.1. La compression se fait en suivant six étapes principales.

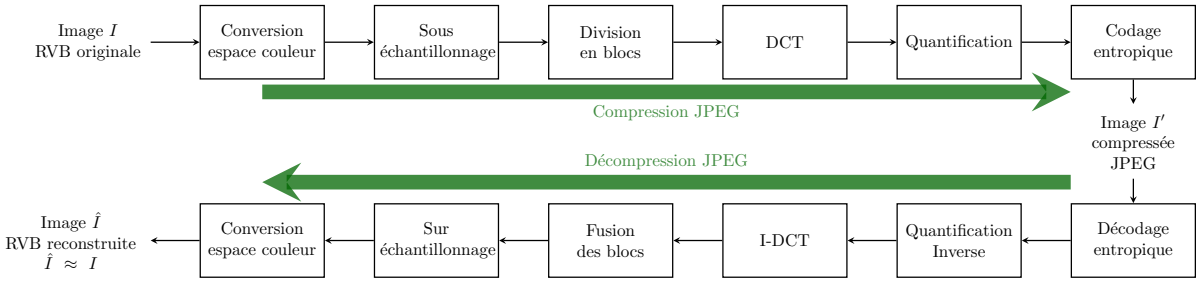


FIGURE 7.1 – Étapes de compression et décompression JPEG.

Dans un premier temps, l'espace colorimétrique de l'image est transformé de l'espace RVB (Rouge, Vert et Bleu) dans un autre espace de couleur qui décorrèle la luminance et la chrominance, par exemple YCrCb. Les plans monochromes peuvent être sous-échantillonnés (verticalement, horizontalement ou dans les deux directions, sur la chrominance et/ou la luminance). Chaque plan est ensuite divisé en blocs de taille $N \times N$ (typiquement $N = 8$) ne se superposant pas et chaque bloc est projeté dans un espace fréquentiel en utilisant la transformée cosinus discrète notée DCT (*Discrete Cosine Transform*). Considérons un bloc B de taille $N \times N$ de l'image I, alors la DCT est :

$$F(u,v) = \frac{1}{\sqrt{2N}} C(u)C(v) \sum_{i=0}^{N} \sum_{j=0}^{N} p(i,j) \cos\left[\frac{(2i+1)u\pi}{2N}\right] \cos\left[\frac{(2j+1)v\pi}{2N}\right], \tag{7.1}$$

avec $p(i,j),\ 0 \leq i,j < N$ les pixels de B, $F(u,v),\ 0 \leq u,v < N$ les coefficients fréquentiels et $C(\alpha) = \frac{1}{\sqrt{2}}$ si $\alpha = 0$ et 1 sinon.

Dans les implémentations pratiques, le calcul de la DCT est optimisé en utilisant les résultats intermédiaires et peut être effectué de façon plus ou moins rapide avec plus ou moins d'approximation (voir les modes *fast*, *slow*, *float* de libjpeg [55]).

Nous distinguons deux types de coefficients DCT, à savoir les coefficients DC et AC. Le coefficient DC, noté $F(0,0)$, est proportionnel à la valeur moyenne des pixels au sein du bloc. Les coefficients AC, notés $F(u,v)$, avec $0 \leq u,v < 8$ et $(u,v) \neq (0,0)$, sont reliés aux fréquences au sein du bloc. Notons que plus les coordonnées (u,v) sont élevées, plus les fréquences sont hautes. De plus, bien que les pixels soient des entiers, les coefficients DCT $F(u,v)$ sont des valeurs réelles.

Par la suite, chaque bloc de coefficients fréquentiels est quantifié en utilisant une table de quantification Q de taille $N \times N$ contenant les coefficients de quantification $q(u,v)$. L'opération de quantification est la division élément par élément des coefficients $F(u,v)$ du bloc B par les éléments de la matrice de quantification Q :

$$F'(u,v) = round\left(\frac{F(u,v)}{q(u,v)}\right), \tag{7.2}$$

où $round(\cdot)$ représente l'opérateur d'arrondi à l'entier le plus proche. Cette étape est effectuée afin de coder les coefficients DCT quantifiés $F'(u,v)$ sur des entiers 8 bits.

Les opérations de sous-échantillonnage et de quantification sont basées sur des études des capacités du SVH. En effet, les canaux sont souvent sous-échantillonnés, car le SVH est très peu sensible à la chrominance. D'autre part, la DCT décompose le signal en $N \times N$ fréquences de la plus basse à la plus élevée. Le SVH est sensible aux

basses fréquences qui représentent le contenu sommairement et est moins sensible aux hautes fréquences, les petits détails, qui peuvent donc être quantifiées davantage. En s'appuyant sur ce principe, l'IJG [55] a proposé des tables de quantification standards Q_{QF}, calculées à partir d'une table de référence Q_{QREF} de coefficients $q_{\text{QREF}}(u, v)$ (fig. 7.2), et d'un facteur de qualité entier noté $\text{QF} \mid \text{QF} \in [1, 100]$, où $\text{QREF} = 50$.

	0	1	2	3	4	5	6	7
0	16	11	10	16	24	40	51	61
1	12	12	14	19	26	58	60	55
2	14	13	16	24	40	57	69	56
3	14	17	22	29	51	87	80	62
4	18	22	37	56	68	109	103	77
5	24	35	55	64	81	104	113	92
6	49	64	78	87	103	121	120	101
7	72	92	95	98	112	100	103	99

FIGURE 7.2 – Table de quantification standard pour la luminance $Q_{\text{QREF}} = Q_{50}$.

Les coefficients $q_{\text{QF}}(u, v)$ de chaque table Q_{QF} sont calculés, à partir de la table de quantification de la fig. 7.2, de la façon suivante :

$$q_{\text{QF}}(u,v) = \begin{cases} \left\lfloor \frac{q_{\text{QREF}}(u,v) \times \left(\frac{5000}{\text{QF}}\right) + 50}{100} \right\rfloor, & \text{si QF} < 50, \\ \left\lfloor \frac{q_{\text{QREF}}(u,v) \times (200 - 2\text{QF}) + 50}{100} \right\rfloor, & \text{sinon.} \end{cases} \tag{7.3}$$

Afin de suivre les recommandations de l'IJG, les coefficients $q_{\text{QF}}(u, v)$ doivent être des entiers compris entre 1 et 255. Sous cette contrainte, nous avons alors :

$$q_{\text{QF}}(u,v) = \begin{cases} 1, & \text{si } q_{\text{QF}}(u,v) < 1, \\ 255, & \text{si } q_{\text{QF}}(u,v) > 255, \\ q_{\text{QF}}(u,v), & \text{sinon.} \end{cases} \tag{7.4}$$

Pour $\text{QF} = 100\%$, tous les coefficients de la table Q_{100} sont à 1. Notons que, même avec cette très haute qualité, il existe toujours une perte d'information du fait de l'arrondi à la partie entière inférieure (équation (7.2)). En pratique, la plupart des applications proposent leurs propres tables comme Adobe Photoshop. Les tables étant incluses dans l'en-tête JPEG, le décodeur ne requiert aucun *a priori*. Le principal problème de la compression JPEG avec un facteur de qualité faible est la dégradation de la texture, l'apparition de grain et d'artefacts de bloc [11].

Finalement, les blocs sont compressés, de façon séquentielle ou progressive, à l'aide d'un des deux codages entropiques spécifiés, un codage de Huffman ou un codage arithmétique. Le flux binaire produit par le codec est le plus souvent encapsulé dans un format d'échange Exif [60] (*Exchangeable image file format*) ou JFIF [57] (*JPEG File Interchange Format*) qui fixe la forme de l'entête et les marqueurs binaires délimitant les champs des paramètres.

La décompression consiste à inverser les opérations effectuées lors de la compression. Les opérations de sous-échantillonnage et de quantification n'étant pas réversibles, elles sont les deux causes principales de la perte d'information lors de la compression JPEG. La quantification inverse consiste à inverser l'équation (7.2), et par conséquent à multiplier les coefficients DCT quantifiés $F'(u,v)$ par leurs quantificateurs de fréquence correspondant $q_{\text{QF}}(u,v)$:

$$\hat{F}(u,v) = F'(u,v) \times q_{\text{QF}}(u,v). \tag{7.5}$$

La transformée en cosinus inverse, notée I-DCT, permet alors de revenir dans le domaine spatial depuis le domaine fréquentiel.

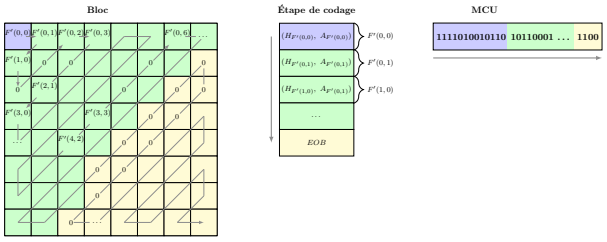


FIGURE 7.3 – Illustration de la construction du code d'un bloc sur un MCU (Unité de Code Minimale), le bloc est parcouru en zigzag pour le codage par plage zéros, puis les coefficients sont codés à l'aide des tables de Huffman. Le coefficient DC est en mauve, les coefficients AC quantifiés en vert. Après le dernier coefficient AC quantifié non nul, le reste du bloc est codé par un signal de fin de bloc EOB.

Dans ce chapitre, nous considérons le choix le plus standard fait pour les options JPEG ; il consiste à encoder une image RVB 24 bits dans un flux binaire ordonné de façon séquentielle, issu d'un codage de Huffman. La taille des blocs est de 8×8 pixels ($N = 8$). De plus, l'espace YCrCb est préféré avec un sous-échantillonnage standard des canaux de chrominance (4:2:0) ou (4:2:2), qui consiste respectivement à diviser par deux ou par quatre la taille des blocs en moyennant la valeur des pixels voisins. Les coefficients fréquentiels quantifiés de chaque bloc sont codés suivant une lecture zigzag sur un MCU (Unité de Code Minimale). Les coefficients sont alors ordonnés en fréquences croissantes et, puisque les hautes fréquences sont quantifiées plus fortement, les blocs tendent à se terminer par des zéros ce qui permet un codage plus efficace. L'étape d'encodage d'un bloc est illustrée, sur un exemple, dans la fig. 7.3. Plus généralement, le code du coefficient DC $F'(0,0)$, proportionnel à la valeur moyenne, correspond à une paire $(H_{F'(0,0)},\ A_{F'(0,0)})$. Le paramètre d'amplitude $A_{F'(0,0)}$ encode l'erreur de prédiction, tandis que le paramètre d'en-tête $H_{F'(0,0)}$ indique le nombre de bits nécessaires à la coder. Les coefficients AC quantifiés $F'(u,v)$, tels que $(u,v) \neq (0,0)$, sont codés par une paire $(H_{F'(u,v)},\ A_{F'(u,v)})$. $A_{F'(u,v)}$ encode l'amplitude du coefficient et l'entête $H_{F'(u,v)}$ est composé de la plage de zéros $\text{RL}_{H_{F'(u,v)}}$ – *run-length* – précédemment calculée, et d'un paramètre renseignant le nombre de bits nécessaires pour coder l'amplitude. Lorsque le dernier coefficient non-nul du bloc a été codé, un symbole indicateur de fin de bloc EOB (*End Of Block*) est ajouté au MCU. La séquence des MCU est placée après l'en-tête. Dans la suite de ce chapitre, par abus de notation, le code associé à un coefficient est noté comme le coefficient.

La fig. 7.4 illustre les images obtenues après l'application d'une compression JPEG avec différents QF à une image de la base UCID [129] (fig. 7.4.a). Notons qu'avec un haut QF, le taux de compression τ est déjà significatif tout en préservant une très bonne qualité de l'image originale (fig. 7.4.b, $\tau = 12,96$ et PSNR $= 41,46\ dB$ et fig. 7.4.c, $\tau = 23,35$ et PSNR $= 38,78\ dB$). Par ailleurs, le principal problème de la compression JPEG avec un QF faible est la dégradation de la texture, l'apparition de grain et d'artefacts de bloc [11], comme illustré en fig. 7.4.d avec QF $= 25\%$ ($\tau = 71,18$ et PSNR $= 32,82\ dB$).

(a) (b) (c) (d)

FIGURE 7.4 – Illustration de la compression JPEG : a) Image non compressée (590,8 Ko) de la base UCID [129], b) Image compressée avec JPEG et QF $= 90\%$ (45,6 Ko), c) Image compressée avec JPEG et QF $= 75\%$ (25,3 Ko), d) Image compressée avec JPEG et QF $= 25\%$ (8,3 Ko).

7.3 Crypto-compression d'images JPEG

Dans cette section, nous détaillons les grandes catégories de méthodes de crypto-compression compatibles au format JPEG. La conformité à d'autres formats peut être étendue. Nous dressons alors un-état-de-l'art à partir de méthodes sélectionnées par leurs performances et leur diversité.

Les premières méthodes de crypto-compression effectuaient séparément le chiffrement et l'étape de compression. Cependant, le principal problème de ces approches est que le chiffrement modifie considérablement les caractéristiques statistiques de l'image. En effet, les méthodes de chiffrement par substitution efficaces tendent à ce que la distribution de l'image chiffrée se rapproche d'une distribution uniforme de variables aléatoires. De plus, les méthodes de chiffrement par mélange suppriment les cohérences spatiales. Par conséquent, l'efficacité de la compression est fortement réduite si le chiffrement est effectué en premier lieu. Dans ce contexte, au cours des dernières années, l'intérêt autour du chiffrement d'images compressées JPEG s'est accru. Le but est d'établir un chiffrement de telle manière que les données chiffrées puissent toujours être représentées dans leur format original (propriété de préservation du format). Dans le format JPEG, toutes les données binaires n'ont pas besoin d'être chiffrées. Le chiffrement se concentre alors sur les données propres à l'image et non au format. A la décompression sans la clé de chiffrement, l'image sous forme de pixels est

chiffrée sélectivement puisque toute l'information n'est pas chiffrée. Pour ce nouveau type de méthodes, la compression et le chiffrement JPEG sont soit effectués conjointement, soit après compression. Nous pouvons lister trois catégories de méthodes : le chiffrement du bit de signe, le chiffrement des coefficients DCT, et les méthodes basées sur le mélange des coefficients DCT ou des blocs JPEG. Shi et Bhargava ont conçu une des premières approches de crypto-compression permettant de chiffrer directement le flux JPEG binaire [135]. Les auteurs proposent de chiffrer le bit de signe des coefficients AC et DC (le signe de la différence pour les coefficients DC). Une séquence pseudo-aléatoire binaire est générée en fonction d'une clé secrète et le chiffrement est effectué par un opération ou-exclusif de cette séquence avec la séquence de tous les bits de signe. Cette méthode préserve la structure JPEG de l'image, qui peut être visualisée avec des éditeurs d'images standards. De plus, le taux de compression n'est pas (ou peu) modifié puisque le nombre de bits est inchangé. Cependant, cette méthode a été montrée non sécurisée par Said [128]. Effectivement, comme la méthode respecte le format JPEG, même si l'image en clair semble chiffrée, la majorité du flux binaire reste inchangé. Il est donc facile de deviner la valeur originale des bits chiffrés. Dans le but d'augmenter la sécurité et la confidentialité visuelle, les méthodes récentes se sont concentrées sur l'augmentation des propriétés de confusion et de diffusion, en suivant deux approches : la crypto-compression par substitution et la crypto-compression par mélange.

7.3.1 Crypto-compression par substitution

La crypto-compression par substitution est une classe de méthodes où les valeurs en clair sont substituées par leurs valeurs chiffrées. En général, les valeurs sont chiffrées en utilisant l'opérateur ou-exclusif entre les valeurs en clair et une séquence pseudo-aléatoire générée à l'aide d'une clé secrète. Cette opération permet de rester compatible avec le format et à limiter l'expansion de la taille de fichier. Van Droogenbroeck et Benedett ont proposé de chiffrer les coefficients AC après la transformation DCT, mais pas les coefficients DC car ils contiennent des informations visibles importantes et sont prévisibles [154]. En suivant cette idée, Puech et Rodrigues ont proposé une méthode de chiffrement sélectif pour les images JPEG qui permet de chiffrer les coefficients DC et AC [101].

Dans cette méthode, illustrée dans la figure 7.5, tous les codes des coefficients DC (optionnels) et les coefficients AC non nuls sont concaténés pour former un flux binaire de 128 bits. Cette séquence binaire est alors chiffrée en utilisant l'algorithme AES [25] en mode OFB. Finalement, l'image JPEG est construite avec les valeurs chiffrées des coefficients. Puisque la taille des coefficients reste inchangée ainsi que le codage par plage de zéros, le codage de Huffman n'est pas modifié et le fichier JPEG produit est exactement de même taille que s'il avait été compressé sans chiffrement avec les mêmes paramètres. Au décodage sans déchiffrement, l'image est peu compréhensible pour le SVH. Si les coefficients DC sont chiffrés, alors l'image est incompréhensible. Par ailleurs, si seulement les coefficients AC sont chiffrés, alors l'image en basse résolution est accessible. Dans le but de protéger non pas l'intégralité de l'image mais seulement l'anonymat des personnes, Rodrigues *et al.* prolongent ce concept [125]. Le chiffrement est effectué sur des régions d'intérêts détectées automatiquement, comme la peau humaine. Ces régions sont détectées en utilisant les coefficients DC en clair des composantes couleur Cr et Cb.

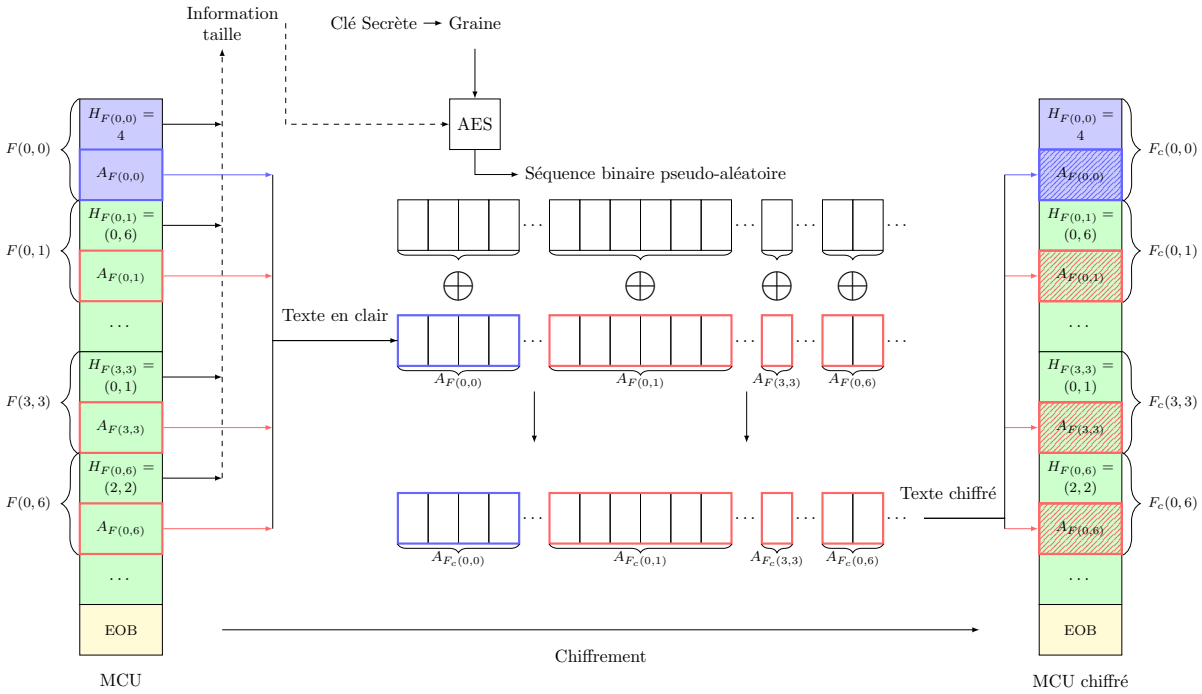


FIGURE 7.5 – Schéma de crypto-compression par substitution mis en place par Puech et Rodrigues [101], où $\oplus$ représente l'opérateur ou-exclusif.

