Archivage Sécurisé d'Images

Jean Motsch

N° d'ordre :

THÈSE

présentée devant

l'INSTITUT NATIONAL DES SCIENCES APPLIQUÉES
DE RENNES

pour obtenir le grade de

Docteur

spécialité : *Traitement du Signal et des Images*

par

Jean MOTSCH

Archivage sécurisé d'images

version provisoire, soutenance prévue le 27 novembre 2008

COMPOSITION DU JURY

Président	
Alexandru SERBANESCU	Professeur à l'Académie Technique Militaire, Bucarest
Rapporteurs	
Béatrice PESQUET-POPESCU	Professeur à Télécom ParisTech
Jean-Pierre GUÉDON	Professeur à Polytech Nantes
Examinateurs	
Yvon ERHEL	Professeur à l'ÉSM de Saint-Cyr, Coëtquidan
Marie BABEL	Maître de conférences à l'INSA de Rennes
Olivier DÉFORGES	Professeur à l'INSA de Rennes

Sommaire

I Codage d'images fixes **7**

1 Codeur d'images LAR **9**

2 Codeur d'images LAR Interleaved S+P **25**

3 Influence de la partition quadtree **37**

II Chiffrement d'images **51**

4 Chiffrement de données **53**

5 Chiffrement d'images **73**

6 Chiffrement partiel à coût nul **95**

7 Chiffrement sélectif du flot LAR iS+P **107**

III Insertion de données **133**

8 Dissimulation de données dans les images **135**

9 Extension de la différence au LAR iS+P **153**

E. P. Jacobs, *La marque jaune*, page 35, case 5.

Introduction générale

Dans une acception ancienne, les archives consistent en un ensemble de documents hors d'usage courant, rassemblés, répertoriés et conservés pour servir à l'histoire d'une collectivité ou d'une individu. Dans une acception plus moderne, en informatique, une archive est un fichier compressé qui peut contenir plusieurs fichiers et répertoires.

À partir de ces définitions, l'archivage de documents photographiques consistait à stocker les clichés, les négatifs et les diapositives dans des conditions optimales propres à leur préservation. Lorsque les fonds concernés sont d'importance, l'accès aux documents est contrôlé et l'utilisation d'index de classification permet de retrouver, en un temps raisonnable, un cliché donné. Lorsque les fonds sont personnels, l'archivage consiste principalement en des albums et des boîtes à chaussures. La problématique est la même, seules changent les échelles en nombre de clichés considérés. Pour préciser ce point, trois contextes applicatifs sont présentées. Le premier concerne une utilisation grand public, le deuxième se concentre sur la transmission d'images extra-terrestres tandis que le troisième s'intéresse à la mise en ligne d'une grande collection de clichés.

Considérons un contexte applicatif grand public pour commencer. Les appareils photographiques numériques (APN) ont progressivement remplacés les appareils photographiques argentiques depuis quelques années. Leur

caractéristique principale, à savoir leur résolution, a augmenté de manière exponentielle les dix dernières années. De quelques centaines de milliers de pixels pour les premiers capteurs des APN grand public, les résolutions sont passés à une dizaine de millions de pixels. Il existe même des boîtiers pour amateurs éclairés dont les capteurs ont cinquante millions de pixels, chacun étant numérisé avec 16 bits. En prenant une moyenne de 5 Mo par clichés, le volume de données équivalent à un film de 36 poses serait de 180 Mo. Néanmoins, ce nombre est tout à fait raisonnable eu égard aux capacités des cartes mémoires et disques durs contemporains. Dans ce contexte, les difficultés apparaissent plutôt lorsque l'on souhaite trier et indexer les images. En effet, il faut *annoter* les clichés et ranger ces annotations dans des fichiers distincts des photographies. Un lien est créé entre d'une part le cliché et d'autre part les données le concernant. De fait, la gestion de ce lien, pour garantir la cohérence entre cliché et annotation ou pour prendre en compte les mises à jour, peut s'avérer une tâche difficile. De même, en cas de copie du cliché, comment s'assurer que les annotations suivent la photographie correspondante ? Pour répondre à ce problème, il peut sembler judicieux d'intégrer les annotations directement aux clichés. Cette intégration peut se faire de deux façons. Une première consiste à ajouter des métadonnées dans un entête normalisé. L'inconvénient majeur est que lors d'un transcodage, ces annotations doivent être manipulées à part de l'image. Une autre méthode d'intégration consiste alors en l'insertion des métadonnées dans le *contenu* du cliché. Ainsi, lorsque l'image est transcodée, les métadonnées persistent dans l'image. La démarche ainsi présentée de manière succinte s'appuie sur les techniques *d'insertion de données*.

Considérons par ailleurs un contexte de transmission d'images *lointaines* pour continuer. La caméra HiRISE[1] est un dispositif de prise de vue à haute résolution développé pour l'exploration martienne. Elle utilise des capteurs linéaires faisant 20264 pixels pour la composante *rouge* et 4048 pixels pour les composantes *bleue*, *verte* et *proche infrarouge*. La conversion analogique-numérique se fait sur 14 bits. La taille des clichés est limitée par la taille de la mémoire embarquée, soit 28 Gbits. Avec une compression de 8 bpp, les dimensions maximales des images *rouges* font donc environ 20000×126000 pixels, soit 2520 MPixels. Pour les autres bandes, les dimensions maximales sont d'environ 4000×126000, soient 504 MPixels. Pour être exact, les images transmises, une fois compressée, ont un volume d'environ 11.2 Gbits, avec un débit de 4.44 bpp. Le système de communication de *Mars Reconnaissance Explorer*, qui emporte la caméra HiRISE, autorise un débit maximal de 6 Mb/s. Ainsi, pour transmettre une image de 5 Gbits, il faut environ quinze minutes.

[1] High Resolution Imaging Science Experiment

Pour une telle application, la principale problématique est d'obtenir une compression efficace, un flot binaire résistant aux erreurs, une description multirésolution, et une complexité calculatoire compatible avec les contraintes des systèmes embarqués spatiaux. De plus, les dimensions des images obtenues rendent intéressantes les approches multirésolutions, en particulier pour faciliter la navigation dans les bases de données images constituées de tous les clichés envoyés. Dans le cas particulier de HiRISE, certaines images sont accessibles au grand public dans un format à la norme JPEG-2000.

Le troisième et dernier cadre applicatif concerne une base de données constituée d'images d'art à haute résolution. Le *Centre de recherche et de restauration des musées de France* (C2RMF) a participé au projet européen CRISATEL[2] qui a développé un système de prise de vues spécifique aux objets d'art. Ce système repose sur une caméra multispectrale prenant des clichés de 240 MPixels. Les bandes spectrales étudiées sont au nombre de 13, dont 10 dans le domaine visible et 3 dans le domaine infrarouge. La résolution spatiale est de 30 microns. La correction dimensionnelle de l'inhomogénéité de la lumière et du bruit de fond permet d'avoir des clichés en *lumière vraie*. La dynamique du capteur est de 16 bits, soit un rapport signal sur bruit d'environ 96 dB. Avec de telles caractéristiques, le volume brut de données à manipuler pour une image dans une bande spectrale est de 480 Mo, et donc, si on considère toutes les bandes spectrales disponibles, un volume de plus de 6 Go. Il apparaît donc que les images sont très volumineuses, longues[3] et coûteuses à acquérir. Néanmoins, elles doivent être accessibles à la communauté scientique, en particulier pour des activités de recherche en histoire de l'art. À cette fin, la base d'images EROS[4] a été constituée. Son fonds consiste en environ 190000 clichés qui correspondent à la numérisation de 19000 peintures et 30000 objets. Cet archivage conséquent peut être consulté à distance au travers du réseau internet. Il met donc en place un mécanisme d'accès sécurisé, réservé aux seuls utilisateurs autorisés. Cependant, ce type de contrôle reste rudimentaire et ne remplit pas la mission de diffusion cet héritage culturel au plus grand nombre.

Afin de proposer de services innovants à cette base de données, le projet ANR[5] TSAR, Transfert Sécurisé d'images d'Art haute Résolution, a été lancé en 2005[6]. La sécurité de la transmission concerne l'information image de haute résolution. L'idée-force du projet était de pouvoir mettre en ligne une

[2]Conservation Restauration Innovation Systems for Image capture and digital Archiving to enhance Training Education and lifelong Learning, programme IST2–1999.

[3]deux heures.

[4]European Research Open System.

[5]Agence Nationale de la Recherche

[6]`http ://www.lirmm.fr/tsar`.

base de données images unique, avec une version basse résolution des images accessible publiquement et une version haute résolution accessible de manière privée. Il s'agit donc de protéger les images en ajoutant des marques durant le transfert, pour les supprimer après réception et authentification. De plus, la gestion des droits d'accès à différents niveaux de résolution d'une image doit être possible, en masquant les données haute résolution par exemple. La figure 1 montre un exemple de telles images. Le projet TSAR repose donc sur la transmission de données cachées de taille importante dans des images elles-même de grandes dimensions. Les domaines de recherche associés concernent la stéganographie, le tatouage numérique, le chiffrement d'image, et, pour assurer des temps de transfert raisonnables, une bonne technique de compression d'images.

(a) basse résolution (b) haute résolution

FIG. 1: une image vue en basse et haute résolution

Les trois contextes applicatifs présentés font apparaître un triptyque indispensable à la mise en œuvre d'un système d'archivage d'images fixes, à savoir :

compression pour faire un usage raisonné des ressources en termes de mémoire, de débit de transmission, d'adéquation aux erreurs du canal et de coût calculatoire ;

sécurité pour permettre un accès différencié aux images et garantir le respect de la propriété intellectuelle et artistique ;

indexation pour faciliter le suivi, la recherche et l'utilisation des données de la base d'images.

Les services demandés peuvent être fournis par l'utilisation conjointe de primitives dédiées. En particulier, la *sécurité* peut être obtenue à l'aide de primitives de chiffrement adaptées aux images. De même, les informations d'*indexation* peuvent être intégrées directement dans le contenu de l'image par des techniques d'insertion de données. Dans ce cadre, ce mémoire va s'attacher à intégrer des solutions de chiffrement et d'insertion de données dans le codeur LAR , un codeur d'images polyvalent. Son organisation va suivre le tryptique évoqué dans le paragraphe précédent.

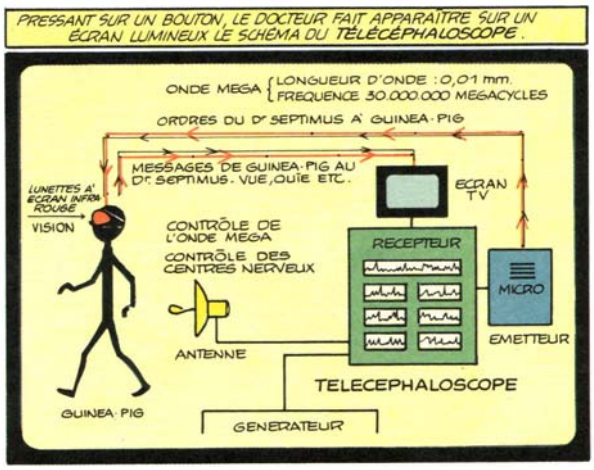
La première partie s'attache à décrire avec quelques détails la famille des codeurs LAR, afin de faire apparaître les bonnes propriétés de ceux-ci. Plus précisement, le chapitre 1 introduit le LAR simple et des éléments d'une approche pyramidale prédictive, tandis que le chapitre 2 s'intéresse au LAR iS+P, pour lequel l'association S+P et approche pyramidale se révèle fructueuse. Dans ces deux chapitres, l'importante d'une partition quadtree guidant l'ensemble du processus apparaît. Cette influence est disséquée dans le chapitre 3. La polyvalence du système de codage est désormais établie.

La seconde partie propose d'intégrer au sein du LAR iS+P un service de chiffrement. Pour cela, le chapitre 4 rappelle la problématique du chiffrement de données, son vocabulaire et les principales structures rencontrées, ainsi que quelques primitives *normalisées* utilisées par la suite. Ensuite, le chapitre 5 dressera un état de l'art du chiffrement dédié aux images. La présentation d'une technique de chiffrement sélectif à coût nul formera le chapitre 6. Enfin, le chapitre 7 introduira une méthode de chiffrement sélectif de flots LAR .

La troisième et dernière partie se concentrera sur l'insertion de données dans les images par l'intermédiaire du LAR iS+P. Le chapitre 8 fera un état de l'art sur la dissimulation de données dans les images. L'adéquation d'une technique particulière d'insertion de données de grande capacité, l'*extension de la différence*, au LAR iS+P sera présentée au chapitre 9. Nous verrons alors la pertinence d'un tel mécanisme, sa mise en œuvre, ses performances et les conséquences sur les performances de compression.

Première partie

Codage d'images fixes



E. P. Jacobs, *La marque jaune*, page 55, case 6.

1

Codeur d'image LAR

Plan

1.1 Principe de codage à deux couches
1.2 Codeur LAR simple
1.3 Approche pyramidale prédictive
1.4 À retenir

La famille des codeurs LAR repose sur le principe suivant : adapter la résolution à l'information présente dans l'image. Le nom LAR pour *Locally Adaptive Resolution* reflète cette philosophie. Cette adaptation peut se faire dans plusieurs directions. D'abord, en résolution spatiale, avec un partitionnement *quadtree* guidé par un critère d'activité locale. Le codeur obtenu est le codeur *flat*-LAR. Ensuite, l'adaptation peut se faire en débit et distorsion, avec l'usage d'une pyramide adaptée pour obtenir la propriété de scalabilité.

La genèse de cette famille de codeurs est la suivante. Il s'agit à l'origine d'un schéma de codage pour effectuer une transmission à très bas débit d'images en niveaux de gris, reconstruites à la réception avec une bonne qualité visuelle. Ce schéma original est complété au fur et à mesure par la gestion de la couleur, l'introduction de régions d'intérêt et des décompositions hiérarchiques. Désormais, le codeur est scalable, tant en distorsion qu'en résolution, avec un continuum entre codage avec et sans pertes. Ce chapitre s'appuie sur la HDR de Déforges (2004) et la thèse de Babel (2005).

1.1 Principe de codage à deux couches

De manière générale, une image I peut s'exprimer comme la superposition de deux images :

1. Une image $\bar{I}$ représentant l'information *globale* de l'image estimée sur un support donné. Cette estimation peut prendre la forme d'une valeur moyenne locale ;
2. Une image $E = I - \bar{I}$ représentant les variations locales autour de $\bar{I}$, c'est-à-dire la texture locale.

Les deux images sont donc telles que :

$$I = \bar{I} + (I - \bar{I}) = \bar{I} + E \tag{1.1}$$

Si l'on veut que l'image *globale* soit une bonne approximation de l'image originale, il est nécessaire de choisir des supports d'estimation adéquats. La famille des codeurs LAR cherche donc à adapter la résolution locale à l'information présente dans l'image. Cette information est évaluée par la mesure d'une activité observée au niveau de blocs de pixels. L'idée est d'obtenir un découpage fin au niveau des contours et un découpage grossier dans les zones uniformes. Ce découpage en bloc respecte donc la sémantique de l'image.

Ainsi, le principe général des codeurs LAR est d'utiliser un codeur réalisé en deux couches :

- *couche spatiale* (ou *flat*-LAR) pour le codage de l'image globale $\bar{I}$;
- *couche spectrale* pour le codage de la texture locale E.

La figure 1.1 illustre ce principe général. Lorsque l'on veut étendre celui-ci aux images couleurs, l'espace de représentation YCrCb est utilisé, avec, pour mémoire :

Y la luminance ;

Cr la chrominance rouge ;

Cb la chrominance bleue.

Cet espace de couleurs permet une décorrélation suffisante de l'information contenue dans les composantes RVB usuelles de l'image. La transformation est également simple et rapide à calculer.

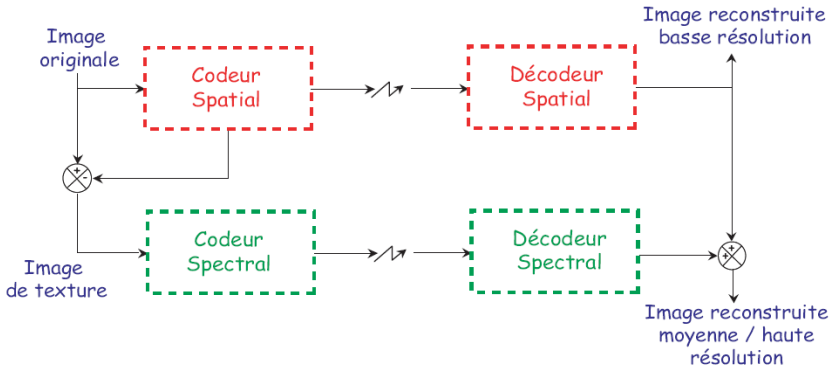


FIG. 1.1: Schéma global LAR à deux couches : codeurs spatial et spectral

Les sections qui suivent présentent le fonctionnement bicouche du codeur de manière plus détaillée. Ensuite, nous verrons comment introduire la scalabilité par l'utilisation d'une construction pyramidale.

1.2 Codeur LAR simple

1.2.1 Codeur spatial *flat*-LAR

Le codeur spatial, ou *flat*-LAR,s'intéresse à fournir l'information globale contenue dans une image. Il permettra d'obtenir un taux de compression élevé, avec une distorsion importante, et en même temps une qualité visuelle meilleure que l'état de l'art. Ce codeur a donc plusieurs fonctions concommitantes :

1. adapter le support de l'information globale $\bar{I}$ afin d'obtenir une image reconstruite de bonne qualité ;
2. coder de manière efficace les valeurs représentant chaque support ;

3. effectuer un post-traitement, non linéaire, pour améliorer la qualité visuelle de l'image reconstruite.

Le *flat*-LAR utilise une représentation de l'image avec un partitionnement à taille de blocs variable. L'adaptation du support se fera à travers l'adéquation d'une partition quadtree de l'image à l'activité locale observée. Ensuite, afin de mettre en œuvre le principe de superposition, chaque bloc considéré sera représenté par sa valeur moyenne. Enfin, un filtrage non linéaire, tenant compte de la taille des blocs, et des positions des contours, permettra d'éliminer les artefacts visuels dus au taux de compression élevé. La figure 1.2 présente un synoptique du *flat*-LAR dont les différentes parties sont précisées dans les sections suivantes.

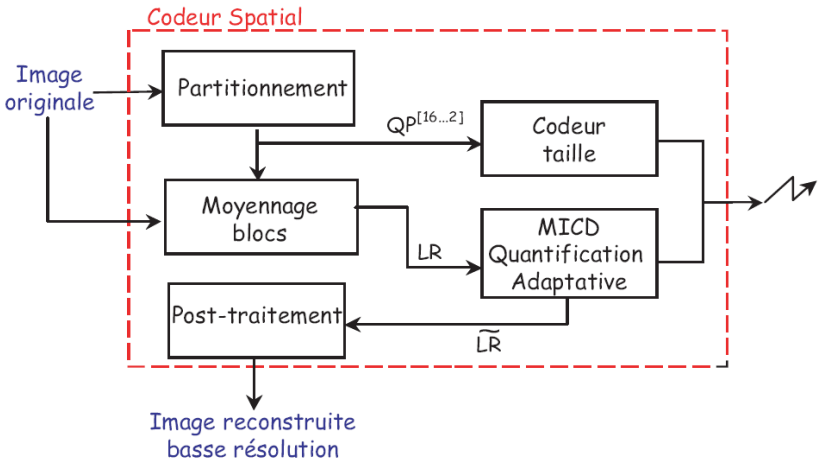


FIG. 1.2: Schéma de principe du codeur spatial

1.2.1.1 Partitionnement quadtree de l'mage

Le partitionnement de l'image en blocs de tailles variables utilise la structure régulière d'un quadtree. En fait, l'image est découpée en bloc de taille $2^N \times 2^N$ et un quadtree est adapté sur chacun de ces blocs. Les quadtrees, introduits par Finkel & Bentley (1974), sont en quelque sorte une extension bidimensionnelle des arbres binaires. Leur utilisation dans le domaine de l'image numérique va de la compression à la représentation. La construction du quadtree se fait usuellement de manière récursive.

À chaque étape, un critère d'activité est ainsi évalué sur le bloc considéré, et, si ce critère dépasse un seuil fixé par ailleurs, le bloc est découpé en 4 et la manœuvre est répétée. La mesure de l'activité consiste en l'évaluation d'un gradient morphologique. Il s'agit de la différence entre les valeurs maximale

et minimale de la luminance du bloc considéré. Le découpage récursif est une description élégante de la construction de la partition quadtree. En fait, une construction de bas en haut, *bottom-up*, ou aggrégative, est utilisée du fait de sa simplicité et de son faible coût de calcul.

En pratique, à la partition quadtree, objet représentatif de l'information globale $\bar{I}$, est associée une image des tailles. Cette image, de même taille que l'image originale, donne la taille du bloc auquel appartient le pixel. Elle permet d'obtenir directement une carte de segmentation, quoique rudimentaire, permettant de distinguer plusieurs classes de blocs :

1. les petits blocs sur les frontières et les zones texturées ;
2. les blocs sur les faibles contours et les zones moyennement texturées ;
3. les grands blocs sur les zones uniformes.

Cette classification permettra de distinguer en particulier les contours des zones uniformes et autorisera la mise en œuvre de techniques de filtrages non linéaires respectant cette dichotomie.

Les images couleurs sont également prises en compte. L'idée première était de retenir en chaque pixel la taille minimale des images des tailles pour l'image Y et les images Cb et Cr. En pratique, la luminance Y est la grandeur la plus souvent choisie pour l'évaluation de l'image des tailles.

Une fois la partition quadtree obtenue, chaque bloc est remplacé par sa valeur moyenne, ou composante continue (DC). Pour les images couleurs, la moyenne de chaque composante est calculée. La section qui suit indique la façon de coder ces valeurs moyennes.

1.2.1.2 Encodage des moyennes par prédiction

Le sous-échantillonnage irrégulier et adaptatif effectué précédemment permet déjà d'obtenir un taux de compression élevé. Les valeurs moyennes des blocs sont ensuite quantifiés de manière uniforme et prédites.

Le pas de quantification dépendra de la taille des blocs. Il est utile de rappeler que le système visuel humain est moins sensible aux variations de luminance et de chrominance dans les zones de contour que dans les zones uniformes, ou, autrement dit, qu'il est plus sensible aux basses fréquences qu'aux hautes fréquences. Ainsi, il est nécessaire de quantifier finement les zones homogènes et plus grossièrement les zones de contours.

Le codage des valeurs moyennes se fait par modulation par impulsions codées différentielles (MICD) en utilisant un prédicteur non linéaire introduit par Graham (1958) et adapté au LAR. Son fonctionnement revient à faire une prédiction linéaire dans les zones uniformes et à orienter une prédiction non

linéaire dans les zones de contours. Les erreurs résultantes seront quantifiées comme vu dans le paragraphe précédent.

La prédiction des valeurs de chrominance profite des corrélations entre celles-ci et la luminance pour adapter le prédicteur et obtenir un gain de l'ordre de 20 % par rapport à un codage effectué directement.

1.2.1.3 Post-traitement

À la reconstruction de l'image codée, des *effets de bloc* peuvent apparaître, principalement sur la composante de luminance. Cet artefact est usuel dans les schémas de compression reposant sur des blocs. Néanmoins, les effets ne sont pas très visibles et ne sont pas comparables à ceux observés sur des images codées par JPEG[1] du fait de l'adaptation de la résolution au contenu de l'image. Ainsi, ces distorsions se remarquent principalement dans les zones homogènes, avec des grands blocs, ou sur les contours, par manque de résolution.

Dans ce contexte, une méthode de lissage se concentrant sur les zones homogènes et conservant les contours est mise en œuvre. Les méthodes directionnelles utilisant des techniques de récupérations de données optimales, dont fait partie la méthode présentée par Muresan & Parks (2001), ont montré leur efficacité.

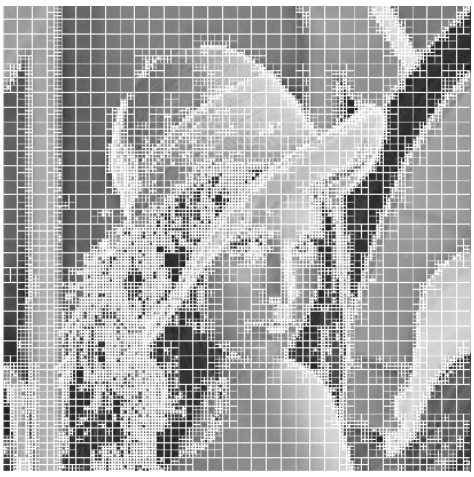
1.2.1.4 Illustration et performance

Le *flat*-LAR est donc un codeur efficace, nécessitant une charge de calcul réduite, propre à le faire fonctionner dans des applications embarquées. Seuls le codage entropique et le post-traitement se révèlent plus complexes. Pour illustrer les performances et la qualité visuelle des images codées par le *flat*-LAR, nous montrons les résultats sur l'image *lena* sur la figure 1.3, figure qui présente également les différentes étapes du codage. On peut observer que les contours sont conservés tandis que les zones homogènes sont lissées. Malgré un taux de compression élevé, les artéfacts sont réduits et peu apparents.

Ainsi, le *flat*-LAR est-il une bonne solution pour faire un codage à très bas débit, fournissant une image codée de qualité acceptable. Cependant, lorsque qu'une distorsion moindre est demandée, le codeur atteint ses limites. Dans ce cas, une approximation de l'image par des blocs constants ne permet pas d'augmenter de manière significative la qualité de l'image reconstruite. De plus, le *flat*-LAR introduit des décalages de contours, invisibles, mais néfastes

[1] Pennebaker & Mitchell (1992) présentent avec quelques détails la structure des codeurs JPEG.

(a) Image source lena, 512x512, 8 bpp



(b) Partition quadtree : 13888 blocs

(c) Image reconstruite basse résolution : 0.2 bpp. Taux de compression : 40, PSNR : 29.2 dB

(d) Image reconstruite après post-traitement sur $QP^{[16..2]}$, PSNR : 29.7 dB

FIG. 1.3: Résultats pour une partition $\mathrm{QP}^{[16..2]}$.

pour le PSNR. Il faut, à ce moment-là, ajouter la texture à l'image *flat*, en utilisant la deuxième couche du codeur, le codeur spectral.

1.2.2 Le codeur spectral

La texture locale E est codée par un codeur par transformée. La transformation utilisée est une transformée en cosinus discret (TCD) à taille de bloc variable, à la façon de JPEG. Ainsi, le codage des coefficients transformés se fait par balayage zigzag et longueur de plages. En outre, une classification inter blocs des coefficients peut aussi être mise en œuvre, améliorant nettement les performances de compression. Ensuite, les coefficients sont quantifiés de manière uniforme avec un pas adapté à la taille du bloc. La figure 1.4 présente un synoptique du codeur spectral.