Le chiffrement sélectif est appliqué aux blocs de la composante de luminance Y lors de la phase de codage entropique. Le SVH étant peu sensible à l'information de chrominance, elle est en général plus quantifiée et sous-échantillonnée. Le chiffrement des coefficients des canaux correspondant n'a alors qu'un faible impact. En utilisant le mode CFB de l'algorithme AES, les coefficients AC quantifiés de la région d'intérêt sont chiffrés. Cette méthode respecte le format JPEG et l'image JPEG crypto-compressée résultante a la même taille qu'après une compression JPEG classique. De plus, le chiffrement partiel est suffisant pour cacher des informations sensibles telles que du texte [95]. L'avantage de ces méthodes de chiffrement réside dans la conservation du taux de compression d'une compression JPEG classique avec les mêmes paramètres. Cependant, ces méthodes ne diffusent pas l'information par mélange, ce qui rend les images crypto-compressées obtenues plus vulnérables à une attaque par force brute.

7.3.2 Crypto-compression par mélange

La crypto-compression par mélange consiste, comme pour le chiffrement par mélange (chapitre 2), à réorganiser les éléments d'une image à l'aide d'un GNPA. Dans la situation d'une compression JPEG, les éléments les plus évidents à mélanger sont les blocs JPEG. Cela introduit une distorsion sur l'image au décodage du fait du codage prédictif des coefficients DC mais ne change pas la taille de l'image puisque les blocs sont codés indépendamment (à part les coefficients DC).

Cependant, les blocs restent en clair et la sécurité n'est pas assurée. Kurihara *et al.* ont conçu un système de chiffrement suivi d'une compression dans lequel les blocs sont permutés dans le domaine spatial [69]. Ce schéma peut être attaqué en utilisant un sol-

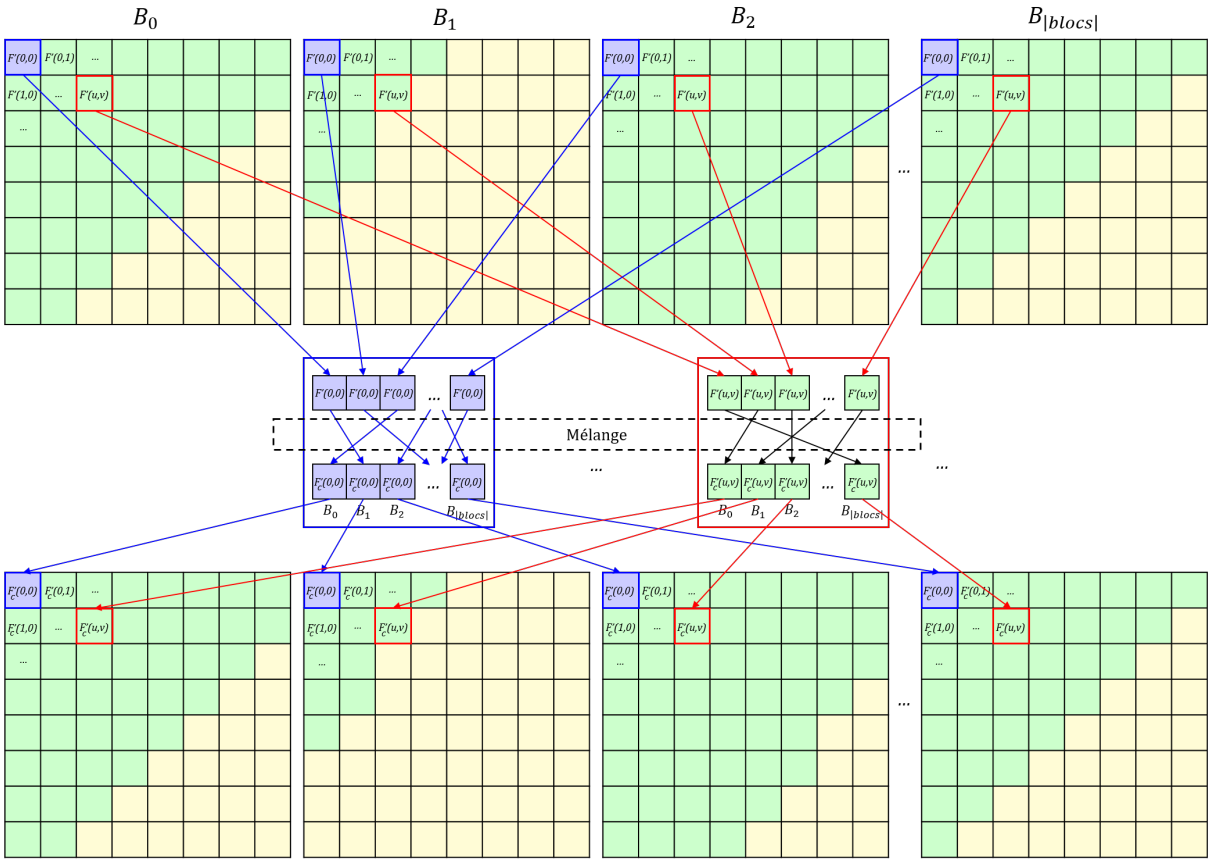
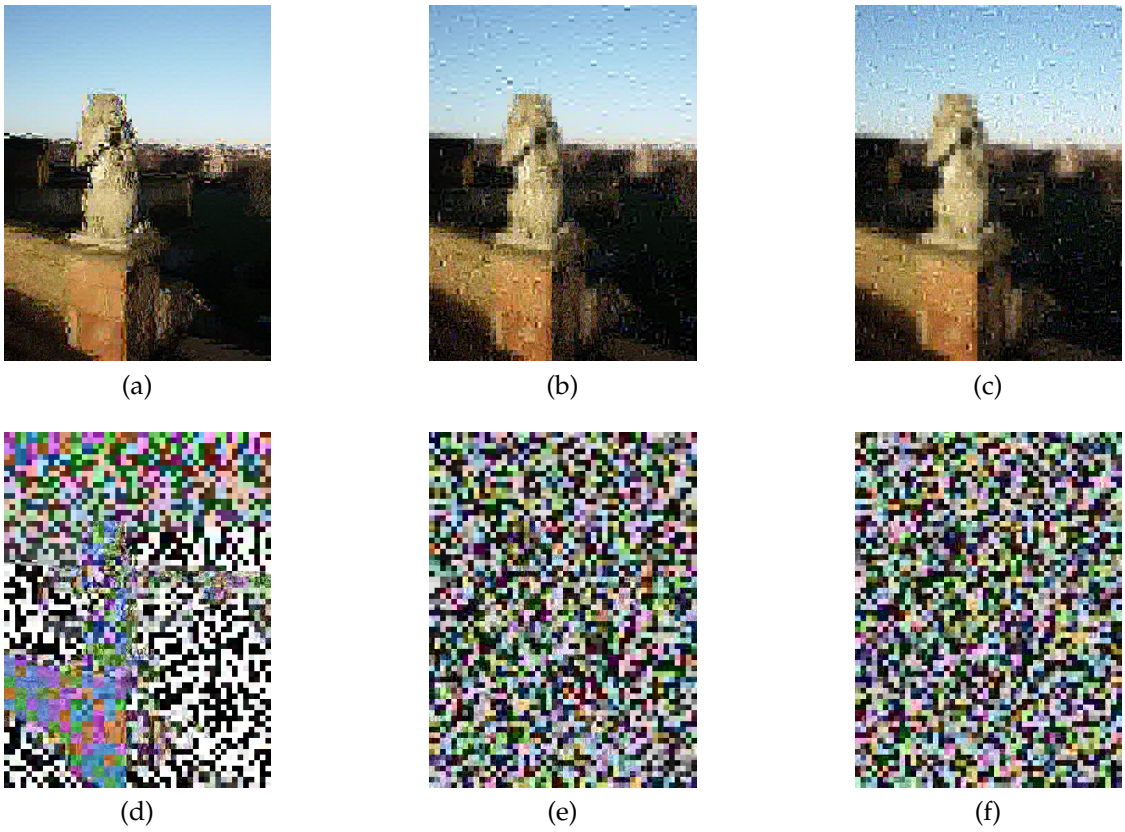


FIGURE 7.6 – Illustration de la méthode de mélange inter-bloc des coefficients fréquentiels (FIBS) proposée par Li et Yuan [73].

veur de puzzles, comme démontré par [19]. L'idée est alors d'utiliser d'autres éléments de l'image JPEG compressée, qui permettent une diffusion efficace et sécurisée de l'information, tout en limitant l'expansion de la taille du fichier JPEG. La distribution des coefficients de même fréquence suit une loi Laplacienne dans les images naturelles [71]. Il semble alors que le mélange des coefficients non-nuls de même fréquence permette une bonne sécurité tout en conservant globalement la taille du fichier. Le codage par plage de zéros est identique, mais la valeur du coefficient peut être modifiée, ce qui entraîne un changement du code de Huffman. Cependant, Li et Yuan [73] ont montré que cette approche pouvait être attaquée pour obtenir une approximation de l'image, ce qui ne garantit donc pas la confidentialité visuelle. Pour remédier à ce problème, ils ont proposé la méthode FIBS (*full inter-block shuffle*) qui consiste à mélanger les coefficients de même fréquence entre les blocs, comme illustré sur la figure 7.6. L'image décodée est alors inintelligible (si les coefficient DC sont chiffrés). Cependant, comme les coefficients sont mélangés, l'étape de codage entropique est moins performante qu'avec une compression JPEG classique. En effet, le codage par plage de zéros est moins efficace puisque les zéros successifs des blocs sont diffusés. Il peut en résulter le changement de position du code de fin de bloc : les MCU sont alors codés sur plus de bits. De plus, l'en-tête du code d'un coefficient peut être modifié en fonction de la taille de la plage de zéros le précédant. Notons que les zones non texturées sont alors impactées par le chiffrement du fait du mélange des zéros. Les méthodes par mélange ne permettent pas d'introduire

de confusion. En effet, après un mélange naïf, la distribution des coefficients DCT suit toujours la même distribution par fréquence [122].

La fig. 7.7 illustre les résultats obtenus suite à l'application de trois méthodes différentes de crypto-compression avec $\text{QF} = 90\%$ sur l'image de la base UCID [129] illustrée en fig. 7.4.a. Les images crypto-compressées de la première ligne ont été obtenues après avoir chiffré uniquement les coefficients AC des trois composantes YCrCb. Par ailleurs, sur celles de la deuxième ligne, les coefficients AC et DC ont été chiffrés. Pour générer les fig. 7.7.a et fig. 7.7.d, la méthode de Puech et Rodrigues par substitution a été appliquée [101]. La fig. 7.7.b et la fig. 7.7.e présentent les images crypto-compressées en mélangeant les coefficients non-nuls. Avec ces deux méthodes, notons que lorsque seuls les coefficients AC sont chiffrés, la taille des images crypto-compressées est la même que celle de l'image JPEG avec $\text{QF} = 90\%$ (fig. 7.4.b). Enfin, avec la méthode FIBS de Li et Yuan [73], l'expansion de la taille résulte de la perte d'efficacité du codage par plage de zéros (fig. 7.7.c de 59,5 Ko, et fig. 7.7.f de 63,3 Ko).



(a) (b) (c)

(d) (e) (f)

FIGURE 7.7 – Images crypto-compressées avec $\text{QF} = 90\%$ en chiffrant les coefficients AC (première ligne) et les coefficients DC et AC (deuxième ligne) des trois composantes YCrCb de l'image illustrée en fig. 7.4.a : a) et d) Avec la méthode de Puech et Rodrigues [101] (45,6 Ko et 48,9 Ko) ; b) et e) En mélangeant les coefficients non-nuls (45,9 Ko et 49,5 Ko) ; c) et f) Avec la méthode FIBS de Li et Yuan [73] (59,5 Ko et 63,3 Ko).

7.3.3 Crypto-compression hybride

Dans le but d'introduire les propriétés de confusion et de diffusion conjointement, certains auteurs se sont tournés vers des approches hybrides. De plus, les méthodes précédemment présentées ne permettent de chiffrer qu'une faible quantité d'information si le facteur de qualité est faible puisque peu de coefficients sont non-nuls. L'avantage des approches hybrides est qu'en superposant des étapes n'augmentant que peu la taille du ficher, l'augmentation globale est faible alors que la confidentialité visuelle est maîtrisée et la sécurité augmentée. Dans ce but, Minemura *et al.* [84] ont permuté les blocs JPEG. A l'intérieur de chaque bloc, les coefficients AC ayant une plage de zéros de taille nulle sont alors permutés. Ensuite, les bits de signe sont chiffrés avec une séquence pseudo-aléatoire par l'opération XOR pour augmenter les perturbations. Ces trois étapes n'augmentent pas la taille. Ensuite, les coefficients DC sont groupés par valeurs proches à l'aide d'une détection de contours dans le domaine fréquentiel pour être traités, ce qui peut augmenter le nombre de bits mais de manière assez limitée. Concrètement, si seulement les coefficients AC sont mélangés, alors les contours de l'image sont toujours perceptibles. Dans une approche similaire, Unterweger et Uhl décrivent une méthode de crypto-compression basée sur trois étapes [151]. La première étape consiste à permuter l'ordre des codes des coefficients (de la forme $(H_{F'(u,v)},\ A_{F'(u,v)})$) dans un MCU en utilisant l'algorithme AES [25] en mode OFB, comme illustré en fig. 7.8. Les coefficients non-nuls et les coefficients nuls codés ensemble sont mélangés, le mélange n'augmente pas la taille du code mais change la position des coefficients des fréquences. Les bits des valeurs des coefficients ($A_{F'(u,v)}$) sont ensuite inversés en fonction du retour (0 ou 1 binaire) du GNPA basé AES utilisé à l'étape précédente. Finalement, l'ordre des MCU partageant les mêmes tables de Huffman est modifié de façon pseudo-aléatoire, ce qui ajoute un niveau de sécurité en fonction du nombre de tables de Huffman. Cette méthode est sécurisée dans le sens où une attaque en force brute sur la clé AES est plus efficace que d'essayer toutes les combinaisons possibles.

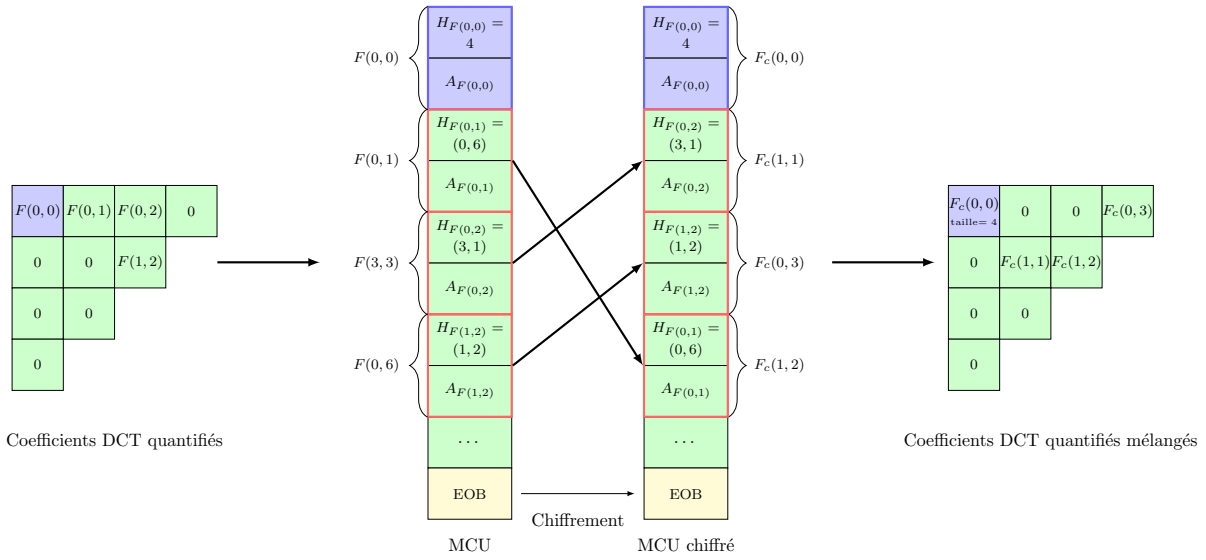


FIGURE 7.8 – Illustration de la méthode de mélange intra-bloc des codes des coefficients proposée par Unterweger et Uhl [151].

Les méthodes présentées dans cette section permettent de chiffrer des images JPEG de façon efficace en terme de confidentialité visuelle et de limitation de la taille du fichier. L'analyse de la sécurité des méthodes de crypto-compression comme effectuée dans l'approche de Unterweger et Uhl [151] n'était que peu faite auparavant, mais la tendance actuelle va dans le sens d'une étude systématique. Bien que la plupart des méthodes soient basées sur le format JPEG, et par conséquent sur la DCT, d'autres schémas existent et ont été popularisés au cours des dernières années. Effectivement, beaucoup de méthodes de crypto-compression d'images compatibles au format JPEG2000 sont présentées dans l'étude d'Engel *et al.* [34]. D'autres méthodes sont basées sur le chiffrement du signe des coefficients des ondelettes [31], sur les permutations [88, 77] ou sur le codage arithmétique rendu aléatoire [46]. Pour adapter ces méthodes à de la vidéo, Shahid *et al.* ont conçu une technique de chiffrement sélectif pour le codec vidéo H.264/AVC pour les normes CAVLC et CABAC [131]. Le chiffrement est effectué lors de la phase du codage entropique en utilisant l'algorithme AES en mode CFB. Dans le but de conserver la taille du fichier et de préserver le format H.264/AVC, le chiffrement est réalisé seulement sur les mots-clés CAVLC et les flux binaires CABAC. De plus, Dufaux et Ebrahimi ont proposé une méthode pour la protection de la vie privée dans les systèmes de vidéo-surveillance basée sur des permutations binaires [30]. Par ailleurs, une étude appronfondie des méthodes de crypto-compression HEVC a été réalisée récemment [48].

L'avantage des méthodes de crypto-compression est de pouvoir manipuler les images crypto-compressées comme des images JPEG classiques pour le partage, le stockage ou la visualisation. Cependant, certaines tâches ne sont plus possibles. En effet, les algorithmes de vision par ordinateur, d'amélioration de la qualité ou l'application d'une seconde compression nécessitent un déchiffrement. Dans ce contexte, des méthodes de chiffrement homomorphe sont apparues [37] et permettent d'effectuer des opérations directement dans le domaine chiffré, comme décrit dans le chapitre 3.

7.4 Méthode proposée

Dans cette partie, nous présentons notre méthode de recompression d'images JPEG crypto-compressées, sans utilisation de la clé secrète utilisée lors du chiffrement. Nous commençons par développer les différentes étapes de notre nouvelle méthode de crypto-compression, où la compression JPEG et le chiffrement sont réalisés de manière conjointe pendant le codage de Huffman, comme dans les travaux de Puech *et al.* [99]. Nous décrivons alors une méthode efficace pour recompresser les images JPEG crypto-compressées directement dans le domaine chiffré, sans avoir accès au contenu en clair de l'image originale. Notons que cette approche préserve le niveau de sécurité de la méthode de crypto-compression. Dans la section 7.4.1, nous donnons un aperçu général de la méthode proposée. La méthode de crypto-compression est détaillée en section 7.4.2 et notre approche proposée pour recompression les images crypto-compressées est décrite en section 7.4.3. Nous expliquons alors comment décoder l'image JPEG crypto-compressée après l'étape de recompression en section 7.4.4.

7.4.1 Description générale de la méthode

Dans l'approche proposée, nous commençons par appliquer les étapes d'une compression classique au format JPEG. Nous pouvons alors choisir de chiffrer seulement la composante de luminance Y. En effet, c'est elle qui porte l'information la plus significative quant au contenu de l'image. Pour un niveau de sécurité optimal, nous devons cependant considérer les trois composantes (Y, Cr et Cb) lors de l'étape de chiffrement. Par souci de conservation du taux de compression, nous chiffrons seulement les coefficients non-nuls après quantification [99]. Une image JPEG crypto-compressée est ainsi obtenue et nous nous intéressons à la recompression de cette image directement dans le domaine chiffré, sans connaître la clé secrète utilisée lors du chiffrement, ni le contenu original de l'image. Le schéma général de cette méthode est illustré en fig. 7.9.

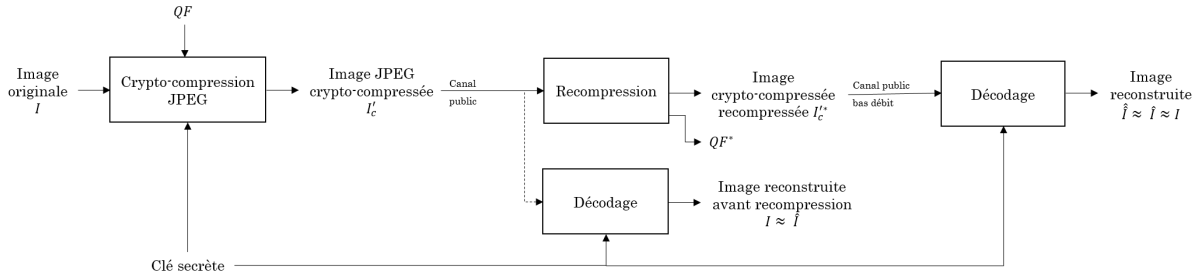


FIGURE 7.9 – Schéma général de la méthode de recompression d'images JPEG crypto-compressées.

Supposons qu'Alice ait recours à un système de crypto-compression JPEG dans le but de se prémunir contre une utilisation malicieuse de ses images et par souci d'utiliser le moins d'espace possible. Alice souhaite stocker ses images sur sa plateforme de *cloud* dans le but de pérenniser ses données et pouvoir y accéder depuis n'importe où. La crypto-compression JPEG est adaptée puisqu'elle permet une certaine confidentialité par rapport aux attaques de la plateforme ou à l'analyse des données que l'hébergeur, lui-même, pourrait en faire. La plupart des services dans les nuages ont des contraintes sur le dépôt d'images (que ce soit sur la taille ou le format) dans le but d'optimiser le stockage et l'utilisation de la bande passante. Si Alice crypto-compresse ses images avec un facteur de qualité QF , l'administrateur du réseau peut vouloir les recompresser. Cette recompression, se fait, usuellement, en appliquant une nouvelle compression JPEG avec une table de quantification spécifique. Néanmoins, la recompression simple d'une image crypto-compressée, bien que techniquement possible, ne permettra pas à Alice de déchiffrer ses images en utilisant sa clé secrète : le contenu de l'image originale serait alors perdu. Les méthodes de crypto-compression ne sont pas robustes à une recompression simple. Alice pourrait donc transmettre sa clé secrète en toute confiance à l'hébergeur pour lui permettre une crypto-compression avec un facteur de qualité plus faible, mais cette solution n'est pas sécurisée. Le but est alors de permettre une recompression des images JPEG crypto-compressées sans avoir accès à la clé secrète. Les images transférées par Alice peuvent donc être recompressées directement sur le serveur de *cloud*, avec un nouveau facteur de qualité QF* tel que QF* < QF.

7.4.2 Une approche de crypto-compression robuste à la recompression

La méthode de crypto-compression utilisée s'inspire des travaux réalisés par Puech *et al.* [99]. Nous proposons d'adapter cette approche de manière à la rendre robuste à l'application d'une ou plusieurs recompressions après une crypto-compression initiale. En effet, si une image crypto-compressée à l'aide de l'approche décrite dans [99] est recompressée sans adaptation, alors il se produit une désynchronisation avec la séquence binaire pseudo-aléatoire utilisée pour le déchiffrement lors de la phase de décodage. Par conséquent, le contenu de l'image en clair ne peut pas être reconstruit sans erreur. Pour résoudre ce problème, dans la méthode de crypto-compression proposée, nous réordonnons les coefficients JPEG pendant la phase de chiffrement. En figure 7.10, une description générale de la méthode de crypto-compression est présentée.

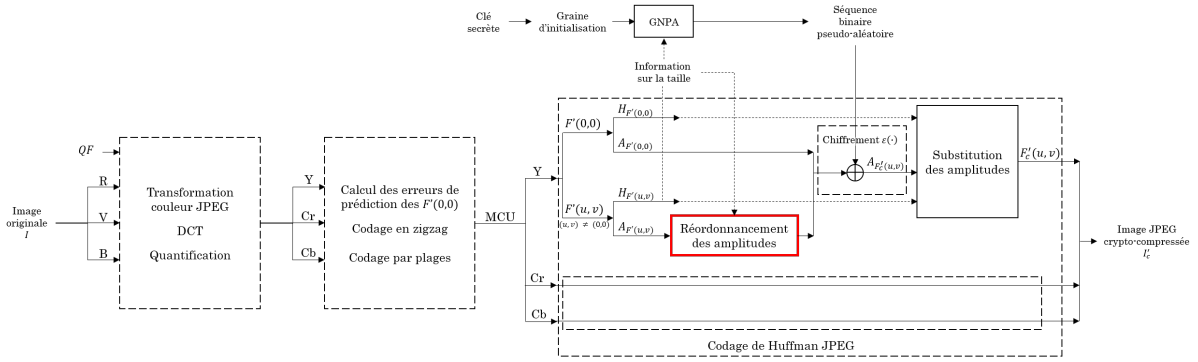


FIGURE 7.10 – Crypto-compression robuste à la recompression.

Les étapes d'une compression JPEG classique sont appliquées à l'image originale I jusqu'à la quantification des coefficients fréquentiels. Après cette dernière, le chiffrement est réalisé lors du codage de Huffman. Dans le but de préserver la confidentialité du contenu original de l'image, le chiffrement doit être appliqué au minimum sur la composante de luminance Y. Rappelons que les deux composantes de chrominances Cr et Cb peuvent être également chiffrées, mais comme illustré en figure 7.10, nous pouvons choisir de les laisser en clair car le SVH y est peu sensible. Dans chaque MCU, tous les coefficients $F'(u,v)$ non-nuls après quantification sont chiffrés. Les coefficients AC $F'(u,v)$ non-nuls sont alors réordonnés suivant leur taille, dans l'ordre décroissant. L'étape de recompression décrite en section 7.4.3 correspond à une division par deux de tous les coefficients AC $F'(u,v)$ non-nuls. Notons que le réordonnancement des coefficients rend possible la resynchronisation du GNPA, même dans le cas d'un coefficient $F'(u,v)$ avec une amplitude codée sur un bit – et donc ramené à zéro après recompression. Ainsi, sans l'étape de réordonnancement, il serait impossible de dissocier ces coefficients de ceux qui étaient déjà nuls avant la recompression : cela entraînerait une désynchronisation avec la séquence binaire pseudo-aléatoire.

Après sélection et réordonnancement des coefficients AC $F'(u,v)$ non-nuls, une clé secrète est utilisée pour générer une graine (*seed*) différente pour chaque MCU. Cette graine sert à initialiser un GNPA afin d'obtenir une séquence binaire pseudo-aléatoire. La taille de cette séquence est calculée en additionnant les tailles de chaque coefficient AC $F'(u,v)$ non-nul, de manière à obtenir un nombre de bits suffisant pour chiffrer

toutes leurs amplitudes. Ainsi, la valeur d'un coefficient chiffré $F'_c(u,v)$ est :

$$\begin{aligned} F'_c(u,v) &= \mathcal{E}\left(F'(u,v),\ \text{taille}(F'(u,v))\right), \\ &= \mathcal{E}\left(\left\{H_{F'(u,v)}, A_{F'(u,v)}\right\},\ \text{taille}(F'(u,v))\right), \\ &= \left\{H_{F'_c(u,v)}, A_{F'_c(u,v)}\right\}, \end{aligned} \tag{7.6}$$

où $H_{F'_c(u,v)} = H_{F'(u,v)}$.

Comme illustré en fig. 7.10, la fonction de chiffrement $\mathcal{E}(\cdot)$ consiste à effectuer un ou-exclusif entre la valeur de l'amplitude d'un coefficient en clair $F'(u,v)$ et la partie correspondante dans la séquence binaire pseudo-aléatoire, suivant sa taille et sa position dans le MCU. Les valeurs des amplitudes $A_{F'(u,v)}$ du flux binaire original sont alors substituées par leurs valeurs chiffrées $A_{F'_c(u,v)}$ pour obtenir le MCU chiffré. Dans cette séquence, l'équivalent chiffré du coefficient $F'(u,v)$, codé par une paire $(H_{F'(u,v)},\ A_{F'(u,v)})$, est $F'_c(u,v)$, codé par une paire $(H_{F'_c(u,v)},\ A_{F'_c(u,v)})$, dont l'entête est inchangé. Notons que le coefficient chiffré est codé sur le même nombre de bits que sa version en clair, puisque seul un ou-exclusif est appliqué. Finalement, l'image JPEG crypto-compressée I'_c est obtenue.

7.4.3 Comment recompresser une image crypto-compressée ?

Comme illustré en figure 7.11, la recompression est directement appliquée sur le flux binaire JPEG chiffré de chaque composante. Chaque MCU, après l'étape de crypto-compression, est composé d'un ou plusieurs coefficients, codés par des paires $(H_{F'_c(u,v)},\ A_{F'_c(u,v)})$. La première étape de la recompression consiste à supprimer le bit le moins significatif du paramètre d'amplitude de chaque coefficient $F'_c(u,v)$. Les coefficients recompressés $F'^*_c(u,v)$ sont ainsi calculés :

$$F'^*_c(u,v) = \begin{cases} \left\lfloor \frac{F'_c(u,v)}{2} \right\rfloor, & \text{si } |F'_c(u,v)| > 1, \\ 0, & \text{si } |F'_c(u,v)| = 1. \end{cases} \tag{7.7}$$

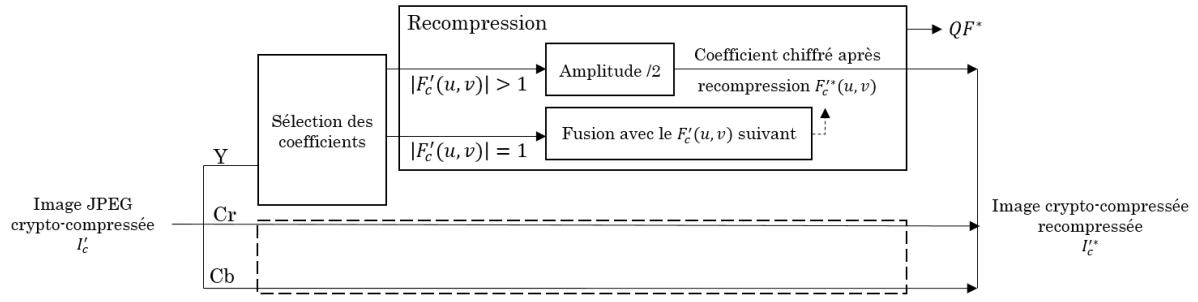


FIGURE 7.11 – Recompression de l'image crypto-compressée.

Supprimer le dernier bit de chaque coefficient non-nul implique une réduction d'un bit de la taille de leur code binaire. Il est donc nécessaire d'adapter en conséquence le paramètre renseignant leur taille dans l'entête. De plus, si l'amplitude d'un coefficient $F'_c(u,v)$ est codée sur un bit avant la recompression, sa version recompressée $F'^*_c(u,v)$ est un coefficient nul. Ainsi, ce coefficient doit être codé dans la plage de zéros du

prochain coefficient $F_c'^*(u,v)$, nécessairement non-nul par construction. La valeur de la plage de zéros $\mathrm{RL}_{H_{F_c'^*(u,v)}}$ est donc adaptée en tenant compte du nombre de coefficients précédents égaux à zéro. La nouvelle valeur de la plage de zéros est égale à la valeur initiale de la plage de zéros, plus celle du précédent coefficient $F_c'(u',v')$, plus un :

$$\mathrm{RL}_{H_{F_c'^*(u,v)}} = \mathrm{RL}_{H_{F_c'(u,v)}} + \mathrm{RL}_{H_{F_c'(u',v')}} + 1. \tag{7.8}$$

Ainsi, la valeur de l'entête après recompression est obtenue :

$$H_{F_c'^*(u,v)} = (\mathrm{RL}_{H_{F_c'^*(u,v)}}, \text{taille} - 1). \tag{7.9}$$

Finalement, dans chaque MCU, les coefficients chiffrés recompressés sont codés par la paire $(H_{F_c'^*(u,v)},\ A_{F_c'^*(u,v)})$. Les coefficients $q_{\mathrm{QF}^*}(u,v)$ de la table de quantification mise à jour Q_{QF^*} sont :

$$q_{\mathrm{QF}^*}(u,v) = \begin{cases} 2 \times q_{\mathrm{QF}}(u,v), & \text{si } 2 \times q_{\mathrm{QF}}(u,v) \leq 255, \\ 255, & \text{sinon.} \end{cases} \tag{7.10}$$

Nous notons que, dans le standard JPEG, les valeurs des coefficients des tables de quantification sont bornées, ce qui implique que si $q_{\mathrm{QF}^*}(u,v) > 255$, alors $q_{\mathrm{QF}^*}(u,v) = 255$, pour assurer la compatibilité avec le format JPEG. De ce fait, la qualité de l'image finale peut être altérée à cause de cette troncature. Comme la nouvelle table de quantification Q_{QF^*} est directement dérivée de la table de quantification Q_{QF} utilisée pour la première compression JPEG, la deuxième compression (*i.e.* après recompression) n'est pas obtenue suivant un facteur de qualité prédéfini de manière standard. Ainsi, il n'est pas possible de choisir une qualité désirée pour l'image recompressée déchiffrée. Le problème est donc d'estimer le facteur de qualité après recompression QF^* en utilisant Q_{QF^*}. Pour donner une approximation de ce facteur de qualité, nous proposons d'inverser l'équation (7.3), de calculer la valeur de chaque coefficient, et de moyenner les résultats obtenus. Nous avons deux équations possibles :

$$\mathrm{EQF}^*_{\leq 50} = \left\lfloor \frac{1}{64} \sum_{u=0}^{7} \sum_{v=0}^{7} \frac{q_{\mathrm{QREF}}(u,v) \times 5000}{q_{\mathrm{QF}^*}(u,v) \times 100 - 50} \right\rfloor, \tag{7.11}$$

$$\mathrm{EQF}^*_{>50} = \left\lfloor \frac{1}{64} \sum_{u=0}^{7} \sum_{v=0}^{7} 100 - \frac{q_{\mathrm{QF}^*}(u,v) \times 50 - 25}{q_{\mathrm{QREF}}(u,v)} \right\rfloor. \tag{7.12}$$

Le facteur de qualité estimé EQF^* est donné par :

$$\mathrm{EQF}^* = \begin{cases} \mathrm{EQF}^*_{\leq 50}, & \text{si } \mathrm{EQF}^*_{\leq 50} \leq 50, \\ \mathrm{EQF}^*_{>50}, & \text{sinon.} \end{cases} \tag{7.13}$$

Cette méthode d'inversion est correcte si l'on ne tient pas compte de l'équation (7.4). Dans le cas contraire, notons que les valeurs hors de l'intervalle $[1, 255]$ sont perdues. Les cas extrêmes sont définis par les deux tables de quantification Q_{QF^-} et Q_{QF^+}, où

tous les coefficients $q_{\text{QF}^-}(u,v)$ et $q_{\text{QF}^+}(u,v)$, $0 \leq u,v < 8$ sont respectivement égaux à 1 et 255. A l'aide de l'équation (7.13), nous obtenons $Q_{\text{QF}^-} = 11$ et $Q_{\text{QF}^+} = 99$ et donc, $\text{EQF}^* \in [11, 99]$, alors que $\text{QF} \in [1, 100]$.

Le principal avantage de la méthode proposée est qu'il est facile de localiser les bits supprimés après recompression. De cette façon, la synchronisation avec la séquence binaire pseudo-aléatoire est possible et la phase de déchiffrement se déroule sans erreur. En effet, puisque les coefficients non-nuls ont été réordonnés selon leurs amplitudes, ceux qui deviennent nuls après recompression se trouvent à la fin du MCU.

L'image crypto-compressée recompressée $I_c'^*$ est toujours au format JPEG car notre méthode préserve le format. Notons que le déchiffrement de cette image peut être réalisé à l'aide de la clé secrète utilisée lors du chiffrement, mais seulement si l'information que l'image a été recompressée est connue. Pour cela, nous pouvons par exemple ajouter un drapeau dans la partie commentaire du fichier JFIF, indiquant le nombre de recompressions qui ont été réalisées.