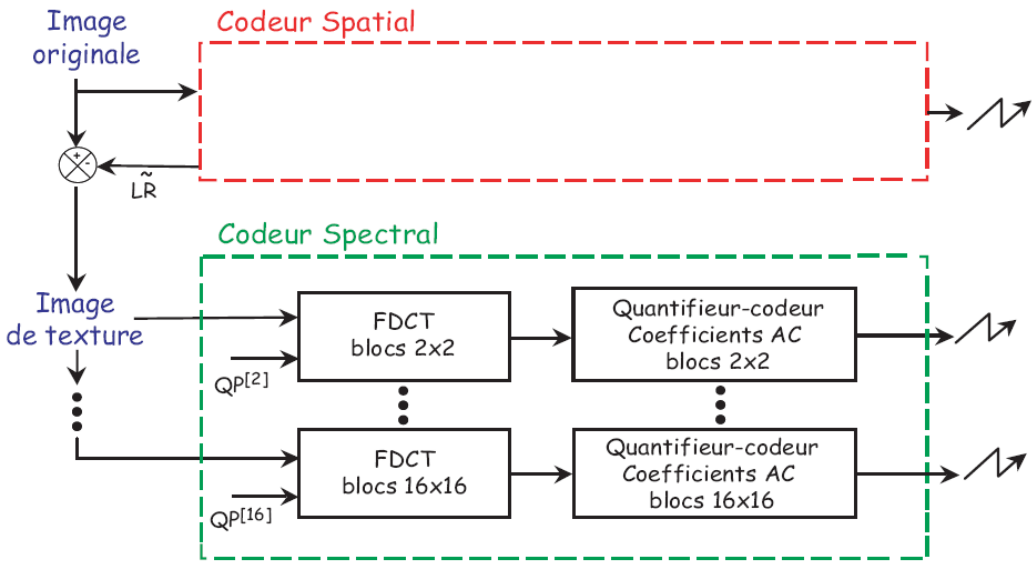


FIG. 1.4: Schéma de principe du codeur spectral pour une partition $QP^{[16..2]}$

Le schéma de la figure 1.4 est progressif. Il permet de transmettre les données suivant la taille des blocs, fournissant autant de flots binaires qu'il y a de tailles de blocs dans la partition quadtree. Par exemple, une transmission suivant l'ordre croissant des tailles de blocs amène à améliorer d'abord les contours, puis les zones moyennement texturées et enfin les zones homogènes. La figure 1.5 décrit l'influence des différents flots sur l'image reconstruite et introduit une notion de scalabilité sémantique. À chaque flot correspond une classe de pixels, ceux des contours ou ceux des zones uniformes. Cette infor-

mation permet de réaliser un codage sémantiquement progressif, autorisant d'améliorer telle ou telle classe de pixels.

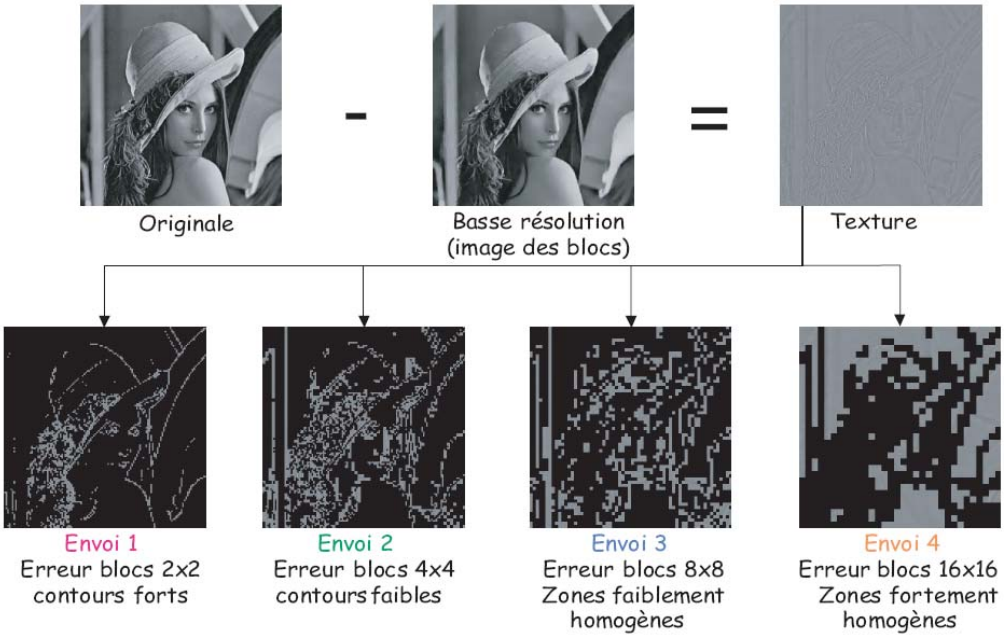


FIG. 1.5: Principe de la progressivité sémantique

1.2.3 Pour la suite

Ici s'achève le codeur LAR simple. Les expérimentations menées par Déforges & Ronsin (2002) montrent que le codeur LAR simple obtient de meilleurs scores que JPEG et JPEG-2000[2] dans des évaluations subjectives de qualité.

Néanmoins, le LAR simple présente l'inconvénient de travailler sur l'image pleine résolution et avec des flots binaires non scalables. Or, les aspects de scalabilité, devenus importants dans l'adéquation entre application et capacité de transmission, ne sont pas pris en compte. Dans les sections suivantes, nous ajoutons la scalabilité en distorsion et en résolution à la scalabilité sémantique via une décomposition pyramidale de l'image.

[2]Taubman & Marcellin (2001) détaillent non seulement la norme mais également nombre d'aspects théoriques.

1.3 Approche pyramidale prédictive

La prise en compte des aspects de scalabilité aboutit au codeur LAR pyramidal prédictif, ou LAR-APP. Cette approche vise deux principaux objectifs :

- fournir une scalabilité par niveaux, en résolution et en qualité ;
- proposer une solution simple et unique de compression permettant indifféremment un codage avec ou sans perte.

L'approche pyramidale prédictive repose sur une pyramide laplacienne de résidus. Le prédicteur utilisé est une adaptation de celui proposé par Wu (1996). Il s'agit d'un prédicteur à contexte enrichi, décrit à la section 1.3.1 Rappelons que les pyramides laplaciennes dans le domaine spatial présentent une redondance importante. Néanmoins, nous verrons que cette redondance pourra être réduite voire supprimée dans le cas de compression avec pertes.

1.3.1 Prédicteur de Wu

La méthode du LAR-APP non scalable repose sur une extension de l'algorithme de Wu. Ce prédicteur nécessite trois passes successives et entrelacées pour encoder l'ensemble des données.

Contrairement à de nombreux algorithmes aux contextes dits à 180 degrés c'est-à-dire restreints à l'espace causal, l'algorithme de Wu permet de modéliser un contexte enrichi. Cela permet d'obtenir une meilleure prédiction et réduit d'autant l'erreur à coder. Les trois passes du prédicteur sont décrites ci-après.

1.3.1.1 Première passe : sous-échantillonnage de l'image

L'algorithme de Wu commence par sous-échantillonner l'image originelle d'un facteur 2 dans les 2 dimensions. La valeur des pixels de l'image sous-échantillonnée est la moyenne des deux pixels de la diagonale des blocs 2×2 de l'image de départ. Cette image est ensuite traitée par une MICD. Les valeurs prédites sont alors calculées ; l'erreur commise sur l'estimation constitue la première information à encoder.

1.3.1.2 Deuxième passe : exploitation du contexte enrichi

La seconde passe a pour but d'encoder la moitié des pixels de l'image, ceux qui sont diagonalement adjacents. Connaissant la moyenne de chaque diagonale, encodée lors de la première passe, il nous suffit de coder l'un des deux pixels de chaque diagonale et nous sommes en mesure de reconstruire l'autre. Suivant les cas de figure (compression avec ou sans perte), certains

termes seront négligés faisant ainsi office de quantification. En effet, si une quantification était effectivement appliquée, la dynamique de ces termes ne leur aurait pas permis de persister.

1.3.1.3 Troisième passe : encodage des pixels restants

L'autre moitié des pixels va maintenant être encodée. Les deux passes précédentes créent un contexte enrichi qui entoure spatialement le pixel à coder. Ainsi, la prédiction des 2 valeurs restantes sera bien meilleure, avec prise en compte non seulement de l'espace causal, mais également d'une estimation des valeurs de pixels dans l'espace anticausal.

1.3.2 Principes généraux de l'approche pyramidale prédictive

Le principe de base du LAR est conservé : deux couches permettent de coder l'information globale pour la première, la texture locale pour la seconde. Le codeur spatial va être remplacé par le LAR-APP. La couche spectrale quant à elle peut soit rester inchangée soit être remplacée par le LAR-APP réalisant une décomposition pyramidale complète.

Cette approche nécessite d'abord la définition d'une méthode de décomposition pyramidale conditionnelle à la partition quadtree. Ensuite, l'opération sera réalisée en deux temps : obtention de l'image de blocs (*flat*) et superposition de la texure locale. La figure 1.6 résume l'approche. La scalabilité est illustrée sur cette figure de la manière suivante : la scalabilité en résolution est obtenue en se déplaçant verticalement dans la *première passe* tandis que la scalabilité en distorsion s'obtient horizontalement entre les deux passes.

1.3.3 Construction de la pyramide

La construction de la pyramide réalise l'adéquation entre le prédicteur de WU et la décomposition pyramidale. Ainsi, chaque pixel des images successives est donné par la moyenne de la première diagonale du bloc 2×2 du niveau précédent. Deux modes de fonctionnements sont à distinguer, suivant que le codage de la texture locale est fait par le LAR-APP ou par le codeur spectral vu à la section 1.2.2.

Dans tous les cas, la première étape de l'algorithme général consiste en l'obtention de l'image *flat*, basse résolution, par la construction de la partition quadtree de l'image originale. La connaissance de ce quadtree reste en effet primordiale, d'une part pour piloter la descente de la pyramide et d'autre

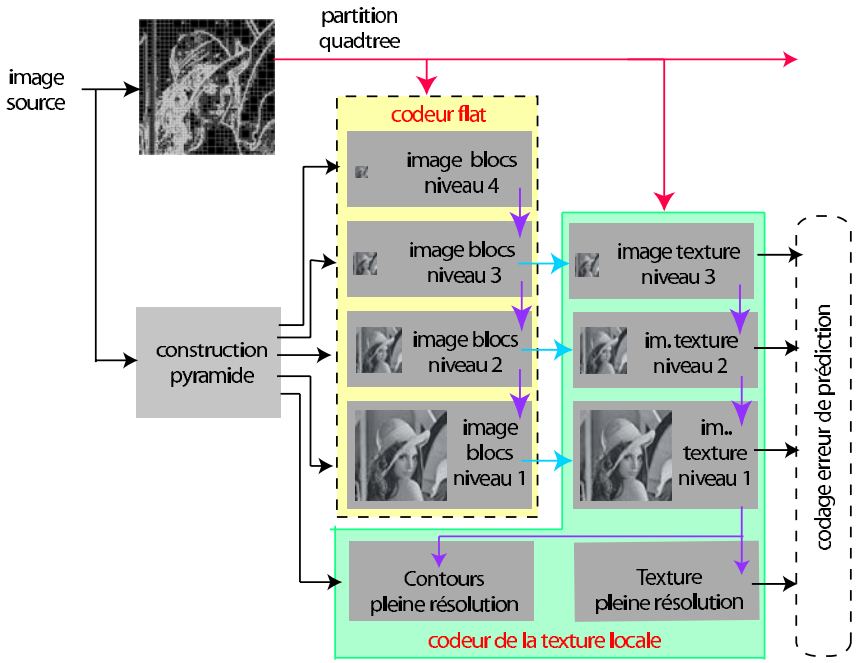


FIG. 1.6: Principe de l'approche pyramidale prédictive

part afin de garantir l'adéquation entre la représentation en blocs issue du *flat*-LAR et celle du schéma hiérarchique.

1.3.3.1 Encodage de la texture par le LAR-APP

Cette solution implique l'utilisation du même processus de descente sur l'ensemble de la pyramide. L'algorithme de construction choisi est de type *bottom-up*. À chaque point d'un niveau l de la pyramide, on fait correspondre un bloc de taille $2^l \times 2^l$ de l'image originale. La valeur de ce bloc correspond donc à la moyenne de la diagonale du bloc étudié dans l'image de départ. La construction pyramidale est illustrée dans la figure 1.7.

L'image des blocs engendrée est donc différente de celle du LAR simple[3], mais en percevoir la différence devient délicat. En effet, la dispersion des valeurs prises par les pixels dans les zones uniformes est faible et plus importante dans les blocs de petite taille. On rappellera en parallèle que l'œil est moins sensible aux variations de luminance à mesure que la taille du bloc considéré diminue. Cette approche engendre donc des différences qui, visuellement, sont pour ainsi dire indétectables.

[3]valeur moyenne calculée sur le bloc *versus* valeur moyenne calculée sur la diagonale du bloc.