7.4.4 Déchiffrement et décodage de l'image JPEG crypto-compressée recompressée

Comme présenté en figure 7.12, la phase de déchiffrement et de décodage comprend quatre étapes principales : le déchiffrement du flux binaire JPEG chiffré pendant le décodage de Huffman, la quantification inverse, la transformation I-DCT et le changement inverse d'espace couleur (YCrCb vers RVB).

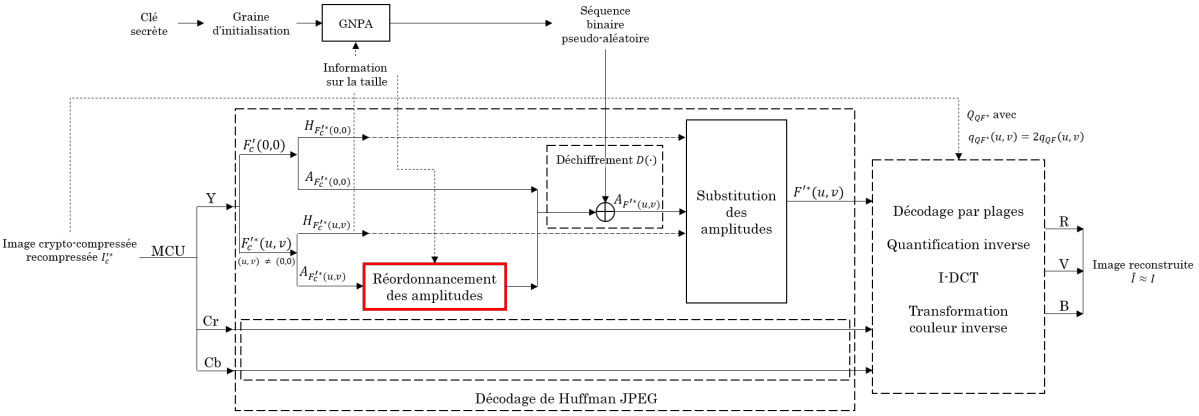


FIGURE 7.12 – Déchiffrement et décodage de l'image JPEG crypto-compressée recompressée.

L'opération de déchiffrement peut être réalisée par toute personne connaissant la clé secrète utilisée lors du chiffrement. En se référant à la partie commentaire du fichier JFIF, il est possible de savoir si l'image crypto-compressée a été recompressée. La première étape du déchiffrement consiste à utiliser la clé secrète comme graine d'initialisation du GNPA et ainsi générer la séquence binaire pseudo-aléatoire nécessaire au déchiffrement. Le déchiffrement de l'image prend alors en compte le fait de décaler la séquence binaire chiffrée pour déchiffrer chaque coefficient. En effet, les coefficients ont été divisés par deux pendant la phase de recompression. Le dernier bit de chaque code a été supprimé et, de ce fait, il ne faut pas prendre en compte le dernier bit des parties de la séquence

binaire pseudo-aléatoire associées à chaque coefficient. La fonction de déchiffrement $\mathcal{D}(\cdot)$ est similaire à la fonction de chiffrement. Elle prend en entrée deux paramètres : un coefficient chiffré recompressé $F_c'^*(u,v)$ et sa taille, connue grâce à son entête. Le coefficient en clair associé est obtenu en appliquant un ou-exclusif entre l'amplitude du coefficient chiffré et la partie associée dans la séquence binaire pseudo-aléatoire :

$$\begin{aligned} F'^*(u,v) &= \mathcal{D}\left(F_c'^*(u,v),\ \text{taille}(F_c'^*(u,v))\right), \\ &= \mathcal{D}\left(\left\{H_{F_c'^*(u,v)}, A_{F_c'^*(u,v)}\right\},\ \text{taille}(F'(u,v))-1\right), \\ &= \left\{H_{F'^*(u,v)}, A_{F'^*(u,v)}\right\}, \end{aligned} \tag{7.14}$$

où $H_{F'^*(u,v)} = H_{F_c'^*(u,v)}$.

Nous remarquons que si la méthode de recompression proposée est directement appliquée sur l'image JPEG en clair I' (*i.e* sans chiffrement), alors nous obtenons exactement les même coefficients $F'^*(u,v)$ qu'après avoir déchiffré les coefficients $F_c'^*(u,v)$. En effet, la méthode de chiffrement/déchiffrement est commutative avec la méthode de recompression, car l'opération ou-exclusif est commutative avec la fonction partie entière inférieure :

$$\begin{aligned} F'^*(u,v) &= \mathcal{D}\left(\left\lfloor \frac{\mathcal{E}(F'(u,v),\ \text{taille}(F'(u,v)))}{2} \right\rfloor,\ \text{taille}(F'(u,v))-1\right), \\ &= \left\lfloor \frac{\mathcal{D}(\mathcal{E}(F'(u,v),\ \text{taille}(F'(u,v))),\ \text{taille}(F'(u,v))-1)}{2} \right\rfloor, \\ &= \left\lfloor \frac{F'(u,v)}{2} \right\rfloor. \end{aligned} \tag{7.15}$$

Après l'étape de déchiffrement, le décodage de Huffman est appliqué et les coefficients DCT quantifiés sont reconstruits. L'opération de quantification inverse est alors réalisée en vue d'obtenir les valeurs déquantifiées $\hat{\hat{F}}(u,v)$. Comme présenté précédemment, l'image déchiffrée correspond à l'image compressée recompressée I'^*. Sa table de quantification Q_{QF^*} est dérivée de la table de quantification Q_{QF} de l'image compressée I', dont les coefficients ont été multipliés par deux :

$$\begin{aligned} \hat{\hat{F}}(u,v) &= F'^*(u,v) \times q_{\text{QF}^*}(u,v) \\ &= \begin{cases} F'^*(u,v) \times 2 \times q_{\text{QF}}(u,v), & \text{si } 2 \times q_{\text{QF}}(u,v) \leq 255, \\ 255, & \text{sinon.} \end{cases} \end{aligned} \tag{7.16}$$

Par ailleurs, chaque coefficient déquantifié $\hat{\hat{F}}(u,v)$ de l'image recompressée peut être exprimé en fonction du coefficient déquantifié $\hat{F}(u,v)$:

$$\begin{aligned} \hat{\hat{F}}(u,v) &= F'^*(u,v) \times q_{\text{QF}^*}(u,v) \\ &= \lfloor \frac{F'(u,v)}{2} \rfloor \times 2 \times q_{\text{QF}}(u,v) \\ &= \begin{cases} \hat{F}(u,v), & \text{si } F'(u,v) \text{ est pair,} \\ \hat{F}(u,v) - q_{\text{QF}}(u,v), & \text{si } F'(u,v) \text{ est impair.} \end{cases} \end{aligned} \tag{7.17}$$

Notons que si un coefficient quantifié est pair avant la recompression, l'opération de quantification inverse appliquée au coefficient quantifié permet de retrouver sa valeur initiale.

Finalement, l'image décompressée RVB $\hat{\hat{I}}$ est obtenue en appliquant la transformation I-DCT pour convertir les coefficients fréquentiels en pixels, puis le changement d'espace couleur inverse, pour convertir l'image YCrCb en RVB. Comme après une compression JPEG classique, en fonction du facteur de qualité QF, le contenu de l'image reconstruite $\hat{\hat{I}}$ est plus ou moins fidèle à celui de l'image originale I.

7.5 Résultats expérimentaux

Dans cette section, nous présentons les résultats obtenus en appliquant notre méthode de recompression des images JPEG crypto-compressées. La section 7.5.1 détaille un exemple complet d'application de notre méthode. Ensuite, dans la section 7.5.2, nous effectuons une analyse afin d'estimer le facteur de qualité de l'image obtenue après recompression. Enfin, dans la section 7.5.3, nous examinons le niveau de sécurité et les propriétés statistiques des images JPEG crypto-compressées afin d'estimer la sécurité visuelle de la méthode de crypto-compression. En outre, nous discutons des paramètres à utiliser en fonction du niveau de sécurité requis.

7.5.1 Exemple complet de la méthode proposée

La fig. 7.13 a été obtenue en appliquant la méthode de recompression d'images JPEG crypto-compressées à l'image *Peppers* (321×481 pixels) en utilisant un facteur de qualité $\text{QF} = 75\%$ pour la première compression JPEG. La première étape de la méthode consiste à crypto-compresser l'image originale. Dans cet exemple d'application, les coefficients AC et DC des trois composantes (Y, Cr et Cb) sont chiffrés pour préserver la confidentialité visuelle du contenu original de l'image. En effet, nous pouvons remarquer qu'il est difficile de distinguer les détails et que la valeur du PSNR couleur est très basse ($11,74\ dB$). Après la phase de décodage, nous pouvons remarquer que l'image JPEG crypto-compressée déchiffrée est très similaire à l'image originale (PSNR = $38,59\ dB$). Notons que cette image est identique à celle obtenue après une compression JPEG classique dans le domaine clair avec $\text{QF} = 75\%$. Nous recompressons alors l'image crypto-compressée obtenue directement dans le domaine chiffré (*i.e.* sans déchiffrer l'image crypto-compressée). En analysant la table de quantification, nous pouvons estimer le facteur de qualité après recompression $\text{EQF}^* = 50\%$. Enfin, l'image JPEG crypto-compressée recompressée peut être parfaitement déchiffrée à l'aide de la clé secrète utilisée pour la crypto-compression. La valeur du PSNR est haute ($35,21\ dB$), ce qui indique une forte similarité entre l'image obtenue et l'image originale *Peppers*.

7.5.2 Analyse du facteur de qualité

Dans cette section, nous nous intéressons à l'estimation du facteur de qualité après recompression EQF^*. Cette estimation est réalisée d'après la table de quantification de la luminance car elle est plus pertinente qu'en utilisant la table de quantification de la chrominance. Dans l'équation (7.13), nous avons montré qu'il était possible d'obtenir la valeur de EQF^* en inversant l'équation (7.3). Du fait de l'intervalle des valeurs possibles, nous avons remarqué que $\text{EQF}^* \in [11, 99]$. Dans le tableau 7.1, nous présentons des QF

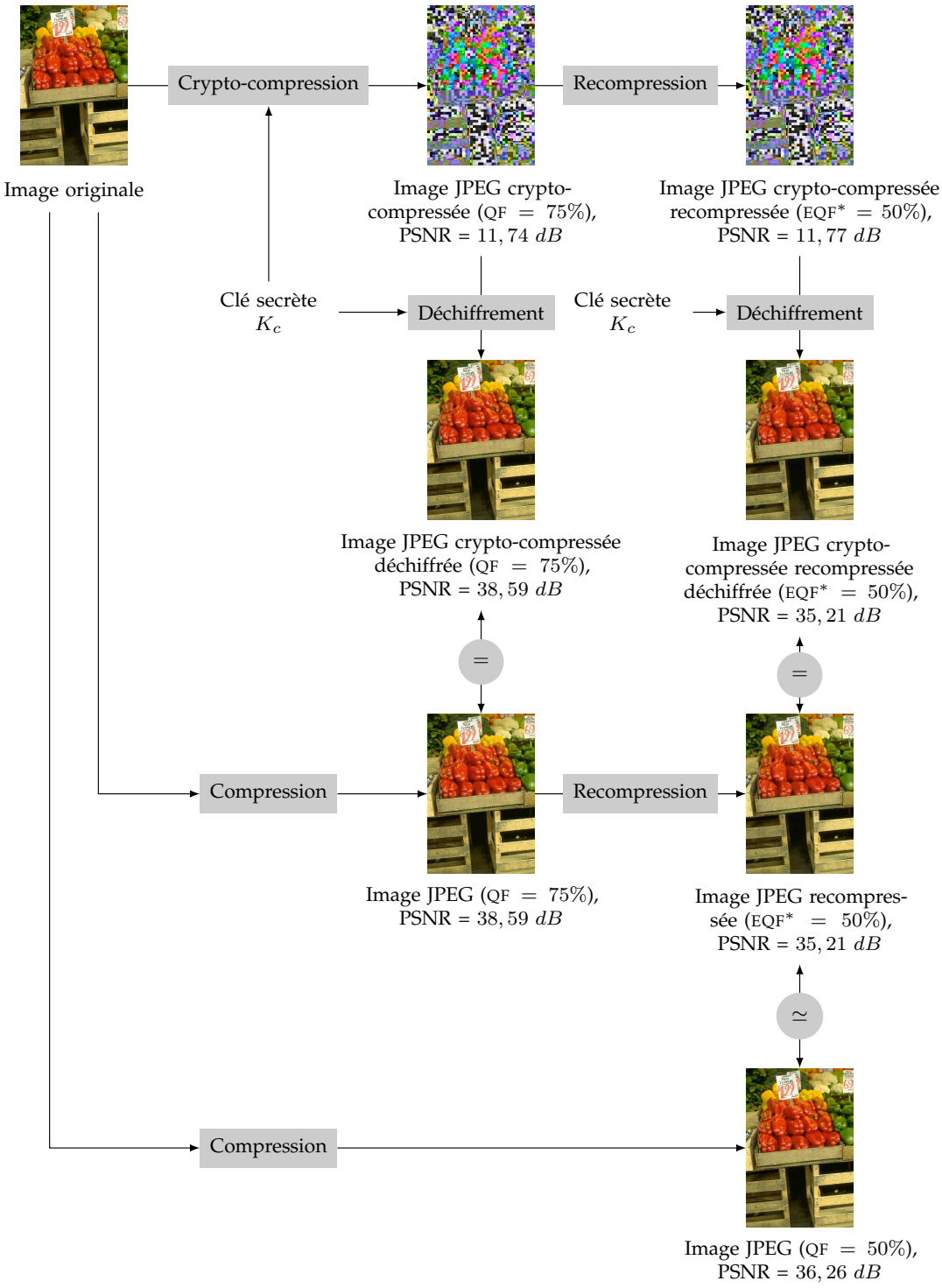


FIGURE 7.13 – Illustration de la méthode : crypto-compression de l'image *Peppers* (QF $= 75\%$, chiffrement des coefficients AC et DC de la luminance et des deux composantes de la chrominance) et recompression de l'image crypto-compressée.

pouvant être utilisés pour la première crypto-compression et les valeurs correspondantes de EQF*. Notons que les valeurs choisies sont représentatives de l'intervalle des valeurs possibles. Nous pouvons voir que, pour QF $\geq 90\%$, EQF* reste élevé ($\simeq -10$ %). Pour

des valeurs faibles de QF ($\text{QF} \leq 25\%$), les valeurs de EQF* sont proches de QF ($\simeq -10\ \%$). Pour des QF très utilisés (par exemple, $\text{QF} = 75\%$ et $\text{QF} = 50\%$), EQF* est bien plus faible qu'avant recompression (de $-25\ \%$ à $-50\ \%$).

QF (%)	100	95	90	75	50	25	15
EQF* (%)	97	90	80	50	25	14	12

TABLE 7.1 – Exemples de QF et leurs valeurs EQF* associées après recompression avec la méthode proposée.

Dans la fig. 7.14, nous illustrons la différence entre la table de quantification Q_{QF^*} obtenue avec notre méthode de recompression et associée au facteur de qualité estimé EQF*, et la table de quantification Q_{QF} telle que $\text{QF} = \text{EQF}^*$. En fig. 7.14.a, nous présentons Q_{75}. Cette table de quantification, générée à partir de Q_{50} (fig. 7.2) selon l'équation 7.3, est utilisée pendant la crypto-compression de l'image originale. En utilisant la méthode proposée de recompression, Q_{QF^*} est calculée en multipliant par 2 chaque coefficient de Q_{QF} (équation (7.10)). La table obtenue est présentée en fig. 7.14.b. De plus, en utilisant l'équation (7.13), EQF* est égal à 50. En comparant la fig. 7.2 et la fig. 7.14.b, nous pouvons voir que les deux tables sont très similaires. En effet, la différence entre deux coefficients à la même position est nulle ou égale à 1.

	0	1	2	3	4	5	6	7
0	8	6	5	8	12	20	26	31
1	6	6	7	10	13	29	30	28
2	7	7	8	12	20	29	35	28
3	7	9	11	15	26	44	40	31
4	9	11	19	28	34	55	52	39
5	12	18	28	32	41	52	57	46
6	25	32	39	44	52	61	60	51
7	36	46	48	49	56	50	52	50

(b)

	0	1	2	3	4	5	6	7
0	16	12	10	16	24	40	52	62
1	12	12	14	20	26	58	60	56
2	14	14	16	24	40	58	70	56
3	14	18	22	30	52	88	80	62
4	18	22	38	56	68	110	104	78
5	24	36	56	64	82	104	114	92
6	50	64	78	88	104	122	120	102
7	72	92	96	98	112	100	104	100

(c)

FIGURE 7.14 – Différence entre Q_{50} et Q_{QF^*}, avec $\text{EQF}^* = 50\%$: a) Table de quantification Q_{75}, pour $\text{QF} = 75\%$, calculée d'après Q_{50} (fig. 7.2), b) Table de quantification Q_{QF^*} avec $\text{EQF}^* = 50\%$, calculée d'après Q_{75}.

En fig. 7.15, nous évaluons la distance $L2$ entre Q_{QF^*} et Q_{QF}, telle que $\text{QF} = \text{EQF}^*$ pour différentes valeurs dans l'intervalle $[11, 99]$. En d'autres termes, nous évaluons la pertinence de notre estimation (EQF*) pour une valeur réelle de QF*. Notons qu'une divergence significative commence à partir d'un EQF* approximativement égal à 34% et jusqu'à 11%. Nous remarquons également un intervalle où les valeurs de la fonction sont en dent de scie à cause d'erreurs d'arrondi des nombres entiers. Néanmoins, nous pouvons donner une estimation pertinente du facteur de qualité après recompression de 99% à 34% avec une distance L2 inférieure à 10. Notons que pour la table de quantification de la chrominance, dont les coefficients sont plus grands, la divergence entre les valeurs réelles et estimées est plus importante.

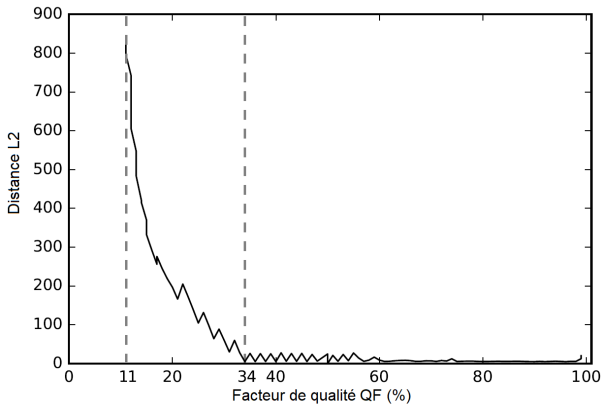


FIGURE 7.15 – Distance $L2$ entre Q_{QF^*} (associée au facteur de qualité EQF^*) et Q_{QF} calculée à partir de la table standard Q_{50} telle que $\mathrm{QF} = \mathrm{EQF}^*$.

7.5.3 Discussion sur le niveau de sécurité visuelle

Dans cette section, nous proposons une discussion sur le niveau de sécurité visuelle fourni par l'approche de crypto-compression utilisée au sein de la méthode proposée de recompression d'images JPEG crypto-compressées. Comme présenté dans la section 7.4 et dans la fig. 7.16, des paramètres différents peuvent être utilisés pendant la crypto-compression de l'image originale : chiffrement des coefficients AC ou chiffrement des coefficients AC et DC, de la composante de luminance (Y) seulement ou des trois composantes (Y, Cr, Cb). Ces paramètres sont choisis en fonction du niveau de sécurité visuelle désiré : chiffrement transparent, chiffrement suffisant ou confidentialité visuelle (chapitre 2) [34].

En chiffrant seulement les coefficients AC non nuls de la composante de luminance (ou ceux de la luminance et des deux composantes de chrominance), comme illustré dans la fig. 7.16, nous pouvons observer que seule la haute qualité de l'image originale est préservée : dans ce cas, le chiffrement est transparent. La méthode suit les conditions établies par Van Droogenbroeck [153] pour le chiffrement sélectif utilisé dans les applications en temps réel : acceptation visuelle (une partie de l'information peut être visible, mais l'image chiffrée semble bruitée), préservation de la taille et du format. De plus, un chiffrement suffisant peut être obtenu en chiffrant les coefficients AC et DC de la composante de luminance. Comme illustré en fig. 7.16, dans ce cas, le contenu original est très dégradé mais les couleurs de l'image originale sont préservées. Cependant, pour un niveau élevé de sécurité visuelle, il est nécessaire de dissimuler le contenu de l'image (niveau confidentiel). Ainsi, les coefficient AC et DC de la luminance et des deux composantes de chrominance doivent être chiffrés. Le chiffrement avec ces paramètres permet un bon niveau de sécurité car seule une quantité limitée d'information sur le contenu original de l'image peut être extraite de l'image crypto-compressée.

Au regard des propriétés statistiques de l'image chiffrée, nous pouvons voir que même si le PSNR avec l'image originale est égal à $11,69\ dB$, les valeurs de l'UACI et du NPCR ne sont pas significatives ($17,61\%$ et $98,14\%$ respectivement). De plus, la valeur de l'entropie est plus élevée que dans l'image originale ($7,61\ bpp > 7,12\ bpp$) mais n'est pas proche de la valeur maximale de $8\ bpp$. Avec le test du χ^2, nous observons aussi que la valeur reste haute après le chiffrement (racine carrée égale à $165,95$). Notons

qu'il n'existe pas, à ce jour, d'outils suffisamment adaptés à l'évaluation du niveau de sécurité des données crypto-compressées. Dans le but d'améliorer le niveau de sécurité en introduisant de la diffusion, il serait possible d'ajouter une opération de mélange à notre méthode de crypto-compression – comme avec FIBS [73] ou la méthode de Lian *et al.* [76] par exemple. Dans ce cas, la méthode de chiffrement serait robuste aux attaques à texte clair choisi. Cependant, la taille de l'image chiffrée pourrait être légèrement plus importante que celle de l'image originale.

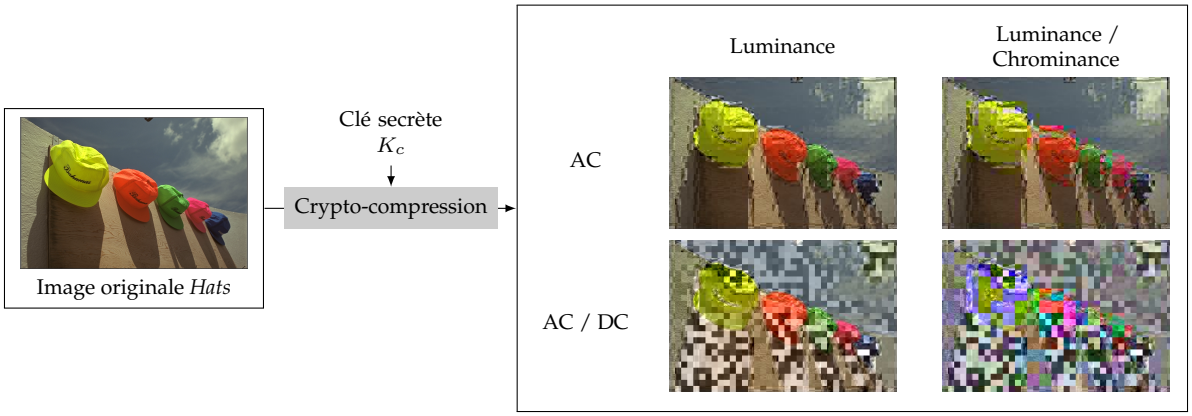
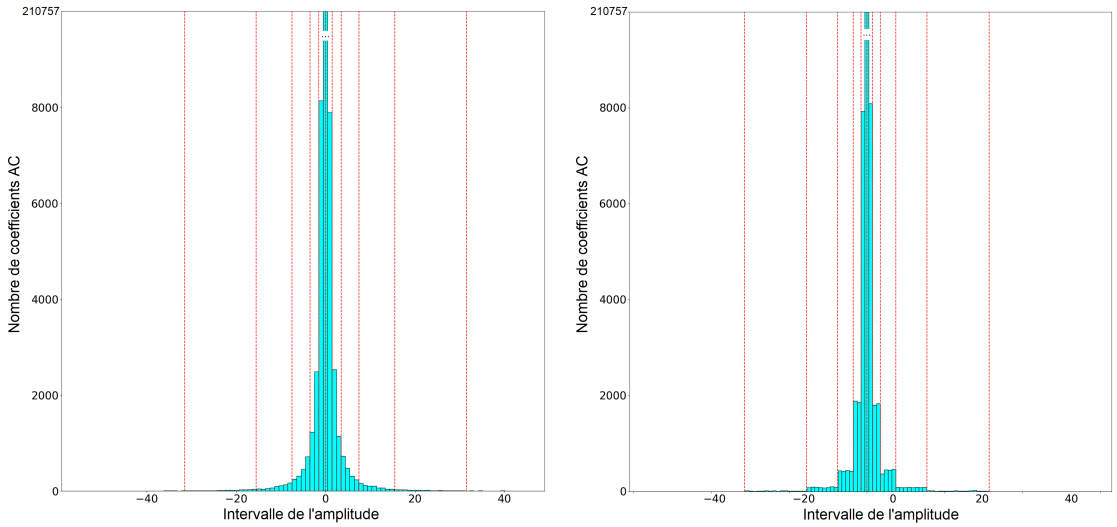


FIGURE 7.16 – Crypto-compression de l'image *Hats* avec QF $= 80\%$ et avec les différents paramètres possibles de notre méthode.

En fig. 7.17, nous avons appliqué le chiffrement aux coefficients AC et DC de la luminance et des deux composantes de chrominance en utilisant un facteur de qualité QF $= 90\%$. Nous présentons la distribution des coefficients AC (après quantification) avant et après le chiffrement. Tout d'abord, notons que dans les deux cas, la distribution semble être Laplacienne. En effet, beaucoup de coefficients sont égaux à zéro après la compression JPEG. Il est aussi important de remarquer que les coefficients chiffrés ont exactement la même taille (en termes de bits) que ceux en clair. En effet, la taille de chaque coefficient à chiffrer est considérée comme un paramètre pour sélectionner le nombre de bits nécessaires au chiffrement dans la séquence aléatoire. La méthode de chiffrement permet ainsi de préserver la structure JPEG et la taille de l'image obtenue après une compression JPEG classique. D'un point de vue sécurité, cela constitue une faiblesse car les coefficients les plus représentés sont codés sur un plus petit nombre de bits (codage de Huffman). Par exemple, les coefficients égaux à 1 ou -1 sont seulement codés sur un bit et, de ce fait, seules deux valeurs sont possibles pour les chiffrer. Par ailleurs, si nous considérons les coefficients codés avec le même nombre de bits, nous pouvons voir que le chiffrement permet d'assurer l'uniformité de leur distribution (fig. 7.17.b). Il n'est alors pas possible d'exploiter leurs propriétés statistiques pour reconstruire les valeurs des coefficients en clair.

7.6 Conclusion

Dans ce chapitre, nous avons proposé une nouvelle méthode de recompression d'images JPEG crypto-compressées, pouvant être appliquée dans le domaine chiffré. A



(a) Avant le chiffrement. (b) Après le chiffrement.

FIGURE 7.17 – Distribution des coefficients AC avant et après le chiffrement de l'image (QF $= 90\%$, chiffrement des coefficients AC et DC des trois composantes Y, Cr et Cb).

notre connaissance, cette méthode est l'une des premières à permettre une recompression directement dans le domaine chiffré, sans connaître la clé de chiffrement utilisée. L'étape de recompression consiste à diviser par deux chaque coefficient DCT quantifié chiffré. De ce fait, le LSB des coefficients quantifiés chiffrés non nuls est supprimé. Les coefficients devenus nuls après cette étape sont alors codés dans la plage de zéros du coefficient non-nul suivant. Pour le décodage, les coefficients de la table de quantification sont adaptés en conséquence, en les multipliant par deux. Comme présenté dans la section 7.5, cette opération de recompression permet d'obtenir un très bon compromis entre le taux de compression et la qualité de l'image. De plus, contrairement à une compression classique avec JPEG, l'image recompressée avec EQF* est très similaire à l'image JPEG obtenue après l'application directe d'une compression JPEG avec le facteur de qualité QF. Il n'existe pas d'artefacts, tels que du grain ou une perte importante de netteté. Pour rendre possible cette opération de recompression dans le domaine chiffré, une nouvelle étape de crypto-compression a dû être développée. En effet, les coefficients DCT quantifiés sont chiffrés en tenant compte de leur taille, dans l'ordre décroissant, pour éviter une désynchronisation pendant la phase de décodage. En outre, dans l'image crypto-compressée, le contenu de l'image originale est tenu secret, comme indiqué par un PSNR proche de $10\ dB$. Notons que, après la recompression, la confidentialité visuelle est préservée car notre méthode de recompression n'introduit pas de faille de sécurité. Ainsi, en plus d'offrir un très haut niveau de sécurité et d'être robuste à une recompression, la méthode de chiffrement utilisée préserve le format, sans augmenter la taille des données originales.

Dans des travaux futurs, nous souhaitons étudier la robustesse d'autres techniques de crypto-compression à notre méthode de recompression dans le domaine chiffré. De plus, pour rendre la méthode proposée sécurisée face aux attaques par clair choisi, il est en fait possible de la combiner avec une méthode par mélange comme, par exemple,

la méthode FIBS. De plus, il serait intéressant d'estimer plus précisément le facteur de qualité EQF*.

Ces travaux ont fait l'objet d'une publication dans la revue internationale IEEE Transactions of Circuits and Systems for Video Technology en 2020 [58].

Conclusion

Dans ce chapitre, nous présentons une synthèse des travaux effectués lors de cette thèse, ainsi que nos pistes de recherche futures. Tout d'abord, nous résumons le contenu du manuscrit et nous dressons un bilan des contributions réalisées. Enfin, nous proposons des perspectives envisageables en prolongement direct de cette thèse.

Conclusion

Dans cette thèse, nous avons proposé de nouvelles méthodes d'analyse et de traitement des images dans le domaine chiffré. Dans l'état-de-l'art, nous avons tout d'abord décrit la structure et les propriétés des images numériques. Nous avons ensuite présenté les différentes méthodes d'insertion de données cachées (IDC), en développant leurs classes, leurs propriétés et les métriques utilisées pour leur évaluation. Nous nous sommes ensuite intéressés aux principes de la cryptographie ainsi qu'à leur application au chiffrement d'images. Ainsi, nous avons décrit les méthodes de l'état-de-l'art et les outils pouvant être utilisées pour mesurer le niveau de sécurité visuelle des images chiffrées. Enfin, nous avons développé la problématique de l'analyse et du traitement des images dans le domaine chiffré, en portant un intérêt particulier aux méthodes d'insertion de données cachées dans les images chiffrées (IDCDC). De cette façon, nous avons réalisé un état-de-l'art des méthodes et principes de référence à partir desquels nos travaux ont été développés et évalués.

Une grande partie des travaux réalisés concerne l'IDCDC. Au début de cette thèse, en 2017, il n'existait à notre connaissance aucune méthode d'IDCDC permettant d'obtenir un compromis intéressant entre la charge utile et la qualité de l'image reconstruite. En effet, une valeur élevée de la charge utile entraînait nécessairement une dégradation importante de l'image reconstruite. Notons également que la plupart des méthodes de l'état-de-l'art étaient basées sur la substitution des bits de poids faible (LSB) et exploitaient peu la redondance entre les pixels dans le domaine clair pour réaliser l'insertion du message secret. Dans les travaux de recherche réalisés lors de cette thèse, nous avons pris à contre-pied toutes les approches de l'état-de-l'art en proposant trois méthodes d'IDCDC basées sur la prédiction des valeurs des bits de poids fort (MSB).

Dans notre première contribution, nous avons développé une méthode d'IDCDC réversible déclinée en deux approches : l'approche IDCHC-CEP (insertion de données cachées haute capacité avec correction des erreurs de prédiction) et l'approche IDCHC-SEP (insertion de données cachées haute capacité avec signalisation des erreurs de prédiction). Dans les deux approches, tous les pixels de l'image en clair qui ne peuvent être prédits en fonction de leurs voisins sont identifiés. Dans l'approche IDCHC-CEP,

l'image originale est pré-traitée pour éliminer toutes les erreurs de prédiction. L'image pré-traitée est ensuite chiffrée et, pour réaliser l'IDC, le MSB de chaque pixel de l'image chiffrée est directement substitué par un bit du message secret. Dans ce cas, la charge utile est égale à $1\ bpp$ et l'image reconstruite correspond à l'image pré-traitée qui est ainsi très similaire à l'image originale (PSNR $> 50\ dB$). Dans l'approche IDCHC-SEP, l'image originale est chiffrée sans aucune modification. Lors de la phase d'IDC, les données cachées insérées comprennent le message secret mais aussi des informations nécessaires à la signalisation des erreurs de prédiction. Dans ce cas, la charge utile est légèrement inférieure à $1\ bpp$ mais la réversibilité est parfaite.

Dans notre deuxième contribution, nous avons montré que tous les plans d'une image chiffrée pouvaient être utilisés pour réaliser l'IDC d'un message secret. Nous avons commencé par présenter une extension de l'approche IDCHC-SEP. En commençant par le plan MSB, chaque plan binaire est analysé itérativement pour mettre en évidence les erreurs de prédiction puis chiffré et marqué par les bits d'un message secret. Après le décodage, l'image originale est parfaitement reconstruite en utilisant les informations sur l'emplacement des erreurs et la prédiction. Les résultats expérimentaux montrent que la charge utile obtenue est de $1,836\ bpp$ en moyenne, ce qui atteste d'une forte augmentation par rapport à l'approche basée sur la seule utilisation du plan MSB. Par la suite, nous avons également proposé une nouvelle méthode d'IDCDC réversible permettant d'obtenir une charge utile encore plus élevée. Contrairement aux approches IDCHC-CEP ou IDCHC-SEP, les erreurs de prédiction ne sont ni corrigées, ni signalées à l'aide de drapeaux. En effet, une adaptation réversible des plans binaires est réalisée pour rendre possible la détection et la correction des erreurs de prédiction lors de la phase de décodage. Selon nos expérimentations, une très haute capacité est obtenue, avec une charge utile égale à $2,459\ bpp$ en moyenne. Par ailleurs, la suppression des drapeaux permet d'augmenter le niveau de sécurité de l'approche proposée.

Notre troisième contribution concerne l'analyse et la correction d'images chiffrées bruitées. Tout d'abord, nous avons présenté une étude de l'utilisation de l'entropie de Shannon dans des blocs de pixels de petite taille. Ensuite, nous avons décrit un nouveau mode de chiffrement basé sur la combinaison du mode CFB et du mode ECB, appelé CFB-puis-ECB. Nous avons alors développé un nouvel algorithme de correction d'images chiffrées bruitées exploitant les caractéristiques de ce mode de chiffrement et comprenant deux étapes principales : l'identification et la localisation des blocs de pixels altérés par le bruit (phase d'initialisation), puis l'analyse et la correction (phase de correction). Dans la phase d'initialisation, à partir de l'image directement chiffrée sans correction, il est possible de discriminer les blocs de pixels en clair de ceux qui ont été mal déchiffrés. Lors de la phase de correction, les configurations associées à chaque bloc de pixels mal déchiffré sont analysées. A l'aide d'un classifieur, la configuration associée au bloc de pixels de l'image originale est identifiée. Ainsi, la majorité des blocs de pixels sont correctement déchiffrés et l'image reconstruite est très similaire à l'image originale, selon les résultats obtenus en termes de PSNR. En outre, l'approche proposée préserve le format de l'image originale et n'augmente pas sa taille, contrairement aux méthodes classiques utilisant des codes correcteurs d'erreur.

Enfin, dans notre quatrième contribution, nous nous sommes intéressés à la recompression d'images JPEG crypto-compressées. Nous avons tout d'abord proposé une méthode de crypto-compression robuste à la recompression grâce à une réorganisation

des coefficients JPEG pendant la phase de chiffrement. La recompression est réalisée directement dans le domaine chiffré, sur le flux binaire JPEG, en supprimant le dernier bit du code des coefficients DCT non nuls et en adaptant leurs codes de Huffman. Cette méthode est efficace pour recompresser une image JPEG crypto-compressée en termes de taux de compression. De plus, comme l'opération de chiffrement est totalement réversible, le déchiffrement de l'image recompressée permet d'obtenir une image ayant une qualité visuelle similaire à celle de l'image compressée originale.

En conclusion, dans ces travaux de recherche, nous avons proposé différentes méthodes d'analyse et de traitement des images dans le domaine chiffré. Nous avons ainsi développé trois méthodes d'IDCDC, une approche de correction des images chiffrées bruitées et un algorithme de recompression d'images JPEG crypto-compressées. Par ailleurs, ces différents travaux peuvent être améliorés ou étendus à d'autres applications. Ainsi, nous décrivons des perspectives pouvant être envisagées dans la section suivante.

Perspectives

Dans cette section, nous présentons des perspectives sur l'insertion de données cachées dans les images chiffrées, l'analyse et le traitement des images JPEG crypto-compressées et la correction d'images chiffrées bruitées qui, selon nous, pourraient être intéressantes à approfondir.