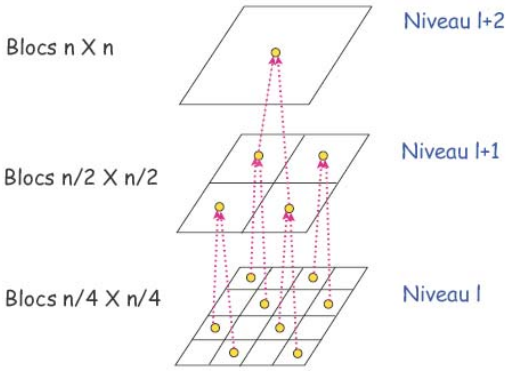


FIG. 1.7: Construction *bottom-up*

1.3.3.2 Encodage de la texture par le codeur spectral

L'encodage de la texture par le codeur spectral implique le respect des conditions posées dans le cas du LAR simple : l'image des blocs est obtenue en calculant la moyenne des pixels de chaque bloc. Sans cela, nous ne pouvons pas appliquer correctement la TCD. On réalise donc une construction incomplète de la pyramide, illustrée par la figure 1.8.

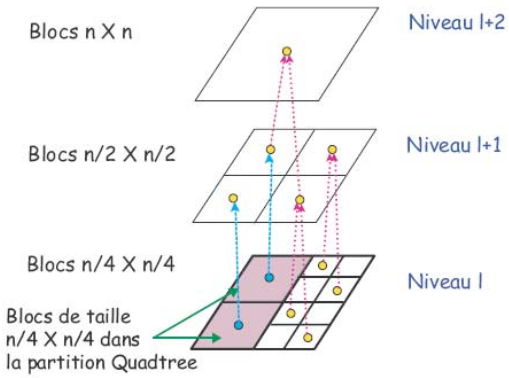


FIG. 1.8: Construction incomplète de la pyramide

1.3.4 Décomposition pyramidale à contexte enrichi

La décomposition pyramidale à contexte enrichi utilise une MICD inter et intra-niveaux. La prédiction intra-niveau fournit l'espace causal tandis que la prédiction inter-niveaux fournit les informations de l'espace anti-causal. La figure 1.9 permet d'illustrer ceci : les prédictions de l'espace anticausal

proviennent des blocs de taille supérieure non encore découpés alors que la prédiction intra-niveaux utilise toutes les valeurs déjà reconstruites dans l'espace causal.

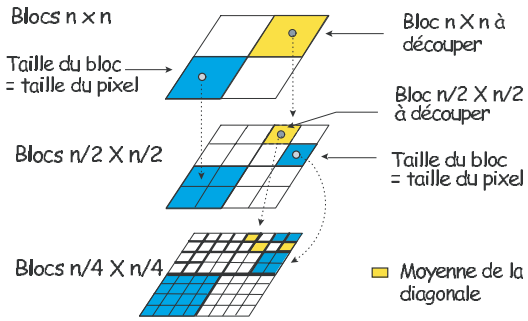


FIG. 1.9: décomposition *top-down* de la pyramide

image	entropie [bpp]			
	LAR-APP sans partition	LAR-APP	CALIC	S+P
baboon	6.30	6.13	6.14	6.11
barb	5.32	5.02	4.93	4.86
barb2	5.35	5.11	4.93	5.01
bike	5.45	4.73	4.53	4.67
cafe	6.26	5.59	5.37	5.56
gold	4.92	4.85	4.65	4.71
hotel	5.32	4.85	4.57	4.73
lena	4.87	4.42	4.33	4.33
peppers	5.09	4.80	4.58	4.64
tools	6.01	5.65	5.53	5.64
us	4.22	3.66	3.60	3.79
zelda	4.80	4.16	3.98	3.96
moyenne	5.32	4.91	4.76	4.83

TAB. 1.1: Entropie [bpp] pour le LAR-APP, CALIC et S+P.

Le tableau 1.1 compare le LAR-APP avec les méthodes de compression sans perte CALIC décrit par Wu & and (1995) et S+P introduit par Said & Pearlman (1993). Les résulats sont bons, même s'ils ne sont pas meilleurs que l'état de l'art.

1.4 À retenir

L'approche pyramidale prédictive donne des résultats très corrects. Toutefois, l'entropie des flots, malgré la diminution de la redondance lors de la décomposition, reste importante. En particulier, une redondance de symboles subsiste. Une manière de diminuer cette redondance est d'utiliser une transformation avant d'effectuer la décomposition pyramidale. Le chapitre 2 présente donc une évolution de l'approche pyramidale, le codeur LAR iS+P.

E. P. Jacobs, *Le secret de l'espadon, tome 2*, page 14, case 6.

2

Codeur d'image LAR Interleaved S+P

Plan

2.1 Principes généraux du LAR IS+P
2.2 Transformée en S et prédicteur de Wu
2.3 Jeu de prédicteurs adaptés
2.4 Prédiction de la texture locale
2.5 Performances en compression sans perte
2.6 À retenir

La section 1.3 a montré que le LAR-APP est une méthode intéressante de codage sans perte des images fixes, avec une scalabilité par niveau de résolution et par niveau de détails. Cependant, les résultats d'entropie brute obtenus restent en deçà de l'état de l'art. La cause principale en est la redondance inhérente à l'emploi d'une pyramide dans le domaine spatial.

L'objet de ce chapitre est de présenter une dérivation du LAR-APP introduisant l'utilisation d'une transformation en S et la construction d'une pyramide dans le domaine ainsi transformé. Cette approche originale repose donc sur une pyramide entrelacée, une transformation en S et le prédicteur P associé. Ainsi, le LAR *Interleaved S+P*, ou LAR iS+P obtiendra des résultats meilleurs que le LAR-APP et que l'état de l'art, comme le montrent Babel et al. (2005a,b). La structure particulière de la méthode permet de considérer la pyramide globale comme étant l'entrelacement de deux pyramides plus simples. La prédiction utilise la méthode S+P.

Après avoir présenté les principes généraux mis en œuvre dans le LAR iS+P, nous verrons le jeu de prédicteurs implantés dans le schéma. Enfin, les résultats probants obtenus permettront de justifier l'emploi du LAR iS+P comme base de travail pour l'ajout de services dans la suite de ce document. Ceci justifie également de rentrer davantage dans les détails de cette variante du codeur LAR au contraire des codeurs LAR vus jusqu'à présent.

2.1 Principes généraux du LAR iS+P

Une analyse fine des performances du LAR-APP montre que la passe 3, qui code les pixels de la seconde diagonale, concerne deux fois plus de blocs que la passe 2. Il en découle un coût de codage des résidus très important, particulièrement pour les blocs codant l'image *flat*-LAR de détails. La transformée en S et l'entrelacement vont permettre de pallier ce déficit de performances, en levant la contrainte de travail dans le domaine spatial. La structure de l'algorithme est cependant conservée, à savoir, d'abord la construction de la pyramide identique, ensuite une descente en deux passes et enfin, différenciation de l'information suivant la nature des blocs.

L'idée principale est d'appliquer une transformée en S unidimensionnelle sur chacune des deux diagonales d'un bloc 2×2 donné et issu de la décomposition d'un bloc appartenant à un niveau supérieur de la pyramide. La figure 2.1 montre à ce propos l'application particulière de la transformée en S sur une image pleine résolution. La transformation en S d'une paire de

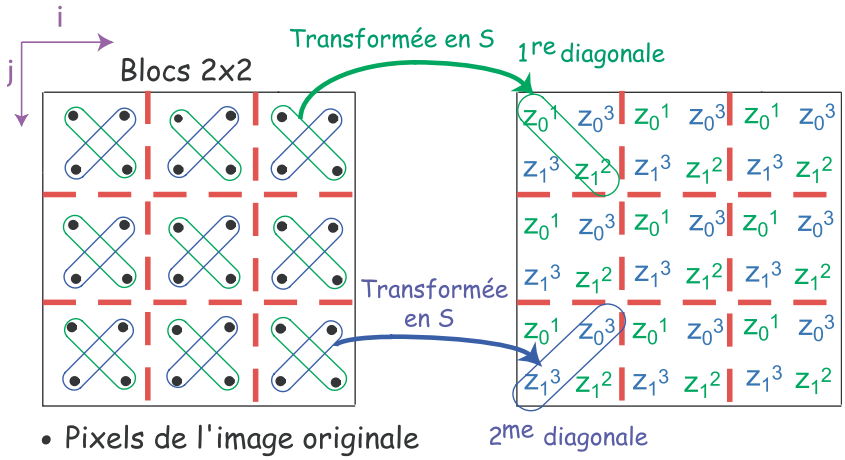


FIG. 2.1: Application particulière de la transformée en S sur une image pleine résolution.

valeurs (u_0, u_1) s'écrit :

$$\begin{cases} z_0 &= \lfloor (u_0 + u_1)/2 \rfloor \\ z_1 &= u_1 - u_0 \end{cases} \tag{2.1}$$

Ainsi, les coefficients transformés obtenus correspondent respectivement à la moyenne arrondie à l'entier inférieur des deux valeurs, et au gradient entre ces deux mêmes valeurs. Les coefficients transformés z_i^k $(i \in \{0,1\}, k \in \{1,2,3\})$ sont encodés par la k$^{\text{e}}$ passe du prédicteur.

Considérons maintenant un bloc 2×2 d'un niveau donnée de la pyramide. Par construction de la pyramide, la valeur moyenne de la première diagonale est déjà connue au niveau immédiatement supérieur. Ainsi, la reconstruction des deux pixels ne nécessite-t'elle que l'obtention, par prédiction, de la valeur du gradient. Par conséquent, contrairement au LAR-APP, il n'y a plus redondance de symboles. La seule redondance présente et inévitable est contenue dans z_1^2, le gradient obtenu dans la passe 2. Il faut donc que le prédicteur associé soit suffisamment efficace pour diminuer la dynamique des résidus. Les coefficients transformés de la seconde diagonale du bloc 2×2 sont également prédits.

L'originalité du LAR IS+P réside dans la possibilité de décrire la construction de la pyramide globale au moyen de deux pyramide entrelacées, comme le montre la figure 2.2. La première pyramide en S est construite comme celle du LAR-APP : la moyenne de la première diagonale correspond à la valeur du pixel correspondant au niveau immédiatement supérieur. La transformée de la seconde diagonale d'un bloc 2×2 peut également se voir comme faisant

partie d'une seconde pyramide en S : l'estimation d'un pixel dépend alors des pixels présents au niveau immédiatement inférieur de la première pyramide en S.

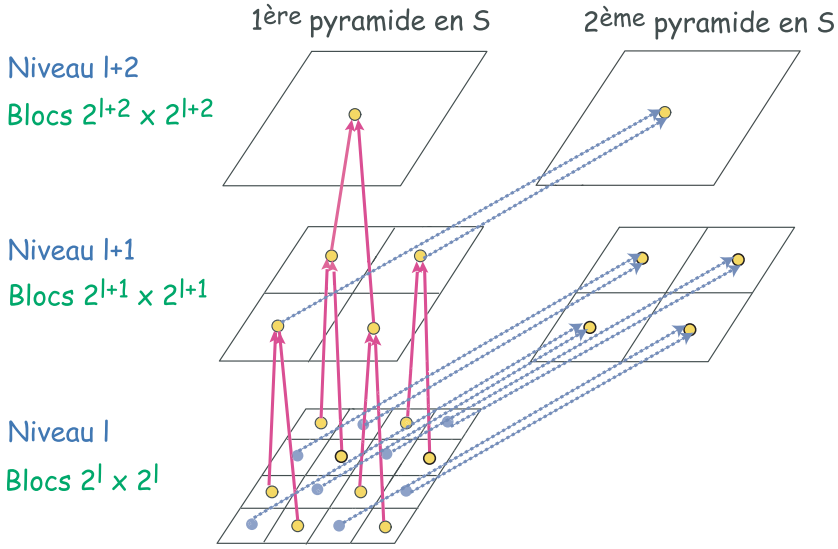


FIG. 2.2: Construction des deux pyramides en S entrelacées.

Il apparaît donc que l'efficacité du prédicteur du gradient des diagonales sera un point clef pour l'efficacité globale du codage. La section suivante précise donc l'association de la transformation et de la prédiction en reprenant le modèle S+P.

2.2 Transformée en S et prédicteur de WU

Cette section présente les notions utiles à la compréhension des prédicteurs mis en œuvre dans le LAR iS+P. L'idée est de différencier les prédicteurs suivants certains critères :

nature du coefficient transformé gradient ou moyenne ;

pyramide en S concernée passe courante ou appartenance du pixel à la passe 2 ou 3 du prédicteur de WU ;

nature de l'information image *flat*-LAR ou texture.

Cette classification permet d'obtenir un jeu de prédicteurs localement adaptés aux pixels traités. Les prédicteurs présentés ultérieurement sont ceux qui fourniront une entropie des résidus la plus faible.

2.2.1 Éléments de notations

Cette section précise les notations relatives aux prédicteurs du reste de ce chapitre. La reconstruction d'un bloc 2×2 $Y_l(b^2(i,j))$ du niveau l de la pyramide s'effectue en deux passes, au travers des deux pyramides entrelacées.

La première phase utilise la première pyramide en S et traite des pixels $Y_l(2i, 2j)$ et $Y_l(2i+1, 2j+1)$ de la première diagonale. La transformée en S appliquée à ce couple de points s'écrit :

$$z_0^{l,1}(2i, 2j) = \left\lfloor \frac{Y_l(2i, 2j) + Y_l(2i+1, 2j+1)}{2} \right\rfloor \tag{2.2}$$

$$z_1^{l,2}(2i+1, 2j+1) = Y_l(2i, 2j) - Y_l(2i+1, 2j+1) \tag{2.3}$$

De manière similaire, les coefficients transformés de la seconde pyramide en S sont calculés par :

$$z_0^{l,3}(2i+1, 2j) = \left\lfloor \frac{Y_l(2i, 2j+1) + Y_l(2i+1, 2j)}{2} \right\rfloor, \tag{2.4}$$

$$z_1^{l,3}(2i, 2j+1) = Y_l(2i+1, 2j) - Y_l(2i, 2j+1), \tag{2.5}$$

où les indices i et j permettent de balayer respectivement les colonnes et les lignes de l'image.