Chiffrement d'images homomorphe à l'IDC

Comme expliqué dans le chapitre 2, une image peut être chiffrée en utilisant un cryptosystème homomorphe, tel que la méthode de Paillier [91]. Dans ce cas, l'IDC d'un message secret est effectuée dans le domaine chiffré sans atteinte à la confidentialité du contenu de l'image en clair (chapitre 3). De plus, après le déchiffrement de l'image chiffrée marquée, une image en clair, marquée par le message secret et similaire à l'image originale est obtenue. Notons néanmoins qu'aucune des méthodes d'IDCDC basées sur l'utilisation du chiffrement à clé publique ne permet une haute capacité sans augmenter de manière significative la taille des données dans le domaine chiffré. En effet, le taux d'expansion de ces méthodes dépend de la taille de la clé. Par exemple, pour des clés de 512 bits, il se situe entre 128 et 256.

Dans les chapitres 4 et 5, nous avons décrit trois méthodes efficaces d'IDCDC permettant d'atteindre une haute capacité, tout en préservant le format et la taille de l'image originale. Cependant, dans les méthodes proposées, l'IDC n'est pas réalisée de manière homomorphe. Ainsi, le message secret inséré ne peut pas être conservé dans le domaine clair, après le déchiffrement de l'image chiffrée marquée.

Nous proposons alors une méthode haute capacité d'IDCDC basée sur l'utilisation du cryptosystème de Paillier, dont le fonctionnement général est illustré en fig. 8.1. Lors de la phase de codage, l'image originale est pré-traitée et chiffrée par blocs de pixels. Les propriétés homomorphiques du cryptosystème de Paillier sont ensuite exploitées pour multiplier l'image chiffrée par le message chiffré, ce qui correspond à substituer toutes les valeurs des LSB de chaque pixel dans le domaine clair par des bits d'un message secret. Le destinataire de l'image chiffrée marquée peut alors reconstruire une image

marquée en clair très similaire à l'image originale. Il peut également extraire sans perte le message secret en clair.

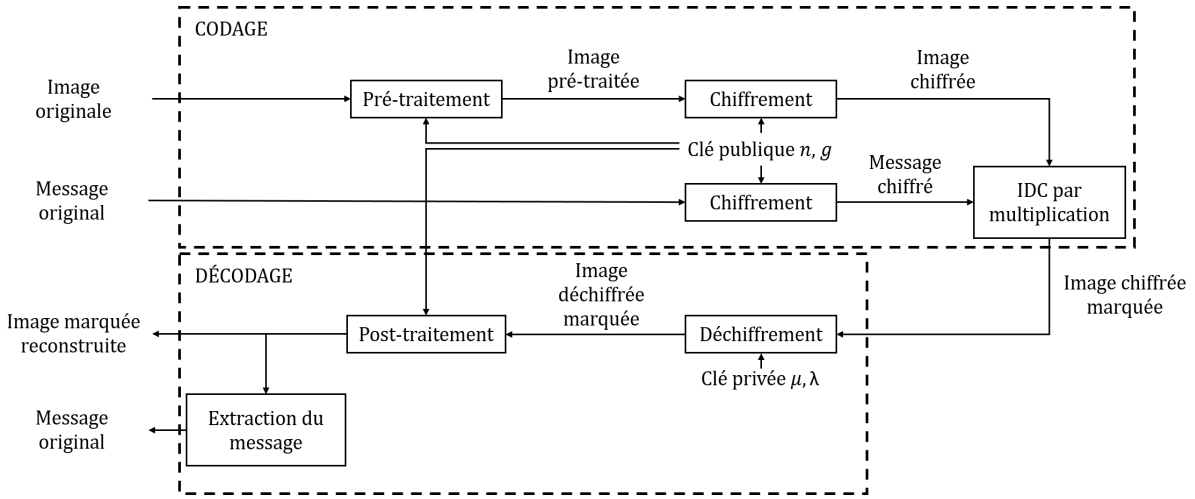


FIGURE 8.1 – Schéma général de la méthode d'IDC haute capacité basée sur l'utilisation du cryptosystème de Paillier.

Selon nos premiers résultats, nous obtenons un compromis très intéressant entre le taux d'expansion, la charge utile et la qualité de l'image reconstruite. En effet, le taux d'expansion des données est égal à 2 avec notre méthode, ce qui est nettement inférieur aux résultats des méthodes de l'état-de-l'art. De plus, quelle que soit l'image utilisée, la valeur du PSNR entre l'image reconstruite marquée en clair et l'image originale correspondante est supérieure à $50\ dB$ et la valeur du SSIM est très proche de 1. En outre, la charge utile obtenue est très élevée car chaque pixel de l'image contient un bit de message secret dans son LSB (charge utile $=\ 1\ bpp$).

Des résultats préliminaires sont présentés dans la revue internationale IEEE Access [114]. Dans de futurs travaux, nous allons investiguer un moyen d'extraire le message secret inséré dans le domaine chiffré. Nous sommes également intéressés par étendre la méthode proposée à l'IDC dans des maillages 3D chiffrés sans augmentation de la taille.

IDC dans les images JPEG crypto-compressées

Le premier article traitant d'IDC dans les images crypto-compressées date de 2014 et a été proposé par Qian *et al.* [119]. Les auteurs suggèrent de conserver la structure JPEG, en modifiant directement le flux binaire. Les bits où l'insertion du message peut être réalisée sont alors identifiés et des codes correcteurs d'erreur sont ensuite utilisés pour coder le message secret. Lors de la phase de reconstruction de l'image et d'extraction du message, si le receveur connaît seulement la clé de chiffrement, il peut reconstruire l'image en clair avec une bonne qualité et sans extraire le message secret inséré (*i.e.* il peut obtenir l'image JPEG en clair marquée). S'il connaît aussi la clé d'insertion, il peut reconstruire parfaitement l'image JPEG en clair en observant les artefacts visuels dûs à l'insertion des bits du message et peut extraire sans erreur le message inséré. Pour chiffrer l'image, les codes des valeurs des amplitudes des coefficients AC non nuls

sont concaténés puis un ou-exclusif est effectué avec une séquence pseudo-aléatoire. Les tables de quantification sont aussi chiffrées. Pour insérer le message secret, les blocs dont les coordonnées sont paires et qui ont au moins un coefficient AC non nul sont sélectionnés. Un bit du message secret, codé à l'aide de codes correcteurs d'erreur, est inséré dans chaque bloc en effectuant un XOR avec chaque valeur des amplitudes chiffrées. Ainsi, le taux d'insertion est très faible. Pour extraire le message et reconstruire l'image originale sans erreur, une fonction d'évaluation de l'homogénéité est utilisée. En utilisant les blocs voisins non marqués, elle permet de détecter les artefacts correspondant à l'insertion d'un bit du message. D'autres articles proposent de combiner une méthode de mélange pour chiffrer l'image à une méthode d'insertion de données cachées [90]. Enfin, des articles plus récents proposent une nouvelle méthode de chiffrement d'image JPEG, permettant d'obtenir une image de taille plus petite et plus sécurisée [120, 117]. Ces articles sont basés sur l'utilisation du format JPEG XT [123]. Dans ces méthodes, un nouveau flux binaire JPEG est construit en sélectionnant certains blocs dans l'image originale. Le flux binaire correspondant aux blocs restants est chiffré et inséré dans l'entête JPEG. Un algorithme de compression est alors utilisé pour réduire la taille de ce flux binaire. Ainsi, la place libérée permet d'insérer le message secret.

En nous inspirant de ces travaux ainsi que des méthodes développées dans les chapitres 4, 5 et 7, nous proposons une méthode d'insertion de données cachées hiérarchique dans les images JPEG crypto-compressées. La phase d'encodage est composée de deux étapes (fig. 8.2) : la crypto-compression de l'image originale, puis l'IDC hiérarchique. En effet, après la crypto-compression d'une image originale, l'image crypto-compressée est téléchargée sur une plateforme de *cloud* par son propriétaire. Une personne tierce peut alors effectuer des traitements directement sur cette image (*i.e.* sans avoir à la déchiffrer) de manière à y dissimuler les bits d'un message secret.

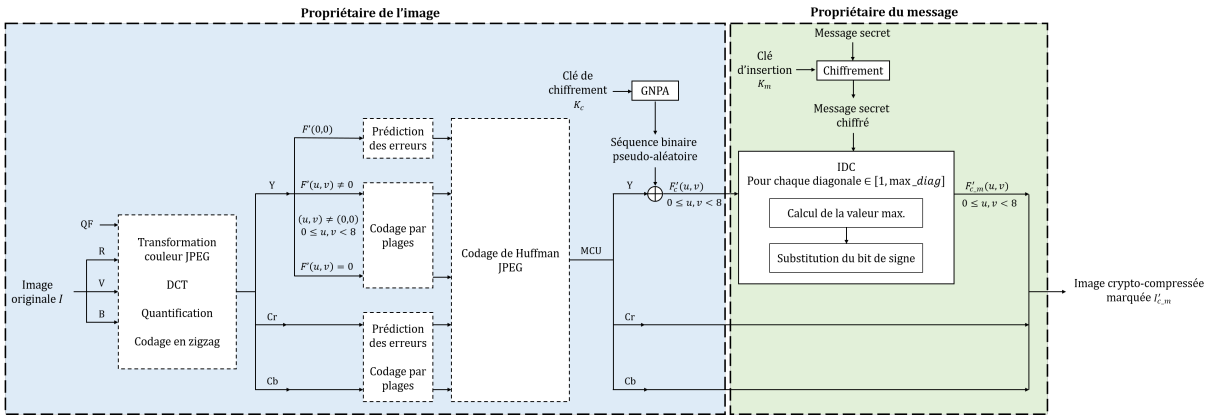


FIGURE 8.2 – Schéma général de la phase d'encodage de la méthode d'IDC hiérarchique dans les images JPEG crypto-compressées.

A partir d'une image originale non-compressée, les premières étapes de la compression JPEG classique sont appliquées jusqu'à la phase de codage de Huffman. Chaque bloc de 8×8 pixels de l'image originale est alors transformé en un MCU de 8×8 codes associés aux coefficients fréquentiels. Chacun des blocs est alors chiffré en effectuant un ou-exclusif entre les valeurs des amplitudes et les bits d'une séquence pseudo-aléatoire générée à partir d'une clé de chiffrement, similairement à la méthode de

crypto-compression décrite dans le chapitre 7.

Dans le flux binaire JPEG crypto-compressé, nous adoptons alors une configuration en forme d'échiquier pour réaliser l'IDC dans un MCU sur deux. En effet, la moitié des MCU n'est pas modifiée afin d'être utilisée pour la prédiction lors de la phase de reconstruction. Chaque MCU utilisé pour l'IDC est parcouru de façon hiérarchique – diagonale par diagonale – des coefficients basses-fréquences aux coefficients hautes-fréquences. Dans chaque diagonale comportant au moins un coefficient AC non nul, nous sélectionnons alors le coefficient qui a la plus grande valeur d'amplitude (en valeur absolue) et remplaçons son MSB (*i.e.* son signe) par un bit du message secret, comme pour les méthodes d'IDCDC présentées dans les chapitres 4 et 5. De plus, les MSB de chaque autre coefficient sont inversés pour introduire du bruit et améliorer la prédiction, et donc les performances lors de la phase de reconstruction.

Durant la phase de décodage, le message secret peut être extrait en examinant le MSB du coefficient qui a la plus grande valeur d'amplitude dans chaque diagonale de chaque MCU. Par ailleurs, afin de reconstruire l'image en clair, la connaissance de la clé de chiffrement est nécessaire. Tous les coefficients DC et les MCU non-utilisés pour l'IDC sont d'abord déchiffrés. Par ailleurs, les coefficients AC des MCU marqués par les bits du message secret sont initialisés à zéro. Pour chaque diagonale de chaque MCU marqué, deux scénarios doivent être considérés : 1) aucun des coefficients chiffrés n'a été modifié, 2) les signes de tous les coefficients chiffrés ont été inversés. Les deux MCU associés sont alors calculés et déchiffrés. Le décodage entropique, la quantification inverse et la DCT inverse sont appliqués à ces deux MCU pour obtenir les deux configurations possibles des blocs de pixels en clair. La corrélation entre les blocs de pixels voisins dans le domaine clair est ensuite exploitée en calculant un score de similarité entre chacune des deux configurations et les blocs de pixels voisins restés en clair. La configuration la plus corrélée avec les blocs de pixels voisins indique les valeurs correctes des coefficients sur la diagonale.

Des résultats préliminaires sont présentés dans un article pour la conférence EUSIPCO 2020 [115].

Correction d'images chiffrées bruitées

Dans le chapitre 6, nous avons proposé une approche efficace pour corriger les images chiffrées bruitées. Notre algorithme s'appuie sur l'hypothèse qu'au plus un bit par bloc de pixels est altéré par le bruit. En pratique, cette hypothèse n'est pas toujours vérifiée. En particulier, si la distribution du bruit n'est pas uniforme – par exemple en cas de bruit blanc gaussien – plusieurs bits au sein du même bloc de pixels peuvent être altérés. Dans certains cas, il est même possible qu'un paquet de données complet soit perdu lors de la transmission. Notre algorithme tel qu'il est décrit dans ce manuscrit peut être utilisé pour localiser ces blocs de pixels mal reconstruits. En effet, les phases d'initialisation et de correction impliquent l'utilisation d'un classifieur permettant de calculer un score. Grâce à ce score de classification, il est possible de discriminer les blocs de pixels mal déchiffrés des blocs de pixels en clair. Si malgré la phase de correction un bloc est identifié comme étant mal déchiffré, ses blocs voisins dans le domaine clair peuvent alors être utilisés pour reconstruire une approximation du bloc de pixels associé

dans l'image originale par interpolation.

Cependant, si le bloc en clair reconstruit est semblable au bloc de pixels de l'image originale, les deux blocs ne sont *a priori* pas identiques. Ainsi, il pourrait être intéressant d'investiguer ce problème. Pour cela, lors de la phase de correction de notre algorithme, toutes les configurations possibles associées au bloc de pixels à corriger doivent être générées et analysées. Notons que supposer que plus d'un bit par bloc de pixels est altéré par le bruit entraîne un plus grand nombre de configurations à examiner et donc une augmentation de la complexité algorithmique.

Liste de publications

Revues internationales

— [106] **P. Puteaux** et W. Puech, An Efficient MSB Prediction-Based Method for High-Capacity Reversible Data Hiding in Encrypted Images, IEEE Trans. Information Forensics and Security 13(7) : 1670-1681 (2018).

— [136] L. Shi, Z. Wang, Z. Qian, N. Huang, **P. Puteaux** et X. Zhang, Distortion Function for Emoji Image Steganography, Computers Mater Continua 59 :943-953 (2019).

— [58] V. Itier, **P. Puteaux** et W. Puech, Recompression of JPEG Crypto-Compressed Images without a Key, IEEE Trans. Circuits and Systems for Video Technology 30(3) : 646-660 (2020).

— [112] **P. Puteaux** et W. Puech, A Recursive Reversible Data Hiding in Encrypted Images Method with a Very High Payload, IEEE Trans. Multimedia (2020), DOI : 10.1109/TMM.2020.2985537.

— [114] **P. Puteaux**, M. Vialle et W. Puech, Homomorphic Encryption-Based LSB Substitution for High Capacity Data Hiding in the Encrypted Domain, IEEE Access 8(1) : 108655-108663 (2020).

— [111] **P. Puteaux** et W. Puech, CFB-then-ECB Mode-Based Image Encryption for an Efficient Correction of Noisy Encrypted Images, IEEE Trans. Circuits and Systems for Video Technology (2020), DOI : 10.1109/TCSVT.2020.3039112.

— [102] **P. Puteaux**, S. Ong, K. Wong et W. Puech, A Survey of (Reversible) Data Hiding in Encrypted Image - The 12 First Years, Journal of Visual Communication and Image Representation (révisions majeures).

Conférences internationales

— [113] **P. Puteaux**, D. Trinel et W. Puech, High-Capacity Data Hiding in Encrypted Images using MSB prediction, IEEE Image Processing Theory, Tools and Applications 2016 : 1-6 (2016).

— [105] **P. Puteaux** et W. Puech, High-Capacity Reversible Data Hiding in Encrypted Images using MSB Prediction, Electronic Imaging, Media Watermarking, Security, and Forensics 2017 : 10-15 (2017).

— [103] **P. Puteaux** et W. Puech, Reversible Data Hiding in Encrypted Images based on Adaptive Local Entropy Analysis, IEEE Image Processing Theory, Tools and Applications 2017 : 1-6 (2017).

— [108] **P. Puteaux** et W. Puech, Noisy Encrypted Image Correction based on Shannon Entropy Measurement in Pixel Blocks of Very Small Size, European Signal Processing Conference 2018 : 161-165 (2018).

— [107] **P. Puteaux** et W. Puech, EPE-based Huge-Capacity Reversible Data Hiding in Encrypted Images, IEEE International Workshop on Information Forensics and Security 2018 : 1-7 (2018).

— [109] **P. Puteaux** et W. Puech, Image Analysis and Processing in the Encrypted Domain, IEEE International Conference on Image Processing 2019 : 3020-3022 (2019).

— [5] S. Beugnon, **P. Puteaux** et W. Puech, Privacy Protection for Social Media based on a Hierarchical Secret Image Sharing Scheme, IEEE International Conference on Image Processing 2019 : 679-683 (2019).

— [115] **P. Puteaux**, Z. Wang, X. Zhang et W. Puech, Hierarchical High Capacity Data Hiding in JPEG Crypto-compressed Images, European Signal Processing Conference 2020.

— [110] **P. Puteaux** et W. Puech, Localization and Correction of Corrupted Pixel Blocks in Noisy Encrypted Images, IEEE Image Processing Theory, Tools and Applications 2020.

Conférences nationales

— [104] **P. Puteaux** et W. Puech, Analyse de la taille minimale d'un bloc de pixels pour avoir une valeur significative de l'entropie : application à la correction d'images chiffrées, COmpression et REprésentation des Signaux Audiovisuels 2017 (2017).

— [4] S. Beugnon, **P. Puteaux** et W. Puech, Protection de la vie privée par partage hybride de photos sur les réseaux sociaux, COmpression et REprésentation des Signaux Audiovisuels 2018 (2018).

— [142] A. Soulier, **P. Puteaux**, F. Comby et W. Puech, Compression sans perte de données GNSS au format RINEX dans un contexte applicatif de véhicules autonomes, COmpression et REprésentation des Signaux Audiovisuels 2020 (2020).

Bibliographie

[1] A. M. Alattar. Reversible watermark using the difference expansion of a generalized integer transform. *IEEE Transactions on Image Processing*, 13(8) :1147–1156, 2004.

[2] V. I. Arnold and A. Avez. *Problèmes ergodiques de la mécanique classique*. Gauthier-Villars, 1967.

[3] P. Bas and T. Furon. Image database of BOWS-2. http ://bows2.ec-lille.fr/.

[4] S. Beugnon, P. Puteaux, and W. Puech. Protection de la vie privée par partage hybride de photos sur les réseaux sociaux. In *COmpression et REprésentation des Signaux Audiovisuels (CORESA)*, 2018.

[5] S. Beugnon, P. Puteaux, and W. Puech. Privacy protection for social media based on a hierarchical secret image sharing scheme. In *IEEE International Conference on Image Processing (ICIP)*, pages 679–683. IEEE, 2019.

[6] G. R. Blakley. Safeguarding cryptographic keys. In *IEEE International Workshop on Managing Requirements Knowledge (MARK)*, pages 313–318. IEEE, 1979.

[7] D. Bouslimi, G. Coatrieux, M. Cozic, and C. Roux. A joint encryption/watermarking system for verifying the reliability of medical images. *IEEE Transactions on Information Technology in Biomedicine*, 16(5) :891–899, 2012.

[8] X. Cao, L. Du, X. Wei, D. Meng, and X. Guo. High capacity reversible data hiding in encrypted images by patch-level sparse representation. *IEEE Transactions on Cybernetics*, 46(5) :1132–1143, 2016.

[9] F. Cayre, C. Fontaine, and T. Furon. Watermarking security : theory and practice. *IEEE Transactions on Signal Processing*, 53(10) :3976–3987, 2005.

[10] M. U. Celik, G. Sharma, A. M. Tekalp, and E. Saber. Lossless generalized-LSB data embedding. *IEEE Transactions on Image Processing*, 14(2) :253–266, 2005.

[11] S. Chan. Recompression of still images. Technical report, University of Kent, Canterbury, UK, March 1992.

[12] R. Chandramouli. A mathematical framework for active steganalysis. *Multimedia Systems*, 9(3) :303–311, 2003.

[13] B. Chen and G. W. Wornell. Quantization index modulation : A class of provably good methods for digital watermarking and information embedding. *IEEE Transactions on Information Theory*, 47(4) :1423–1443, 2001.

[14] G. Chen, Y. Mao, and C. K. Chui. A symmetric image encryption scheme based on 3D chaotic cat maps. *Chaos, Solitons & Fractals*, 21(3) :749–761, 2004.

[15] G. Chen and T. Ueta. Yet another chaotic attractor. *International Journal of Bifurcation and Chaos*, 9(7) :1465–1466, 1999.

[16] K. Chen and C.-C. Chang. High-capacity reversible data hiding in encrypted images based on extended run-length coding and block-based MSB plane rearrangement. *Journal of Visual Communication and Image Representation*, 58 :334–344, 2019.

[17] Y.-C. Chen, C.-W. Shiu, and G. Horng. Encrypted signal-based reversible data hiding with public key cryptosystem. *Journal of Visual Communication and Image Representation*, 25(5) :1164–1170, 2014.

[18] T. S. Cho, S. Avidan, and W. T. Freeman. A probabilistic image jigsaw puzzle solver. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 183–190. IEEE, 2010.

[19] T. Chuman, K. Kurihara, and H. Kiya. On the security of block scrambling-based ETC systems against jigsaw puzzle solver attacks. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2157–2161. IEEE, 2017.

[20] CISCO. Cisco visual networking index : Forecast and trends, 2017–2022, 2018.

[21] D. Coltuc and J.-M. Chassery. Very fast watermarking by reversible contrast mapping. *IEEE Signal Processing Letters*, 14(4) :255–258, 2007.

[22] I. Cox, M. Miller, J. Bloom, J. Fridrich, and T. Kalker. *Digital watermarking and steganography*. Morgan Kaufmann, 2007.

[23] I. J. Cox, J. Kilian, F. T. Leighton, and T. Shamoon. Secure spread spectrum watermarking for multimedia. *IEEE Transactions on Image Processing*, 6(12) :1673–1687, 1997.

[24] M. Czapski and M. Nikodem. Error detection and error correction procedures for the Advanced Encryption Standard. *Designs, Codes and Cryptography*, 49(1-3) :217–232, 2008.

[25] J. Daemen and V. Rijmen. AES proposal : Rijndael. 1999.

[26] R. Davis. The data encryption standard in perspective. *IEEE Communications Society Magazine*, 16(6) :5–9, 1978.

[27] T. D. Denemark, M. Boroumand, and J. Fridrich. Steganalysis features for content-adaptive JPEG steganography. *IEEE Transactions on Information Forensics and Security*, 11(8) :1736–1746, 2016.

[28] I. C. Dragoi, H.-G. Coanda, and D. Coltuc. Improved reversible data hiding in encrypted images based on reserving room after encryption and pixel prediction. In *European Signal Processing Conference (EUSIPCO)*, pages 2186–2190, 2017.

[29] I. C. Dragoi and D. Coltuc. Reversible data hiding in encrypted images based on reserving room after encryption and multiple predictors. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2102–2105. IEEE, 2018.

[30] F. Dufaux and T. Ebrahimi. Scrambling for privacy protection in video surveillance systems. *IEEE Transactions on Circuits and Systems for Video Technology*, 18(8) :1168–1174, 2008.

[31] F. Dufaux, S. Wee, J. Apostolopoulos, and T. Ebrahimi. JPSEC for Secure Imaging in JPEG 2000. In *SPIE Annual Meeting of Optical Science and Technology, Applications of Digital Image Processing*, volume 5558, pages 1–12. SPIE – The International Society for Optical Engineering, 2004.

[32] M. Dworkin. Recommendation for block cipher modes of operation : Methods and techniques. 800-38A, 2001.

[33] T. El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31(4) :469–472, 1985.

[34] D. Engel, T. Stütz, and A. Uhl. A survey on JPEG2000 encryption. *Multimedia Systems*, 15(4) :243–270, 2009.

[35] Z. Erkin, A. Piva, S. Katzenbeisser, R. L. Lagendijk, J. Shokrollahi, G. Neven, and M. Barni. Protection and retrieval of encrypted multimedia content : When cryptography meets signal processing. *EURASIP Journal on Information Security*, 2007 :17, 2007.

[36] B. Ferreira, J. Rodrigues, J. Leitao, and H. Domingos. Privacy-preserving content-based image retrieval in the cloud. In *IEEE Symposium on Reliable Distributed Systems (SRDS)*, pages 11–20. IEEE, 2015.

[37] C. Fontaine and F. Galand. A survey of homomorphic encryption for nonspecialists. *EURASIP Journal on Information Security*, 2007 :1–10, 2007.

[38] J. Fridrich. Image encryption based on chaotic maps. In *IEEE International Conference on Systems, Man, and Cybernetics (SMC). Computational Cybernetics and Simulations*, pages 1105–1110. IEEE, 1997.

[39] J. Fridrich. Symmetric ciphers based on two-dimensional chaotic maps. *International Journal of Bifurcation and Chaos*, 8(06) :1259–1284, 1998.

[40] J. Fridrich, M. Goljan, and R. Du. Invertible authentication. In *Electronic Imaging, Security and Watermarking of Multimedia Contents*, volume 4314, pages 197–208. International Society for Optics and Photonics, 2001.

[41] J. Fridrich and J. Kodovsky. Rich models for steganalysis of digital images. *IEEE Transactions on Information Forensics and Security*, 7(3) :868–882, 2012.

[42] T. Furon and M. Desoubeaux. Tardos codes for real. In *IEEE International Workshop on Information Forensics and Security (WIFS)*, pages 24–29. IEEE, 2014.

[43] X. Gao, L. An, Y. Yuan, D. Tao, and X. Li. Lossless data embedding using generalized statistical quantity histogram. *IEEE Transactions on Circuits and Systems for Video Technology*, 21(8) :1061–1070, 2011.

[44] H. Ge, Y. Chen, Z. Qian, and J. Wang. A high capacity multi-level approach for reversible data hiding in encrypted images. *IEEE Transactions on Circuits and Systems for Video Technology*, 29(8) :2285–2295, 2019.

[45] M. Goljan, J. J. Fridrich, and R. Du. Distortion-free data embedding for images. In *International Workshop on Information Hiding (IH)*, pages 27–41. Springer, 2001.

[46] M. Grangetto, E. Magli, and G. Olmo. Multimedia selective encryption by means of randomized arithmetic coding. *IEEE Transactions on Multimedia*, 8(5) :905–917, 2006.

[47] Z.-H. Guan, F. Huang, and W. Guan. Chaos-based image encryption algorithm. *Physics Letters A*, 346(1) :153–157, 2005.

[48] W. Hamidouche, M. Farajallah, N. Sidaty, S. El Assad, and O. Deforges. Real-time selective video encryption based on the chaos system in scalable HEVC extension. *Signal Processing : Image Communication*, 58 :73–86, 2017.

[49] H. He and J. Zhang. Cryptanalysis on majority-voting based self-recovery watermarking scheme. *Telecommunication Systems*, 49(2) :231–238, 2012.

[50] T. Hey, S. Tansley, and K. Tolle. *The Fourth Paradigm : Data-Intensive Scientific Discovery*. Microsoft Research, 2009.

[51] W. Hong, T.-S. Chen, and H.-Y. Wu. An improved reversible data hiding in encrypted images using side match. *IEEE Signal Processing Letters*, 19(4) :199–202, 2012.

[52] C.-Y. Hsu, C.-S. Lu, and S.-C. Pei. Image feature extraction in encrypted domain with privacy-preserving SIFT. *IEEE Transactions on Image Processing*, 21(11) :4593–4607, 2012.

[53] X. Hu, W. Zhang, H. Hu, and N. Yu. Non-local denoising in encrypted images. In *International Conference on Internet of Vehicles (IOV)*, pages 386–395. Springer, 2014.

[54] F. Huang, J. Huang, and Y.-Q. Shi. New framework for reversible data hiding in encrypted domain. *IEEE Transactions on Information Forensics and Security*, 11(12) :2777–2789, 2016.

[55] Independent JPEG Group. https ://www.ijg.org/, 1991.

[56] N. Islam, Z. Shahid, and W. Puech. Denoising and error correction in noisy AES-encrypted images using statistical measures. *Signal Processing : Image Communication*, 41(C) :15–27, 2016.

[57] ISO/IEC 10918-5 :2013. Information technology — Digital compression and coding of continuous-tone still images :JPEG File Interchange Format (JFIF) — Part 5. Technical report, International Organization for Standardization, 2013.

[58] V. Itier, P. Puteaux, and W. Puech. Recompression of JPEG crypto-compressed images without a key. *IEEE Transactions on Circuits and Systems for Video Technology*, 30(3) :646–660, 2020.

[59] A. K. Jain, J.-E. Lee, R. Jin, and N. Gregg. Content-based image retrieval : An application to tattoo images. In *IEEE International Conference on Image Processing (ICIP)*, pages 2745–2748. IEEE, 2009.

[60] JEITA CP-3451E. Exchangeable image file format for digital still cameras : Exif Version 2.32. Technical report, Japan Electronics and Information Technology Industries Association, 2002.

[61] R. Jiang, H. Zhou, W. Zhang, and N. Yu. Reversible data hiding in encrypted three-dimensional mesh models. *IEEE Transactions on Multimedia*, 20(1) :55–67, 2017.

[62] M. Johnson, P. Ishwar, V. Prabhakaran, D. Schonberg, and K. Ramchandran. On compressing encrypted data. *IEEE Transactions on Signal Processing*, 52(10) :2992–3006, 2004.

[63] L. Kamstra and H. J. Heijmans. Reversible data embedding into images using wavelet techniques and sorting. *IEEE Transactions on Image Processing*, 14(12) :2082–2090, 2005.

[64] P. Karn, P. Metzger, and W. Simpson. The ESP triple DES transform. *RFC1851*, 1995.

[65] Y. Ke, M. Zhang, and J. Liu. Separable multiple bits reversible data hiding in encrypted domain. In *International Workshop on Digital Watermarking (IWDW)*, pages 470–484. Springer, 2016.

[66] Y. Ke, M. Zhang, J. Liu, T. Su, and X. Yang. A multilevel reversible data hiding scheme in encrypted domain based on LWE. *Journal of Visual Communication and Image Representation*, 54 :133–144, 2018.

[67] A. Kerckhoffs. La cryptographie militaire. *Journal des sciences militaires*, IX :5–38, 161–191, 1883.

[68] A. A. Kumar and A. Makur. Distributed source coding based encryption and lossless compression of gray scale and color images. In *IEEE Workshop on Multimedia Signal Processing (MMSP)*, pages 760–764. IEEE, 2008.

[69] K. Kurihara, S. Imaizumi, S. Shiota, and H. Kiya. An encryption-then-compression system for lossless image compression standards. *IEICE Transactions on Information and Systems*, 100(1) :52–56, 2017.

[70] K. Kurihara, S. Shiota, and H. Kiya. An encryption-then-compression system for JPEG standard. In *IEEE Picture Coding Symposium (PCS)*, pages 119–123. IEEE, 2015.

[71] E. Y. Lam and J. W. Goodman. A mathematical analysis of the DCT coefficient distributions for images. *IEEE Transactions on Image Processing*, 9(10) :1661–1666, 2000.

[72] R. Lazzeretti and M. Barni. Lossless compression of encrypted grey-level and color images. In *European Signal Processing Conference (EUSIPCO)*, pages 1–5, 2008.

[73] W. Li and Y. Yuan. A leak and its remedy in JPEG image encryption. *International Journal of Computer Mathematics*, 84(9) :1367–1378, 2007.

[74] X. Li, J. Li, B. Li, and B. Yang. High-fidelity reversible data hiding scheme based on pixel-value-ordering and prediction-error expansion. *Signal Processing*, 93(1) :198–205, 2013.

[75] X. Li, B. Yang, and T. Zeng. Efficient reversible watermarking based on adaptive prediction-error expansion and pixel selection. *IEEE Transactions on Image Processing*, 20(12) :3524–3533, 2011.

[76] S. Lian, J. Sun, and Z. Wang. A novel image encryption scheme based-on JPEG encoding. In *IEEE International Conference on Information Visualisation (IV)*, pages 217–220. IEEE, 2004.

[77] S. Lian, J. Sun, and Z. Wang. Perceptual cryptography on JPEG2000 compressed images or videos. In *IEEE International Conference on Computer and Information Technology (CIT)*, pages 78–83. IEEE, 2004.

[78] W. Liu, W. Zeng, L. Dong, and Q. Yao. Efficient compression of encrypted grayscale images. *IEEE Transactions on Image Processing*, 19(4) :1097–1102, 2009.

[79] W. Lu, A. Swaminathan, A. L. Varna, and M. Wu. Enabling search over encrypted multimedia databases. In *Electronic Imaging, Media Forensics and Security*, volume 7254, pages 1–11. International Society for Optics and Photonics, 2009.

[80] W. Lu, A. L. Varna, A. Swaminathan, and M. Wu. Secure image retrieval through feature protection. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1533–1536. IEEE, 2009.

[81] K. Ma, W. Zhang, X. Zhao, N. Yu, and F. Li. Reversible data hiding in encrypted images by reserving room before encryption. *IEEE Transactions on Information Forensics and Security*, 8(3) :553–562, 2013.

[82] Y. Mao, G. Chen, and S. Lian. A novel fast image encryption scheme based on 3D chaotic baker maps. *International Journal of Bifurcation and Chaos*, 14(10) :3613–3624, 2004.

[83] N. D. Memon, X. Wu, V. Sippy, and G. Miller. Interband coding extension of the new lossless JPEG standard. In *Visual Communications and Image Processing (VCIP)*, volume 3024, pages 47–58. International Society for Optics and Photonics, 1997.

[84] K. Minemura, Z. Moayed, K. Wong, X. Qi, and K. Tanaka. JPEG image scrambling without expansion in bitstream size. In *IEEE International Conference on Image Processing (ICIP)*, pages 261–264. IEEE, 2012.

[85] R. H. Morelos-Zaragoza. *The art of error correcting coding*. John Wiley & Sons, 2006.

[86] M. Naor and A. Shamir. Visual cryptography. In *Workshop on the Theory and Application of of Cryptographic Techniques*, pages 1–12. Springer, 1994.

[87] Z. Ni, Y.-Q. Shi, N. Ansari, and W. Su. Reversible data hiding. *IEEE Transactions on Circuits and Systems for Video Technology*, 16(3) :354–362, 2006.

[88] R. Norcen and A. Uhl. Encryption of wavelet-coded imagery using random permutations. In *IEEE International Conference on Image Processing (ICIP)*, pages 3431–3434. IEEE, 2004.

[89] V. Ocheretnij, G. Kouznetsov, M. Gossel, and R. Karri. On-line error detection and bist for the AES encryption algorithm with different S-box implementations. In *IEEE International On-Line Testing Symposium (IOLTS)*, pages 141–146. IEEE, 2005.

[90] S. Ong, K. Wong, and K. Tanaka. Scrambling-embedding for JPEG compressed image. *Signal Processing*, 109 :38–53, 2015.

[91] P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT)*, pages 223–238. Springer, 1999.

[92] C. Pavlopoulou, A. C. Kak, and C. E. Brodley. Content-based image retrieval for medical imagery. In *Medical Imaging, PACS and Integrated Medical Information Systems : Design and Evaluation*, volume 5033, pages 85–96. International Society for Optics and Photonics, 2003.

[93] A. Pedrouzo-Ulloa, J. R. Troncoso-Pastoriza, and F. Pérez-González. Image denoising in the encrypted domain. In *8th IEEE International Workshop on Information Forensics and Security (WIFS)*, pages 1–6, 2016.

[94] T. Pevny, J. Fridrich, and A. D. Ker. From blind to quantitative steganalysis. *IEEE Transactions on Information Forensics and Security*, 7(2) :445–454, 2011.

[95] M. Pinto, W. Puech, and G. Subsol. Protection of JPEG compressed e-comics by selective encryption. In *IEEE International Conference on Image Processing (ICIP)*, pages 4588–4592. IEEE, 2013.

[96] K. M. Poh, G. S.and Martin. An efficient buyer-seller watermarking protocol based on chameleon encryption. In *International Workshop on Digital Watermarking (IWDW)*, pages 433–447. Springer, 2009.

[97] M. Preishuber, T. Hütter, S. Katzenbeisser, and A. Uhl. Depreciating motivation and empirical security analysis of chaos-based image and video encryption. *IEEE Transactions on Information Forensics and Security*, 13(9) :2137–2150, 2018.

[98] P. Premaratne and M. Premaratne. Key-based scrambling for secure image communication. In *International Conference on Intelligent Computing (ICIC)*, pages 259–263. Springer, 2012.

[99] W. Puech, A. G. Bors, and J. M. Rodrigues. Protection of colour images by selective encryption. In *Advanced Color Image Processing and Analysis*, pages 397–421. Springer, 2013.

[100] W. Puech, M. Chaumont, and O. Strauss. A reversible data hiding method for encrypted images. In *Electronic Imaging, Security, Forensics, Steganography, and Watermarking of Multimedia Contents*, volume X, pages 68191E–68191E. International Society for Optics and Photonics, 2008.

[101] W. Puech and J. M. Rodrigues. Crypto-compression of medical images by selective encryption of DCT. In *European Signal Processing Conference (EUSIPCO)*, pages 1–4, 2005.

[102] P. Puteaux, S. Ong, K. Wong, and W. Puech. A survey of (reversible) data hiding in encrypted image - the 12 first years. *Jounal of Visual Communication and Image Representation (révisions majeures)*, 2020.