2.2.2 Moyenne de la première diagonale

Lors de la passe 2 du prédicteur de WU, la valeur moyenne de la première diagonale $z_0^{l,1}(2i, 2j)$ est donnée par la valeur du bloc au niveau supérieur dont elle dépend directement, de la manière suivante :

$$z_0^{l,1}(2i, 2j) = Y_{l+1}(i, j). \tag{2.6}$$

Pour un pixel placé au sommet de la pyramide, *ie.* lorsque $l = l_{max}$, la moyenne est utilisée :

$$Y_{l_{max}}(i, j) = z_0^{l_{max}-1,1}(2i, 2j) \tag{2.7}$$

La prédiction pour estimer les valeurs de ces pixels est alors réalisée comme pour le LAR-APP. La section 2.3 va définir un jeu de prédicteurs efficaces à l'encodage de l'information issue de l'image des blocs.

2.3 Jeu de prédicteurs adaptés

Sur le modèle du LAR-APP, la reconstruction de l'image *flat*-LAR induit, pour un niveau de la pyramide l donné, le traitement des seuls pixels de taille $Siz(x \times l, y \times l) \leq 2^l$. La distinction sera faite naturellement entre la prédiction de la moyenne et celle du gradient, et entre la prédiction pour la première pyramide en S de celle utilisée pour la seconde. Les prédicteurs qui sont présentés par la suite ont été déterminés de manière empirique.

2.3.1 Prédicteurs pour la première pyramide en S – image des blocs

La valeur du coefficient $z_0^{l,1}$ étant déjà connue au niveau supérieur de la première pyramide en S, il suffit d'estimer $z_1^{l,2}$, le gradient de la première diagonale d'un bloc 2×2. Cette estimation est faite par un prédicteur linéaire du même type que le prédicteur de WU :

$$\begin{aligned} \tilde{z}_1^{l,2}(2i+1, 2j+1) = 2.1 \Big[& 0.9 Y_{l+1}(i,j) \\ & + \frac{Y_l(2i+1, 2j-1) + Y_l(2i-1, 2j-1) + Y_l(2i-1, 2j+1)}{6} \\ & - 0.05 \Big(Y_l(2i, 2j-2) + Y_l(2i-2, 2j) \Big) \\ & - 0.15 \Big(Y_{l+1}(i, j+1) + Y_{l+1}(i+1, j) \Big) \\ & - Y_{l+1}(i,j) \Big) \Big]. \end{aligned} \tag{2.8}$$

2.3.2 Prédicteurs pour la seconde pyramide en S – image des blocs

La passe 3 de l'algorithme de WU adapté à notre problématique encode les coefficients transformés issus de la seconde diagonale d'un bloc 2×2, appartenant de fait à la seconde pyramide en S.

2.3.2.1 Moyenne de la seconde diagonale

Les prédicteurs des valeurs $z_0^{l,3}$ codant pour l'image *flat*-LAR ou la texture locale ont la même forme ; seuls des coefficients multiplicateurs diffèrent. L'estimation des coefficients $z_0^{l,3}$ tire parti des information inter- et intra-

niveaux, par la relation :

$$\breve{z}_0^{l,3}(2i+1,2j) = \alpha_0 \frac{Y_l(2i-1,2j+1) + Y_l(2i,2j+2) + Y_l(2i+2,2j) + Y_l(2i+1,2j-1)}{4} + \beta_0 \widehat{z}_0^{l,1}(2i,2j), \tag{2.9}$$

avec $\widehat{z}_0^{l,1}(2i,2j)$ la valeur reconstruite, après prédiction, du coefficient $z_0^{l,1}(2i,2j)$. Notons qu'une compression sans perte est obtenue en prenant $\widehat{z}_0^{l,1}(2i,2j) = z_0^{l,1}(2i,2j)$. L'obtention de l'image *flat*-LAR se fait en utilisant le couple de valeurs $(\alpha_0, \beta_0) = \left(\frac{1}{4}, \frac{3}{4}\right)$.

2.3.2.2 Gradient de la seconde pyramide en S

La meilleure prédiction des coefficients $z_1^{l,3}$ est obtenue par le prédicteur :

$$\begin{aligned}\breve{z}_1^{l,3}(2i,2j+1) =& \alpha_1 \left(Y_l(2i-1,2j+1) + Y_l(2i,2j+2) - Y_l(2i+1,2j-1) - Y_l(2i+2,2j)\right) \\ & - \beta_1 \left(Y_l(2i-1,2j) + Y_l(2i-1,2j+2) - Y_l(2i,2j-1) - \breve{Y}_l(2i,2j+1)\right)\end{aligned} \tag{2.10}$$

où $(\alpha_1, \beta_1) = \left(\frac{3}{8}, \frac{1}{8}\right)$. $\breve{Y}_l(2i,2j+1)$ est la valeur du pixel estimée $Y_l(2i,2j+1)$ dans la passe 3 du prédicteur de WU.

2.4 Prédiction de la texture locale

La deuxième descente de la pyramide permet la reconstruction de la texture. À chaque niveau de la pyramide, seuls les pixels de taille $Siz(x \times l, y \times l) > 2^l$ sont prédits, les autres ayant été traités lors de la première descente. À nouveau, la distinction sera faite entre les valeurs issues de la première pyramide en S (première diagonale), de celles de la deuxième pyramide en S (seconde diagonale).

2.4.1 Première pyramide en S – texture

L'information de texture se caractérise par une faible activité locale. En particulier, les valeurs de gradient $z_1^{l,2}$ sont difficilement appréciables à travers un processus linéaire. C'est pourquoi nous nous appuyons sur des opérateurs non linéaires de type médian.

2.4.1.1 Gradient de la première pyramide en S

Soit $m_e(u_1, u_2, \ldots, u_n)$ la valeur médiane d'un ensemble $(u_1, u_2, \ldots, u_n)$ de n valeurs. La valeur estimée du coefficient $z_1^{l,2}(2i, 2j)$ situé dans une zone de texture est donnée par :

$$\tilde{z}_1^{l,2}(2i+1, 2j+1) = \frac{1}{4}\Big(m_e\big(Y_l(2i-2, 2j), Y_l(2i, 2j-2), Y_l(2i-1, 2j-1)\big) + m_e\big(Y_{l+1}(i+1, j), Y_{l+1}(i, j+1), Y_{l+1}(i+1, j+1)\big)\Big). \tag{2.11}$$

2.4.1.2 Contrôle de l'erreur de prédiction de z_1^l – texture

La construction de la partition quadtree implique que les blocs de taille 4×4 à $N_{max} \times N_{max}$ sont constitués de pixels dont la valeur est dans un intervalle d'amplitude T_h, le seuil de découpage. Ainsi, la valeur prédite de gradient d'un bloc de zone homogène ne peut dépasser, en valeur absolue, T_h. Ceci vaut aussi bien pour la passe 2 que pour la passe 3 de notre décomposition. De ce fait, un contrôle est effectué après estimation de z_1^l afin de borner l'erreur commise.

2.4.2 Deuxième pyramide en S – texture

Le contexte intra-niveau très riche disponible pour l'estimation de $z_0^{l,3}$ et $z_1^{l,3}$ suffit à une prédiction linéaire simple. Ainsi, les moyennes $z_0^{l,3}$ sont obtenues en utilisant la relation 2.9 avec $(\alpha_0, \beta_0) = (0.37, 0.63)$. La prédiction du gradient de $z_1^{l,3}$ est obtenue par l'équation 2.10, avec le couple de valeurs $(\alpha_1, \beta_1) = \big(\frac{1}{4}, 0\big)$.

2.5 Performances en compression sans perte

Nous présentons dans cette section les performances du LAR iS+P pour des images naturelles mises en regard d'autres codeurs sans pertes constituant l'état de l'art. De même, l'influence des différents prédicteurs sera analysée.

Le tableau 2.1 propose une comparaison des résultats obtenus par application de LAR iS+P, de CALIC, et de S+P pour un ensemble consistant d'images naturelles. Il apparaît un gain significatif apporté par la combinaison de l'approche pyramidale prédictive et de la transformée en S. En outre, le LAR iS+P est bien meilleur que le S+P et meilleur en moyenne que CALIC. De plus, le LAR iS+P est scalable, ce qui n'est pas le cas de CALIC.

images	entropie brute [bpp]			
	LAR iS+P sans partition	LAR iS+P	CALIC	S+P
Barb2	5.21	4.96	4.93	5.01
Cafe	5.66	5.43	5.37	5.56
Gold	4.82	4.64	4.65	4.71
Hotel	4.91	4.66	4.57	4.73
Lena	4.46	4.30	4.33	4.33
Peppers	4.74	4.57	4.58	4.64
tools	5.78	5.52	5.53	5.64
us	4.01	3.55	3.60	3.78
moyenne	4.95	4.70	4.70	4.80

TAB. 2.1: Entropies comparées des flots binaires de compression sans pertes obtenues par LAR iS+P, CALIC et S+P

Le tableau 2.1 fait également apparaître l'importance de la partition quadtree dans l'efficacité de compression. En effet, ce partitionnement permet de tirer profit d'une classification explicite qui bénéficie aux différents prédicteurs utilisés.

En nous intéressant à une image particulière, *lena*, nous pouvons analyser de manière plus fine les données entropiques obtenues. Le tableau 2.2 indique l'entropie des différents éléments codant l'image, tandis que le tableau 2.3 montre le nombre de blocs concernés à chaque niveau de la décomposition, suivant la passe du prédicteur et le coefficient transformé à coder. Cer dernier tableau montre en particulier que le nombre de blocs concernés par la texture, plus faible que le nombre de blocs du *flat*-LAR pour les trois premiers niveaux de la pyramide, devient nettement plus grand pour les deux derniers niveaux.

Le gain constaté est dû à une meilleure prédiction des coefficients $z_0^{l,3}$ et $z_1^{l,3}$ issus de la passe 3 d'une part, et des coefficients $z_1^{l,2}$ au niveau de la texture d'autre part. De plus, le regroupement des données compressées suivant leur nature (image des blocs/texture, passe 2/passe 3, coefficients z_0^l/z_1^l) et par niveau de décomposition, apporte une fois encore une diminution notable de l'entropie globale brute. Toutefois, les résultats obtenus sur la prédiction des valeurs de gradient de la première diagonale ne sont pas totalement satisfaisants pour les blocs codant l'image *flat*-LAR. Il reste là encore des pistes à explorer.

niveau	taille des blocs	entropie [bpp]		
4	tous blocs	Passe 1 : 6.06		
		$z_1^{l,2}$	$z_0^{l,3}$	$z_1^{l,3}$
3	texture 16×16	3.46	2.97	3.13
	LAR : 8×8, taille4 2×2	6.37	5.10	5.97
2	texture : 16×16, 8×8	3.83	2.80	3.43
	LAR : 4×4, 2×2	6.17	4.98	5.84
1	texture : 16×16, 8×8, 4×4	3.97	3.08	3.79
	LAR : 2×2	6.14	4.89	5.78
0	texture : 16×16, 8×8, 4×4	4.02	3.10	3.92
	Contours : 2×2	5.56	4.23	5.13

TAB. 2.2: Entropie de l'image *lena* codée sans perte (le seuil valant 20) par niveau de la pyramide, par numéro de passe, par coefficient et par taille. Entropie totale 4.31 bpp.

niveau	taille des blocs	nombre de blocs
4	tous blocs	Passe 1 : 1024
3	texture : 16×16	148
	LAR : 8×8, 4×4, 2×2	876
2	texture : 16×16, 8×8	1464
	LAR : 4×4, 2×2	2632
1	texture : 16×16, 8×8, 4×4	10022
	LAR : 2×2	6362
0	texture : 16×16, 8×8, 4×4	40088
	contours : 2×2	25448

TAB. 2.3: Nombre de blocs concernés pour chaque niveau, selon la passe et la nature du coefficient pour l'image *lena*

2.6 À retenir

Le codeur LAR iS+P associe de manière originale une ondelette simple et une décomposition spatiale pyramidale. Le codeur en sous-bande obtenu est extrêmement efficace tant pour l'entropie que pour la qualité visuelle des images intermédiaires. Ses performances sont meilleures que celles de l'état de l'art pour la compression sans perte, avec de qualités supplémentaires. L'absence de redondance dans la pyramide, une scalabilité en résolution et en qualité, une image *flat*-LAR identique au LAR simple, font du LAR iS+P la bête de somme[1] qui servira pour la suite de notre étude.

Il est utile de répéter que le bon comportement du codeur repose beaucoup sur la partition quadtree qui pilote les différentes phases de la prédiction. Le chapitre suivant va se consacrer plus en détails sur cette partition, en particulier, sur l'influence de celle-ci sur les flots décodés.

[1] ou le cheval de trait ?

E. P. JACOBS, *Le mystère de la grande pyramide, tome 1*, page 22, case 12.

3

Influence de la partition quadtree

Plan

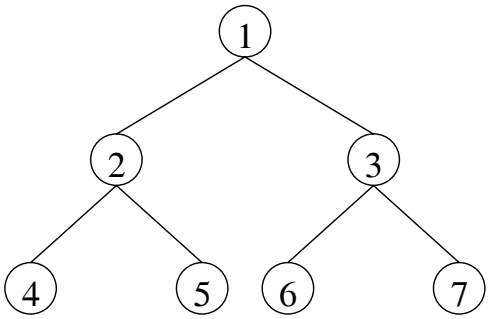
3.1 Codage de la partition quadtree
3.2 Structure du flot binaire LAR
3.3 Influence des erreurs du flot binaire
3.4 À retenir

Les chapitres 1 et 2 montrent que les performances obtenues par le codeur LAR iS+P dépendent, entre autres, du partitionnement quadtree de l'image. La partition obtenue s'adapte au contenu sémantique de l'image et permet une classification implicite des pixels et des coefficients transformés correspondants.

L'objectif de ce chapitre est de préciser l'influence de la partition quadtree, non plus du point de vue de l'efficacité de compression, mais de la résilience aux erreurs. La section 3.1 indique la manière de coder de manière efficace la partition quadtree. La section suivante précise la structure du flot binaire LAR iS+P. Cette description permettra de justifier des distorsions observées liées aux erreurs de transmission commises sur la partition. Enfin, la section 3.3, après une analyse théorique, présente l'impact visuel d'une erreur binaire sur un des flots codant la partition.

3.1 Codage de la partition quadtree

La compréhension de l'influence de la partition quadtree passe par une réflexion sur la façon de la coder. Les travaux s'intéressant au codage des partitions quadtree sont nombreux. Ils s'appuient principalement sur la structure régulière d'arbre propre à cette représentation. L'essentiel réside dans le parcours de l'arbre quaternaire correspondant à la partition quadtree. Les techniques de parcours des quadtrees sont des adaptations des parcours des arbres binaires. Il existe 5 types de parcours : préfixe, infixe, postfixe, en profondeur et en largeur. La figure 3.1 montre les parcours que l'on obtient avec un arbre binaire complet de profondeur 2.



parcours :

préfixe : 1 2 4 5 3 6 7
postfixe : 4 5 2 6 7 3 1
infixe : 4 2 5 1 6 3 7
en largeur : 1 2 3 4 5 6 7
en profondeur : 1 2 4 5 3 6 7

FIG. 3.1: arbre binaire et ses parcours

Dans le parcours *préfixe*, chaque nœud est visité ainsi que chacun de ses fils. Pour le parcours *postfixe*, chaque nœud est visité après chacun de ses fils. Le parcours *infixe* visite chaque nœud entre les nœuds de son sous-

arbre gauche et les nœuds de son sous-arbre droit. Les valeurs sont données dans l'ordre croissant. Le parcours en *profondeur* cherche à visiter d'abord les nœuds les plus éloignés de la racine, dont les nœuds parents ont déjà été visités. Enfin, le parcours en *largeur* visite toujours les nœuds les plus proches de la racine. Le lecteur intéressé trouvera davantage de détails dans le livre de Cormen et al. (2002)[1].

La technique que nous avons retenue s'inspire de celle présentée par Nicolas (1992) dans le cadre de l'estimation de mouvement par appariement de blocs. Il s'agit d'un parcours de type *en largeur* d'encodage progressif de la partition quadtree. L'algorithme procède comme suit : en partant des blocs de taille maximale, indiquer par un 0 si le bloc n'est pas à découper, et par un 1 si le bloc est subdivisé en 4. Cette passe est itérée pour les blocs de taille décroissante, en ne s'intéressant qu'aux blocs éventuellement à subdiviser à nouveau. La figure 3.2 illustre ce propos pour le codage d'une partition quadtree $QP^{[16..2]}$. Pour résumer, l'algorithme effectue un codage binaire itératif conditionnel inter-niveaux.

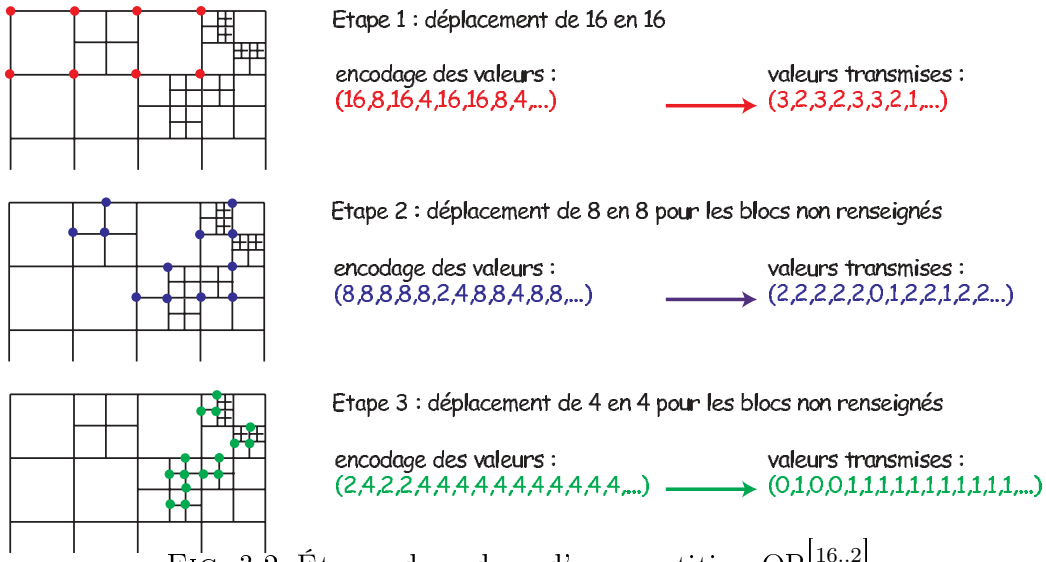


FIG. 3.2: Étapes de codage d'une partition $QP^{[16..2]}$.

3.2 Structure du flot binaire LAR

Le flot binaire LAR est scindé en deux. D'une part la partie permettant de reconstituer l'image plate au niveau de résolution souhaité (passe 1) et d'autre

[1] Une *Introduction à l'algorithmique* de plus de 1000 pages.

part, la partie ajoutant la texture à l'image plate (passe 2). La figure 3.3 présente le détail du flot. Dans chaque partie, le flot est décomposé en sous-flots, un par niveau. Chaque sous-flot de la passe 1 comprend le flot de la partition LAR comment décrit en 3.1 et le flot de construction de l'image plate.

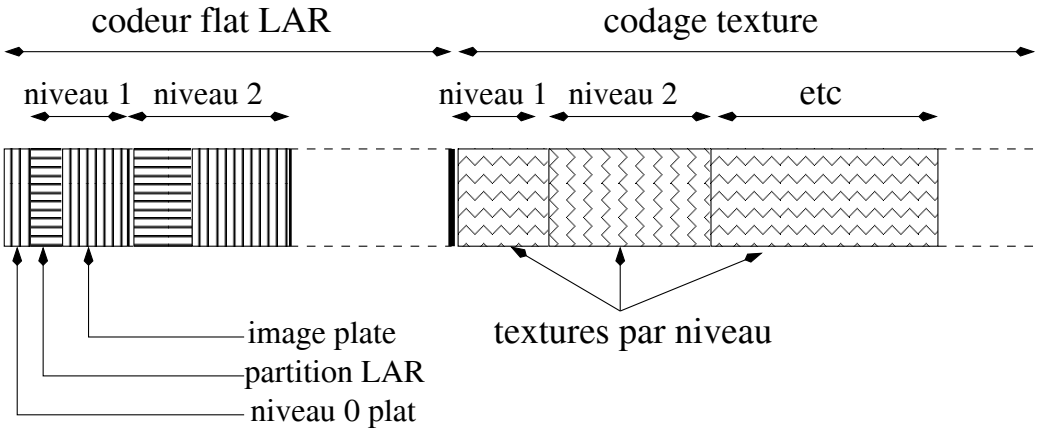


FIG. 3.3: Structure du flot binaire LAR

3.3 Influence des erreurs du flot binaire

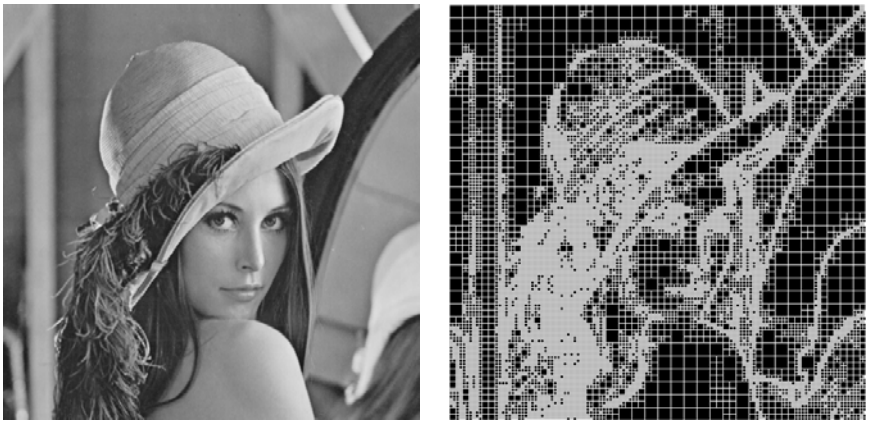
Pour évaluer la résilience du codeur LAR IS+P et comprendre l'importance d'une valeur transmise par rapport aux autres, l'impact visuel de différentes erreurs a été testé. Cette section examine deux types d'erreurs :

- une erreur dans la partition ;
- une erreur sur un résidu.

Nous montrerons les résultats pour l'image *lena* pour laquelle la figure 3.4 indique la partition quadtree $QP^{[16..2]}$ associée. Une telle partition nécessite donc trois passes d'encodage qui génèrent alors trois sous-flots binaires.

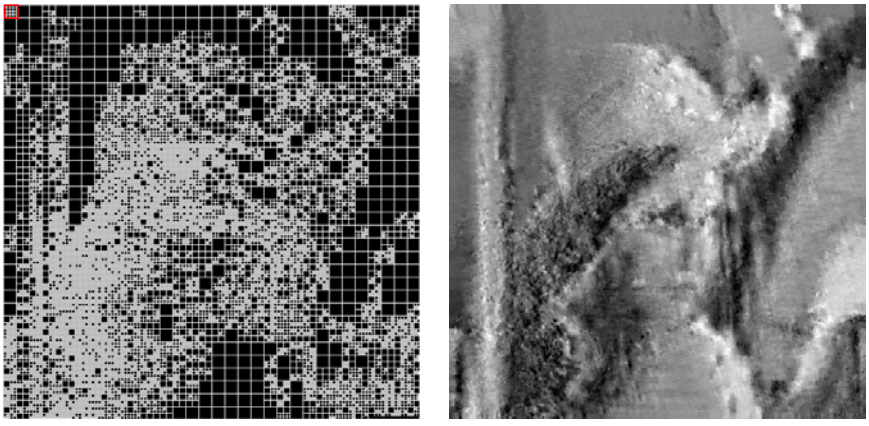
3.3.1 Erreur dans la partition

Pour évaluer l'impact d'une erreur commise sur la partition, une seule valeur est modifiée au décodeur à la réception du flot. Une première expérience consiste à modifier une valeur dans le premier sous-flot binaire, celui qui indique si les blocs de taille 16×16 doivent être découpés ou non. La figure 3.5 montre l'effet du remplacement d'un bloc 16×16 par un bloc 2×2, c'est-à-dire l'effet du découpage d'un bloc uniforme.



(a) image originale (b) partition associée

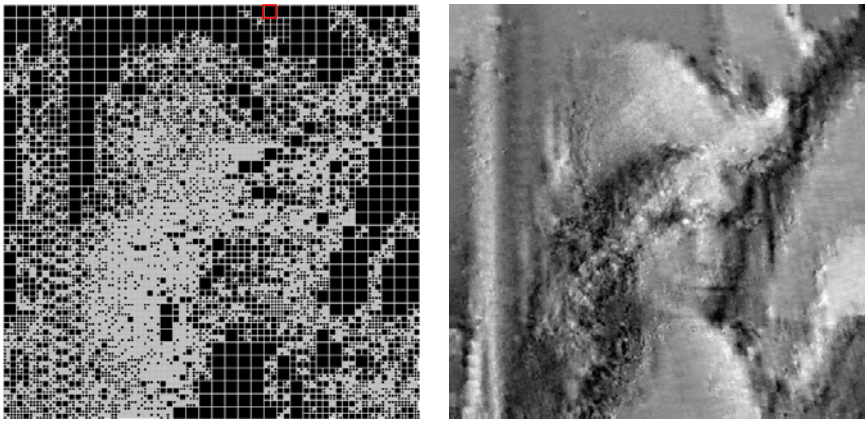
FIG. 3.4: image *lena* et la partition quadtree $\mathrm{QP}^{[16..2]}$ associée



(a) partition modifiée (b) image décodée, PSNR = 5.22 dB

FIG. 3.5: Première étape du codage de la partition : remplacement d'un bloc 16×16 par un bloc 2×2 dans le coin en haut à gauche.

La figure 3.6 montre, au contraire, l'effet produit par le remplacement d'un bloc de taille plus petite par un bloc 16×16 et illustre ce qui se passe lorsque qu'un bloc à forte activité est pris pour uniforme du fait de l'erreur commise.