[103] P. Puteaux and W. Puech. Reversible data hiding in encrypted images based on adaptive local entropy analysis. In *IEEE International Conference on Image Processing Theory, Tools and Applications (IPTA)*, pages 1–6. IEEE.

[104] P. Puteaux and W. Puech. Analyse de la taille minimale d'un bloc de pixels pour avoir une valeur significative de l'entropie : application à la correction d'images chiffrées. In *COmpression et REprésentation des Signaux Audiovisuels (CORESA)*, 2017.

[105] P. Puteaux and W. Puech. High-capacity reversible data hiding in encrypted images using MSB prediction. In *Electronic Imaging, Media Watermarking, Security, and Forensics*, number 7, pages 10–15. Society for Imaging Science and Technology, 2017.

[106] P. Puteaux and W. Puech. An efficient MSB prediction-based method for high-capacity reversible data hiding in encrypted images. *IEEE Transactions on Information Forensics and Security*, 13(7) :1670–1681, 2018.

[107] P. Puteaux and W. Puech. EPE-based huge-capacity reversible data hiding in encrypted images. In *IEEE Workshop on Information Forensics and Security (WIFS)*, pages 1–7. IEEE, 2018.

[108] P. Puteaux and W. Puech. Noisy encrypted image correction based on Shannon entropy measurement in pixel blocks of very small size. In *European Signal Processing Conference (EUSIPCO)*, pages 161–165, 2018.

[109] P. Puteaux and W. Puech. Image analysis and processing in the encrypted domain. In *IEEE International Conference on Image Processing (ICIP)*, pages 3020–3022. IEEE, 2019.

[110] P. Puteaux and W. Puech. Localization and correction of corrupted pixel blocks in noisy encrypted images. In *International Conference on Image Processing Theory, Tools and Applications (IPTA)*. IEEE, 2020.

[111] P. Puteaux and W. Puech. CFB-then-ECB mode-based image encryption for an efficient correction of noisy encrypted images. *IEEE Transactions on Circuits and Systems for Video Technology*, 2020, DOI : 10.1109/TCSVT.2020.3039112.

[112] P. Puteaux and W. Puech. A recursive reversible data hiding in encrypted images method with a very high capacity. *IEEE Transactions on Multimedia*, pages 1–1, 2020, DOI : 10.1109/TMM.2020.2985537.

[113] P. Puteaux, D. Trinel, and W. Puech. High-capacity data hiding in encrypted images using MSB prediction. In *International Conference on Image Processing Theory, Tools and Applications (IPTA)*, pages 1–6. IEEE, 2016.

[114] P. Puteaux, M. Vialle, and W. Puech. Homomorphic encryption-based LSB substitution for high capacity data hiding in the encrypted domain. *IEEE Access*, 8 :108655–108663, 2020.

[115] P. Puteaux, Z. Wang, X. Zhang, and W. Puech. Hierarchical high capacity data hiding in JPEG crypto-compressed images. In *European Signal Processing Conference (EUSIPCO)*, 2020.

[116] Y. Puyang, Z. Yin, and Z. Qian. Reversible data hiding in encrypted images with two-MSB prediction. In *IEEE Workshop on Information Forensics and Security (WIFS)*, pages 1–7. IEEE, 2018.

[117] Z. Qian, H. Xu, X. Luo, and X. Zhang. New framework of reversible data hiding in encrypted JPEG bitstreams. *IEEE Transactions on Circuits and Systems for Video Technology*, 29(2) :351–362, 2018.

[118] Z. Qian and X. Zhang. Reversible data hiding in encrypted images with distributed source encoding. *IEEE Transactions on Circuits and Systems for Video Technology*, 26(4) :636–646, 2016.

[119] Z. Qian, X. Zhang, and S. Wang. Reversible data hiding in encrypted JPEG bitstream. *IEEE Transactions on Multimedia*, 16(5) :1486–1491, 2014.

[120] Z. Qian, H. Zhou, X. Zhang, and W. Zhang. Separable reversible data hiding in encrypted JPEG bitstreams. *IEEE Transactions on Dependable and Secure Computing*, 15(6) :1055–1067, 2016.

[121] C. Qin, P. Ji, X. Zhang, J. Dong, and J. Wang. Fragile image watermarking with pixel-wise recovery based on overlapping embedding strategy. *Signal Processing*, 138 :280–293, 2017.

[122] R. Reininger and J. Gibson. Distributions of the two-dimensional DCT coefficients for images. *IEEE Transactions on Communications*, 31(6) :835–839, 1983.

[123] T. Richter, A. Artusi, and T. Ebrahimi. JPEG XT : A new family of JPEG backward-compatible standards. *IEEE Multimedia*, 23(3) :80–88, 2016.

[124] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2) :120–126, 1978.

[125] J. M. Rodrigues, W. Puech, and A. G. Bors. Selective encryption of human skin in jpeg images. In *IEEE International Conference on Image Processing (ICIP)*, pages 1981–1984. IEEE, 2006.

[126] V. Sachnev, H.-J. Kim, J. Nam, S. Suresh, and Y.-Q. Shi. Reversible watermarking algorithm using sorting and prediction. *IEEE Transactions on Circuits and Systems for Video Technology*, 19(7) :989–999, 2009.

[127] S. M. SaghaianNejadEsfahani, Y. Luo, and S.-C. S. Cheung. Privacy protected image denoising with secret shares. In *IEEE International Conference on Image Processing (ICIP)*, pages 253–256. IEEE, 2012.

[128] A. Said. Measuring the strength of partial encryption schemes. In *IEEE International Conference on Image Processing (ICIP)*, pages 1126–1129. IEEE, 2005.

[129] G. Schaefer and M. Stich. UCID : an uncompressed color image database. In *Electronic Imaging, Storage and Retrieval Methods and Applications for Multimedia*, volume 5307, pages 472 – 480. International Society for Optics and Photonics, SPIE, 2003.

[130] J. Scharinger and F. Pichler. Efficient image encryption based on chaotic maps. *Pattern Recognition*, pages 159–170, 1996.

[131] Z. Shahid, M. Chaumont, and W. Puech. Fast protection of H. 264/AVC by selective encryption of CAVLC and CABAC for I and P frames. *IEEE Transactions on Circuits and Systems for Video Technology*, 21(5) :565–576, 2011.

[132] A. Shamir. How to share a secret. *Communications of the ACM*, 22(11) :612–613, 1979.

[133] C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27 :379–423, 1948.

[134] C. E. Shannon. Communication theory of secrecy systems. *Bell System Technical Journal*, 28(4) :656–715, 1949.

[135] C. Shi and B. Bhargava. A fast MPEG video encryption algorithm. In *ACM International Conference on Multimedia (MM)*, pages 81–88. ACM, 1998.

[136] L. Shi, Z. Wang, Z. Qian, N. Huang, P. Puteaux, and X. Zhang. Distortion function for emoji image steganography. *Computers Mater Continua*, 59 :943–953, 2019.

[137] Y.-Q. Shi, X. Li, X. Zhang, H.-T. Wu, and B. Ma. Reversible data hiding : advances in the past two decades. *IEEE Access*, 4 :3210–3237, 2016.

[138] C.-W. Shiu, Y.-C. Chen, and W. Hong. Encrypted image-based reversible data hiding with public key cryptography from difference expansion. *Signal Processing : Image Communication*, 39 :226–233, 2015.

[139] G. J. Simmons. The prisoners' problem and the subliminal channel. In *Advances in Cryptology*, pages 51–67. Springer, 1984.

[140] G. Singh. A study of encryption algorithms (RSA, DES, 3DES and AES) for information security. *International Journal of Computer Applications*, 67(19), 2013.

[141] D. Slepian and J. Wolf. Noiseless coding of correlated information sources. *IEEE Transactions on Information Theory*, 19(4) :471–480, 1973.

[142] A. Soulier, P. Puteaux, F. Comby, and W. Puech. Compression sans perte de données GNSS au format RINEX dans un contexte applicatif de véhicules autonomes. In *COmpression et REprésentation des Signaux Audiovisuels (CORESA)*, 2020.

[143] W.-L. Tai, C.-M. Yeh, and C.-C. Chang. Reversible data hiding based on histogram modification of pixel differences. *IEEE Transactions on Circuits and Systems for Video Technology*, 19(6) :906–910, 2009.

[144] H. Tao, L. Chongmin, J. M. Zain, and A. N. Abdalla. Robust image watermarking theories and techniques : A review. *Journal of Applied Research and Technology*, 12(1) :122–138, 2014.

[145] C. Thien and J. Lin. Secret image sharing. *Computers & Graphics*, 26(5) :765–770, 2002.

[146] D. M. Thodi and J. J. Rodriguez. Prediction-error based reversible watermarking. In *IEEE International Conference on Image Processing (ICIP)*, pages 1549–1552. IEEE, 2004.

[147] D. M. Thodi and J. J. Rodriguez. Expansion embedding techniques for reversible watermarking. *IEEE Transactions on Image Processing*, 16(3) :721–730, 2007.

[148] J. Tian. Reversible watermarking by difference expansion. In *ACM Workshop on Multimedia and Security (MMSec)*, volume 19. ACM, 2002.

[149] J. Tian. Reversible data embedding using a difference expansion. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(8) :890–896, 2003.

[150] P. Tsai, Y.-C. Hu, and H.-L. Yeh. Reversible image hiding scheme using predictive coding and histogram shifting. *Signal processing*, 89(6) :1129–1143, 2009.

[151] A. Unterweger and A. Uhl. Length-preserving bit-stream-based JPEG encryption. In *ACM Workshop on Multimedia and Security (MMSec)*, pages 85–90. ACM, 2012.

[152] K. Usman, H. Juzoji, I. Nakajima, S. Soegidjoko, M. Ramdhani, T. Hori, and S. Igi. Medical image encryption based on pixel arrangement and random permutation for transmission security. In *IEEE International Conference on e-Health Networking, Application and Services (Healthcom)*, pages 244–247. IEEE, 2007.

[153] M. Van Droogenbroeck. Partial encryption of images for real-time applications. In *IEEE Signal Processing Symposium*, pages 11–15. IEEE, 2004.

[154] M. Van Droogenbroeck and R. Benedett. Techniques for a selective encryption of uncompressed and compressed images. In *International Conference on Advanced Concepts for Intelligent Vision Systems (ACIVS)*, pages 90–97. Springer, 2002.

[155] G. S. Vernam. Cipher printing telegraph systems : For secret wire and radio telegraphic communications. *Journal of the AIEE*, 45(2) :109–115, 1926.

[156] G. K. Wallace. The JPEG still picture compression standard. *IEEE Transactions on Consumer Electronics*, 38(1) :XVIII–XXXIV, 1992.

[157] M.-S. Wang and W.-C. Chen. A majority-voting based watermarking scheme for color image tamper detection and recovery. *Computer Standards & Interfaces*, 29(5) :561–570, 2007.

[158] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment : from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4) :600–612, 2004.

[159] S. B. Wicker. *Error control systems for digital communication and storage*, volume 1. Prentice hall Englewood Cliffs, 1995.

[160] K. S. Wong and K. Tanaka. Data embedding for geo-tagging any contents in smart device. In *IEEE Region 10 Symposium*, pages 527–530. IEEE, 2014.

[161] C. V. Wright, W.-C. Feng, and F. Liu. Thumbnail-preserving encryption for JPEG. In *ACM Workshop on Information Hiding and Multimedia Security (IH&MMSec)*, pages 141–146. ACM, 2015.

[162] H.-T. Wu, Y.-M. Cheung, and J. Huang. Reversible data hiding in Paillier cryptosystem. *Journal of Visual Communication and Image Representation*, 40 :765–771, 2016.

[163] K. Wu, R. Karri, G. Kuznetsov, and M. Goessel. Low cost concurrent error detection for the advanced encryption standard. In *IEEE International Test Conference (ITC)*, pages 1242–1248. IEEE, 2004.

[164] X. Wu and W. Sun. High-capacity reversible data hiding in encrypted images by prediction error. *Signal Processing*, 104 :387–400, 2014.

[165] Y. Wu, J. P. Noonan, and S. Agaian. NPCR and UACI randomness tests for image encryption. *Cyber journals : Multidisciplinary Journals in Science and Technology, Journal of Selected Areas in Telecommunications (JSAT)*, 1(2) :31–38, 2011.

[166] Z. Xia, X. Wang, L. Zhang, Z. Qin, X. Sun, and K. Ren. A privacy-preserving and copy-deterrence content-based image retrieval scheme in cloud computing. *IEEE Transactions on Information Forensics and Security*, 11(11) :2594–2608, 2016.

[167] T. Xiang, K.-W. Wong, and X. Liao. Selective image encryption using a spatio-temporal chaotic system. *Chaos : An Interdisciplinary Journal of Nonlinear Science*, 17(2) :023115, 2007.

[168] D. Xiao, Y. Xiang, H. Zheng, and Y. Wang. Separable reversible data hiding in encrypted image based on pixel value ordering and additive homomorphism. *Journal of Visual Communication and Image Representation*, 45 :1–10, 2017.

[169] D. Xu, R. Wang, and Y. Q. Shi. Reversible data hiding in encrypted H. 264/AVC video streams. In *International Workshop on Digital Watermarking (IWDW)*, pages 141–152. Springer, 2013.

[170] S. Yi and Y. Zhou. Separable and reversible data hiding in encrypted images using parametric binary tree labeling. *IEEE Transactions on Multimedia*, 21(1) :51–64, 2019.

[171] W. Zhang, K. Ma, and N. Yu. Reversibility improved data hiding in encrypted images. *Signal Processing*, 94 :118–127, 2014.

[172] X. Zhang. Reversible data hiding in encrypted image. *IEEE Signal Processing Letters*, 18(4) :255–258, 2011.

[173] X. Zhang. Separable reversible data hiding in encrypted image. *IEEE Transactions on Information Forensics and Security*, 7(2) :826–832, 2012.

[174] X. Zhang, J. Long, Z. Wang, and H. Cheng. Lossless and reversible data hiding in encrypted images with public-key cryptography. *IEEE Transactions on Circuits and Systems for Video Technology*, 26(9) :1622–1631, 2016.

[175] X. Zhang and S. Wang. Statistical fragile watermarking capable of locating individual tampered pixels. *IEEE Signal Processing Letters*, 14(10) :727–730, 2007.

[176] Z.-L. Zhu, W. Zhang, K.-W. Wong, and H. Yu. A chaos-based symmetric image encryption scheme using a bit-level permutation. *Information Sciences*, 181(6) :1171–1186, 2011.

[177] P. R. Zimmermann. *The official PGP user's guide*, volume 5. MIT press Cambridge, 1995.

Résumé

Durant cette dernière décennie, la sécurité des données multimédia, telles que les images, les vidéos et les données 3D, est devenue un problème majeur incontournable. Avec le développement d'Internet, de plus en plus d'images sont transmises sur les réseaux et stockées sur le *cloud*. Ces données visuelles sont généralement à caractère personnel ou peuvent avoir une valeur marchande. Ainsi, des outils informatiques permettant d'assurer leur sécurité ont été développés.

Le but du chiffrement est de garantir la confidentialité visuelle des images en rendant aléatoire leur contenu. Par ailleurs, pendant la transmission ou l'archivage des images chiffrées, il est souvent nécessaire de les analyser ou de les traiter sans connaître leur contenu original, ni la clé secrète utilisée pendant la phase de chiffrement. Ce sujet de thèse propose de se pencher sur cette problématique. En effet, de nombreuses applications existent telles que le partage d'images secrètes, l'insertion de données cachées dans des images chiffrées, l'indexation et la recherche d'images dans des bases de données chiffrées, la recompression d'images crypto-compressées, ou encore la correction d'images chiffrées bruitées.

Dans un premier axe de recherche, nous présentons tout d'abord une nouvelle méthode d'insertion de données cachées haute capacité dans le domaine chiffré. Dans la plupart des approches de l'état-de-l'art, les valeurs des bits de poids faible sont remplacées pour réaliser l'insertion d'un message secret. Nous prenons ces approches à contre-pied en proposant de prédire les bits de poids fort. Ainsi, une charge utile nettement supérieure est obtenue, tout en conservant une haute qualité de l'image reconstruite. Par la suite, nous montrons qu'il est en effet possible de traiter récursivement tous les plans binaires d'une image pour réaliser l'insertion de données cachées dans le domaine chiffré.

Dans un second axe de recherche, nous expliquons comment exploiter des mesures statistiques (entropie de Shannon et réseau neuronal convolutif) dans des blocs de pixels de petite taille *i.e.* avec peu d'échantillons) pour différencier un bloc en clair d'un bloc chiffré dans une image. Nous utilisons alors cette analyse dans une application à la correction d'images chiffrées bruitées.

Enfin, le troisième axe de recherche développé dans ces travaux de thèse porte sur la recompression d'images crypto-compressées. Dans le domaine clair, les images JPEG peuvent être recompressées avant leur transmission sur des réseaux bas débit, mais l'opération est bien plus complexe dans le domaine chiffré. Nous proposons alors une méthode de recompression des images JPEG crypto-compressées directement dans le domaine chiffré et sans connaître la clé secrète, en s'appuyant sur un décalage binaire des coefficients réorganisés.

Abstract

During the last decade, the security of multimedia data, such as images, videos and 3D data, has become a major issue. With the development of the Internet, more and more images are transmitted over networks and stored in the cloud. This visual data is usually personal or may have a market value. Thus, computer tools have been developed to ensure their security.

The purpose of encryption is to guarantee the visual confidentiality of images by making their content random. Moreover, during the transmission or archiving of encrypted images, it is often necessary to analyze or process them without knowing their original content or the secret key used during the encryption phase. This PhD thesis proposes to address this issue. Indeed, many applications exist such as secret images sharing, data hiding in encrypted images, images indexing and retrieval in encrypted databases, recompression of crypto-compressed images, or correction of noisy encrypted images.

In a first line of research, we present a new method of high-capacity data hiding in encrypted images. In most state-of-the-art approaches, the values of the least significant bits are replaced to achieve the embedding of a secret message. We take the opposing view of these approaches by proposing to predict the most significant bits. Thus, a significantly higher payload is obtained, while maintaining a high quality of the reconstructed image. Subsequently, we showed that it was possible to recursively process all bit planes of an image to achieve data hiding in the encrypted domain.

In a second line of research, we explain how to exploit statistical measures (Shannon entropy and convolutional neural network) in small pixel blocks (*i.e.* with few samples) to discriminate a clear pixel block from an encrypted pixel block in an image. We then use this analysis in an application to correct noisy encrypted images.

Finally, the third line of research developed in this thesis concerns the recompression of crypto-compressed images. In the clear domain, JPEG images can be recompressed before transmission over low-speed networks, but the operation is much more complex in the encrypted domain. We then proposed a method for recompressing crypto-compressed JPEG images directly in the encrypted domain and without knowing the secret key, using a bit shift of the reorganized coefficients.