(a) partition modifiée (b) image décodée, PSNR = 18.47 dB

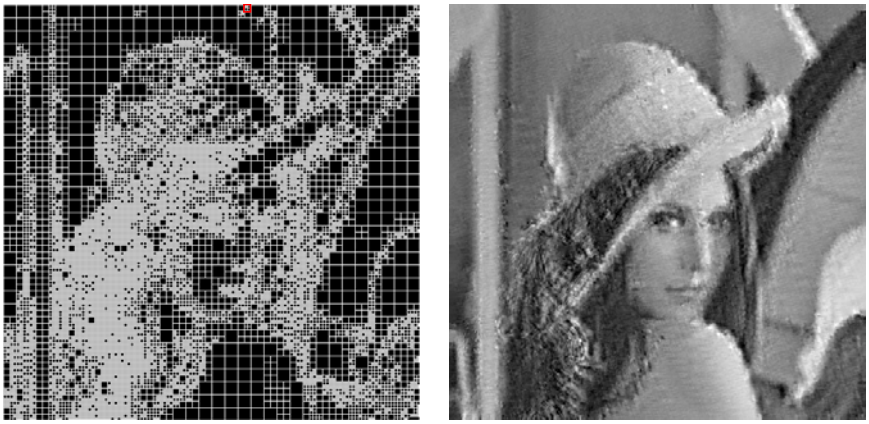
FIG. 3.6: Première étape du codage de la partition : remplacement d'un bloc 2×2, 4×4 ou 8×8 par un bloc 16×16 aux deux tiers de la première ligne de blocs.

La figure 3.7 montre quand à elle l'effet d'une erreur dans le deuxième sous-flot binaire, correspondant au découpage éventuel des blocs de taille 8×8.

Enfin, la figure 3.8 s'intéresse aux effets d'une erreur sur le dernier sous-flot binaire codant le découpage conditionnel des blocs de taille 4×4.

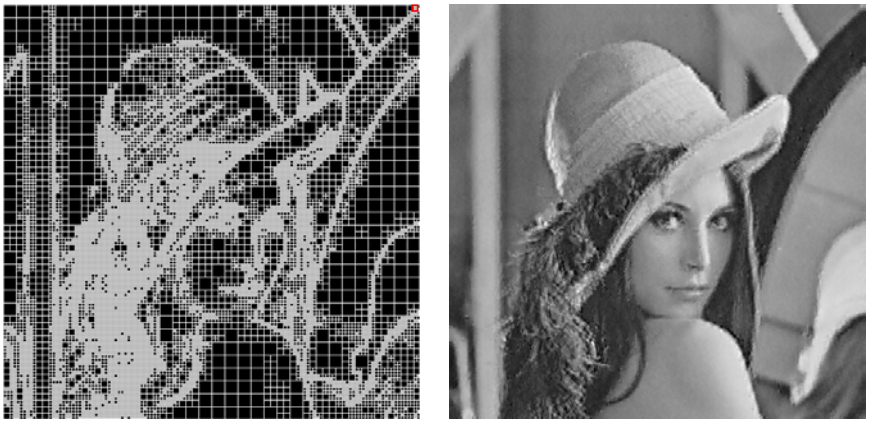
L'analyse de la qualité visuelle des images résultantes après transmission erronée d'un seul bit de codage de la partition quadtree montre qu'une erreur sur le premier sous-flot provoque les dégradations les plus importantes. En effet, une valeur erronée modifie la représentation non uniforme de l'image. Ainsi, si le premier bloc, en haut à gauche, est au départ de taille 16×16 et qu'une erreur le contraint à être découpé en bloc 8×8, cette contrainte se répercutera sur l'ensemble des blocs concernés par les deuxième et troisième étapes. Cela explique également pourquoi l'impact visuel de l'erreur diminue lorsque celle-ci se produit dans les sous-flots binaires pour les blocs plus petits. On remarque également que les effets se font voir dans la partie *anti-causale* de l'image, et la visibilité de l'erreur dépendra donc également de sa position dans l'image.

Ce que l'on peut retenir de cette expérimentation est que l'information codant la partition quadtree est **nécessaire** pour reconstruire correctement



(a) partition modifiée (b) image décodée, PSNR = 21.60 dB

FIG. 3.7: Deuxième étape du codage de la partition : remplacement d'un bloc 8×8 par un bloc 2×2 situé aux deux tiers de la première ligne de blocs.



(a) partition modifiée (b) image décodée, PSNR = 25.52 dB

FIG. 3.8: Troisième étape du codage de la partition : remplacement d'un bloc 2×2 par un bloc 4×4 situé en haut à droite.

l'image. Ainsi, l'absence totale de cette partition empêche-t'elle un quelconque décodage de l'image. De même, deviner la partition étant difficile, utiliser une partition uniforme constituée de blocs 16×16 ne résoud pas le problème, comme le montre la figure 3.9. L'image décodée est certe reconnaissable, mais en aucun cas utilisable. Le chapitre 6 proposera une application de cette propriété.

FIG. 3.9: Partition inconnue au décodeur : hypothèse d'une partition uniforme constituée de blocs 16×16.

3.3.2 Erreur sur la valeur d'un résidu de prédiction

3.3.2.1 Impact théorique d'une erreur

Il s'agit d'abord d'évaluer théoriquement la propagation de l'erreur sur les phases de prédiction. Considérons d'abord la toute première valeur transmise, à savoir celle encodant le premier pixel du niveau supérieur de la pyramide. Le balayage *raster* de la MICD propage l'erreur sur l'ensemble de la décomposition et on observe un décalage sur l'image entière.

Pour les niveaux inférieurs, il faut tenir compte des deux pyramides entrelacées et séparer l'information globale de l'information de texture. La figure 3.10 illustre la façon dont se propage une erreur réalisée sur la première ou la seconde diagonale d'un bloc 2×2 au sein d'un même niveau de la pyramide, sans distinction de la nature des blocs. La figure fait apparaître que les limites de la propagation d'une erreur sont plus éloignées dans le cas

d'une erreur sur un résidu de la première pyramide que pour une erreur sur un résidu de la seconde pyramide.

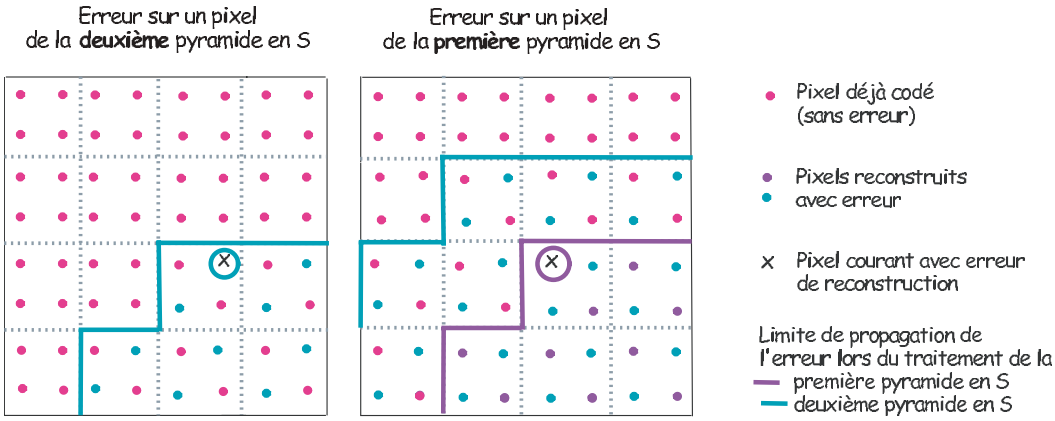


FIG. 3.10: Propagation intra-niveau des erreurs

3.3.2.2 Expérimentations

Les expérimentations introduiront donc une erreur dans la transmission des résidus de prédiction. La nature de l'erreur sera soit une erreur sur le niveau supérieur de la pyramide soit une erreur sur les niveaux inférieurs pour lesquels le *flat*-LAR et la texture locale sont traités différemment.

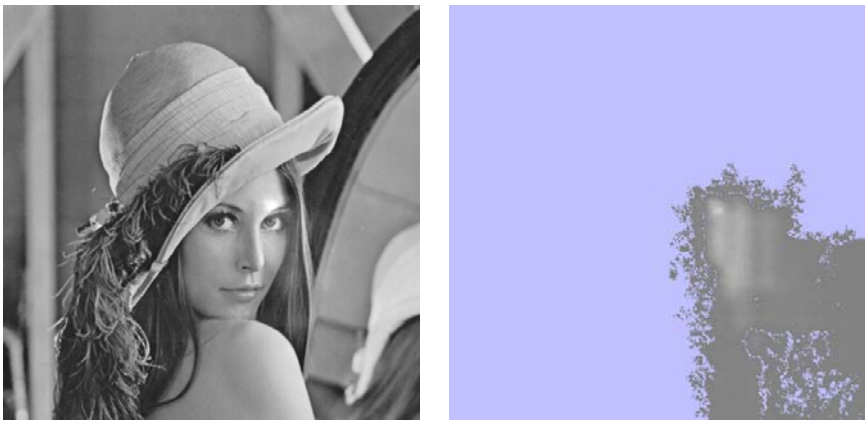
(a) image originale (b) image reconstruite (c) lena : propagation de l'erreur

FIG. 3.11: Valeur d'erreur de prédiction erronée observée sur le pixel $Y_{l_T}(0,0) = Y_4(0,0)$.

La figure 3.11 montre l'effet d'une erreur sur le premier résidu correspondant au pixel $Y_{l_T}(0,0)$. La valeur utilisée pour l'erreur est éloignée de la

valeur originale pour avoir un effet visible. On observe bien un décalage de la valeur moyenne de l'image.

La figure 3.12 présente l'impact visuel d'une erreur ayant lieu sur un résidu du niveau le plus haut de la pyramide, différent du cas précédent. On peut observer que l'erreur se propage surtout dans le quadrant de l'image en bas à droite dont le point d'origine est le pixel modifié. Les prédictions inter et intra-niveaux appliquées successivement contribuent à élargir la zone modifiée.



(a) image reconstruite (b) propagation de l'erreur

FIG. 3.12: Valeur d'erreur de prédiction erronée observée sur un pixel du niveau $l_T = 4$.

Il apparaît que la perception visuelle de l'erreur sera d'autant plus grande qu'elle est présente dans les niveaux élevés de la pyramide. Nous présenterons donc principalement des résultats concernant le niveau $l_T - 1 = 3$ de la pyramide.

La figure 3.13 recense les trois cas de propagation de l'erreur lors de la première décomposition pyramidale pour l'obtention de l'image *flat*-LAR. Les pixels affectés correspondent bien à ceux prévus dans notre étude théorique. Cependant, malgré la forte perturbation introduite, les dégradations restent très localisées, de telle sorte que les images résultantes à pleine résolution gardent globalement une qualité visuelle étonnante, comme le montrent les figures 3.14 et 3.15.

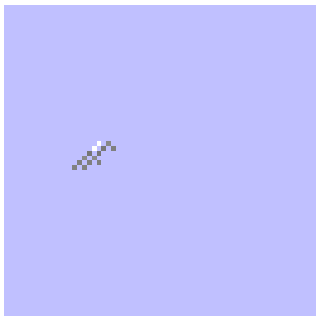
L'explication en est que deux phénomènes entrent conjointement en jeu. D'une part, la prédiction opère une élimination naturelle des valeurs aberrantes, comme un filtrage médian par exemple. D'autre part, l'estimation d'une valeur de pixel tient compte d'un grand nombre de voisins, favorisant ainsi le lissage de la donnée erronée. Pour un même niveau de la décompo-

(a) Erreur sur $z_1^{3,2}$, gradient de la 1re pyramide en S du niveau 3

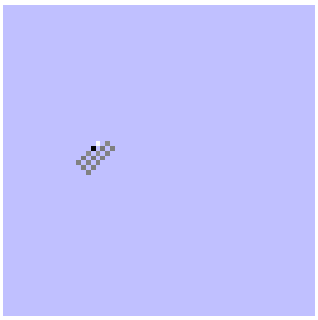
(b) Erreur sur $z_0^{3,3}$, moyenne de la 2e pyramide en S du niveau 3

(c) Erreur sur $z_1^{3,3}$, gradient de la 2e pyramide en S du niveau 3

(d) Propagation de l'erreur issue de 3.13a



(e) Propagation de l'erreur issue de 3.13b

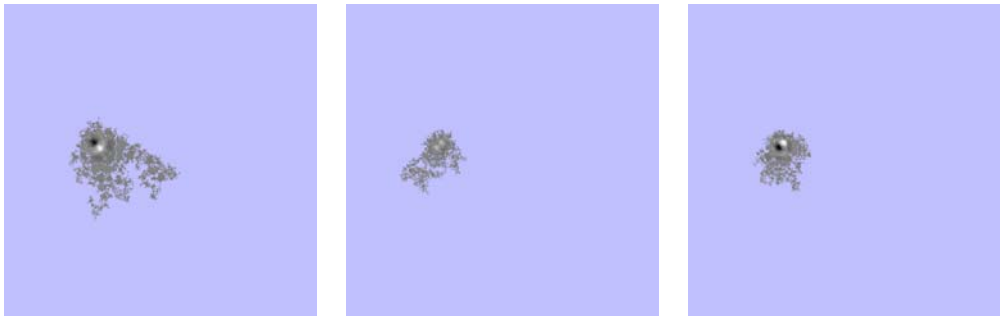


(f) Propagation de l'erreur issue de 3.13c

FIG. 3.13: Observation de la propagation intra-niveau de l'erreur.

sition, l'erreur se propagera d'autant plus qu'elle sera effective au niveau de la première descente pyramidale.

(a) Erreur sur $z_1^{3,2}$ (b) Erreur sur $z_0^{3,3}$ (c) Erreur sur $z_1^{3,3}$



(d) Propagation de l'erreur issue de 3.14a (e) Propagation de l'erreur issue de 3.14b (f) Propagation de l'erreur issue de 3.14c

FIG. 3.14: Observation de la propagation complète de l'erreur (intra- et inter-niveau) appliquée sur l'image LAR basse résolution : décomposition totale de la pyramide.

La figure 3.16 permet de vérifier les faits suivants : d'abord, l'impact visuel est négligeable en cas de *petite* erreur commise sur une valeur reconstruite, comme sur la figure 3.16a, pour laquelle $\hat{z}_1^{3,3} - z_1^{3,3} = 50$, et ensuite, si la valeur erronée est obtenue dans les niveaux inférieurs de la pyramide, les déformations résultantes sont moindres. Enfin, l'erreur diffuse différemment selon qu'elle s'applique lors de la première ou de la deuxième descente de la pyramide : dans ce dernier cas, sa propagation reste relativement restreinte.

(a) Erreur sur $z_1^{3,2}$

(b) Erreur sur $z_0^{3,3}$

(c) Erreur sur $z_1^{3,3}$

(d) Propagation de l'erreur issue de 3.15a

(e) Propagation de l'erreur issue de 3.15b

(f) Propagation de l'erreur issue de 3.15c

FIG. 3.15: Observation de la propagation complète de l'erreur (intra- et inter-niveau) appliquée sur l'information de texture : décomposition totale de la pyramide.

(a) petite erreur sur $z_1^{3,3}$ - image LAR

(b) erreur sur $z_1^{1,2}$ - image LAR

(c) erreur sur $z_1^{1,2}$ - texture

FIG. 3.16: Exemple d'images reconstruites après réception d'une valeur erronée.

3.4 À retenir

Le codage de la partition quadtree est particulièrement efficace et fait de cette partie du LAR iS+P un élément indispensable pour le décodage de l'image. Il apparaît également que des erreurs apparaissant sur les niveaux hauts de la pyramide ont des effets désastreux sur la qualité de l'image.

Deux principaux enseignements peuvent être tirés de ces observations. D'abord, l'importance relative des différents flots binaires du LAR iS+P permet d'adapter leur protection vis-à-vis des erreurs de transmission. Babel et al. (2005c) est un exemple d'ajout de redondance prenant en compte cette différentiation. Cette redondance est amenée par l'utilisation de la transformée MOJETTE introduite par Guédon. Ensuite, la difficulté de reconstruction de l'image en l'absence de la partition quadtree amène à considérer cette partition comme la clef de l'image. Sans cette clef, il est *quasiment* impossible de décoder l'image originale. Le chapitre 6 en fera sa pierre angulaire.

Deuxième partie

Chiffrement d'images

E. P. Jacobs, *Le secret de l'espadon, tome 1*, page 20, case 2.

4

Chiffrement de données

Plan

4.1 Crypto-vocabulaire
4.2 Sécurité des cryptosystèmes
4.3 Structures de chiffrement
4.4 Chiffrement par flot
4.5 Chiffrement par bloc
4.6 Primitives de chiffrement
4.7 À retenir

Les premiers chapitres de ce manuscrit ont présenté la famille de codeur LAR, en particulier le codeur LAR IS+P. Ce codeur est la brique fondamentale d'un système d'archivage sécurisé. Ainsi, le présent chapitre a pour objectif de rappeler les principales problématiques liées au chiffrement de données. Après un bref rappel du vocabulaire utilisé, les notions liées à la sécurité sont montrées. Ensuite, les principales structures de chiffrement, symétrique et asymétrique, sont présentées. Les mises en œuvre particulières, de chiffrement par flot et par bloc forment les sections qui suivent. Enfin, le chiffrement d'image utilisant des primitives de chiffrement de données dans son fonctionnement, nous verrons quelques unes des primitives et normes de chiffrement les plus utilisées.

4.1 Crypto-vocabulaire

Chiffrer est la méthode ou le processus permettant de protéger une information d'une attaque indésirable en transformant cette information en un message illisible pour l'attaquant.

Le chiffrement est une discipline scientifique très vaste, allant de l'informatique théorique à l'électronique rapide en passant par la théorie des nombres. Il est donc illusoire de penser le présenter de manière exhaustive en quelques pages. Le lecteur curieux trouvera dans Schneier (1996) des éléments factuels d'approches relativement aisées, tandis que Menezes et al. (1996) est un ouvrage abordant en plus les aspects théoriques avancés. La lecture de Talbot & Welsh (2006) pourra compléter les deux premiers ouvrages.

La cryptologie, étymologiquement la science du secret, ne peut être vraiment considérée comme une science que depuis peu de temps. Cette science englobe la cryptographie – l'écriture secrète – et la cryptanalyse – l'analyse de cette dernière. C'est une science nouvelle dans le sens où ce thème de recherche scientifique académique (comprendre universitaire) n'a émergé que depuis les années 1970. Cette discipline est liée à beaucoup d'autres, par exemple la théorie des nombres, l'algèbre, la complexité, la théorie de l'information, ou encore les codes correcteurs d'erreurs.

La **cryptographie** est l'étude des écritures secrètes, la science visant à créer des cryptogrammes. La cryptographie est ainsi l'art de chiffrer.

La **cryptanalyse** est la science de l'analyse des cryptogrammes en vue de les décrypter. Elle s'oppose donc en quelque sorte à la cryptographie. En effet, si déchiffrer consiste à retrouver le texte clair au moyen d'une clef, cryptanalyser, c'est tenter de se passer de cette dernière.

Pour être plus précis, les opérations existantes relatives à la cryptologie sont respectivement le chiffrement/déchiffrement et le décryptage. Le chiffre-

ment est la transformation à l'aide d'une clef de chiffrement d'un message en clair en un message incompréhensible si on ne dispose pas d'une clef de déchiffrement (en anglais *encryption*). D'ailleurs, le chiffre est la variante moderne du code secret, et par extension, la méthode ou l'algorithme utilisé pour le chiffrement. Le cryptogramme est le message chiffré. Par conséquent, décrypter consiste à retrouver le message clair correspondant à un message chiffré sans posséder la clef de déchiffrement (terme que ne possède pas les anglophones, qui eux « cassent » des codes secrets).

La taxonomie des cryptosystèmes utilise différentes caractéristiques comme :

- **le type de clef :** symétrique, asymétrique ou hybride ;
- **la technique de chiffrement :** par bloc ou par flot ;
- **le lien entre textes clairs et chiffrés :** synchrone ou asynchrone.

Ainsi, le cryptosystème AES décrit par Daemen & Rijmen (2002) est-il symétrique par bloc. Ces différentes notions seront explicitées par la suite.

4.2 Sécurité des cryptosystèmes

4.2.1 Principes

L'étude de la sécurité des cryptosystèmes est aussi ancienne que le chiffrement lui-même. Si nous ne regardons que des considérations modernes, la véritable base de la cryptographie moderne peut être vue à travers le travail de Kerckhoffs (1883). Celui-ci énonce en particulier les six principes suivants :

- le système doit être matériellement, sinon mathématiquement, indéchiffrable ;
- il faut qu'il n'exige pas le secret, et qu'il puisse sans inconvénient tomber entre les mains de l'ennemi ;
- la clef doit pouvoir être communiquée et retenue sans le secours de notes écrites, et être changée ou modifiée au gré des correspondants ;
- il faut qu'il soit applicable à la correspondance télégraphique ;
- il faut qu'il soit portatif, et que son maniement ou son fonctionnement n'exige pas le concours de plusieurs personnes ;
- enfin, il est nécessaire, vu les circonstances qui en commandent l'application, que le système soit d'un usage facile, ne demandant ni tension d'esprit, ni la connaissance d'une longue série de règles à observer.

Les principales notions de sécurité sont présentes. En particulier, l'idée est que la sécurité réside dans le secret entourant la clef et non dans le secret de la méthode de chiffrement. Tout à la différence de l'idiome, encore trop fréquent, de *security by obscurity*, qui consiste à croire qu'une méthode de chiffrement uniquement si son algorithme est secret.

La deuxième base de la sécurité est le travail de Shannon (1949) qui introduit la notion de **sécurité inconditionnelle**. L'idée est simple : un cryptosystème est à sécurité inconditionnelle si aucune information ne peut être obtenue sur le texte clair à partir du texte chiffré correspondant. Il doit donc y avoir parfaite indépendance, au sens statistique, entre les deux. Dans la foulée, SHANNON montre que le seul chiffrement qui convienne est le chiffrement par *masque jetable*. Celui-ci utilise une clef aussi longue que le message à chiffrer. De plus, cette clef doit être *parfaitement aléatoire*. Une séquence binaire pseudo-aléatoire n'en est qu'une approximation imparfaite. Il faudrait utiliser un processus physique réel, comme le bruit thermique dans une résistance, par exemple, pour éventuellement générer une telle clef. SHANNON présente aussi les deux principaux mécanismes du chiffrement, à savoir la confusion et la diffusion. La confusion mesure l'absence de corrélation entre la clef de chiffrement et le texte chiffré. Un mécanisme de chiffrement destiné à rendre confus peut consister en la substitution d'un symbole par un autre. La diffusion, quant à elle, indique la dispersion de la redondance statistique du texte clair dans le texte chiffré. La transposition de symboles est un moyen d'augmenter la diffusion.

La contrainte de perfection énoncée par SHANNON n'étant pas atteignable, Diffie & Hellman (1976) introduisent la notion de **sécurité calculatoire**. L'idée est la suivante : les ressources de calcul sont limitées, en quantité et en durée. Ainsi, il suffit que le décryptage du message consomme plus de ressources que celles disponibles, avec quelques ordres de grandeurs pour assurer une meilleure sécurité. De même, si la durée de décryptage est beaucoup plus longue que la pertinence du secret, le chiffrement sera considéré comme sûr calculatoirement.

Même si nous anticipons sur la présentation des principales structures de chiffrement de la section 4.3, le type de sécurité assuré dépendra du type de chiffrement, symétrique ou asymétrique utilisé. Pour les chiffrements symétriques, seule la notion de sécurité inconditionnelle est valable. En effet, nous ne sommes capables que de montrer leur résistance à la cryptanalyse existante ou aux attaques connues. Il n'est pas possible d'anticiper les résultats pour des attaques inconnues. Par contre, les chiffrements asymétriques reposent sur la difficulté calculatoire de problèmes de la théorie des nombres ou de l'algèbre discrète. Dans ce domaine, des preuves mathématiques sont disponibles et donc l'évaluation de la complexité calculatoire du décryptage est possible. Seule une révolution dans les mathématiques ou leur mise en œuvre peut modifier ces résultats.

4.2.2 Cryptanalyse

Même si les cryptanalystes sont décrits comme des briseurs de code, il convient de remarquer qu'un algorithme de chiffrement est considéré comme cassé lorsqu'une attaque permet de retrouver la clef en effectuant moins d'opérations que pour l'attaque par force brute qui consiste à essayer toutes les solutions possibles. L'algorithme ainsi cassé ne devient pas inutile pour autant, mais son degré de sécurité, c'est-à-dire le nombre moyen d'opérations nécessaires pour le décrypter, s'affaiblit.

4.2.2.1 Typologie des attaques

Les attaques sont le moyen qu'utilisent les cryptanalystes pour retrouver la clef de chiffrement. La première des attaques est l'attaque par force brute qui opère en essayant toutes les clefs. Il s'agit donc d'une recherche exhaustive qui peut réussir si, d'une part, l'espace des clefs n'est pas trop grand, et si, d'autre part, l'opération de déchiffrement n'est pas trop longue. Ce type d'attaque bénéficie des avancées technologiques en termes de puissance de calcul et de la mise en parallèle de ressources distribuées. En ce qui concerne les attaques *classiques*, elles sont caractérisées par les données qu'elles nécessitent :

- **attaque à texte chiffré seul :** le cryptanalyste ne possède que des exemplaires chiffrés des messages. Il ne peut pas faire d'hypothèses sur les messages originaux qu'il ne possède pas. Dans ce cas, la cryptanalyse est la plus ardue par manque d'informations disponibles ;
- **attaque à texte clair connu :** le cryptanalyse possède des messages ou des parties de messages en clair ainsi que des versions chiffrées. La cryptanalyse linéaire entre dans cette catégorie ;
- **attaque à texte clair choisi :** le cryptanalyste possède des messages en clair. Il peut générer les versions chiffrées de ces messages avec l'algorithme que l'on peut dès lors considérer comme une boîte noire. La cryptanalyse différentielle est un exemple d'attaque à texte clair choisi ;
- **attaque à texte chiffré choisi :** le cryptanalyste possède des messages chiffrés et demande la version en clair de certains de ces messages pour mener l'attaque.

Par ailleurs, certaines attaques n'ont pas lieu dans le domaine informatique. Par exemple, le relevé de signaux électriques sur des câbles permet de récupérer une information de bord sur le chiffrement. L'idée sous-jacente est que la récupération de la moindre information donne une indication pour le décryptage du message chiffré.

4.2.2.2 Techniques de cryptanalyse

Il existe plusieurs familles d'attaques cryptanalytiques, les plus connues étant l'analyse fréquentielle, la cryptanalyse différentielle et la cryptanalyse linéaire.

L'analyse fréquentielle est la première attaque développée. Elle examine les répétitions des lettres du message chiffré afin de trouver la clef. Elle est inefficace contre les chiffrements modernes. Elle est principalement utilisée contre les chiffrements mono-alphabétiques qui substituent chaque lettre par une autre et qui présentent un biais statistique. Sinkov (1966) donne davantage de détails à ce propos.

Ainsi, l'indice de coïncidence permet de calculer la probabilité de répétitions des lettres du message chiffré. Il est souvent couplé avec l'analyse fréquentielle. Cela permet de savoir le type de chiffrement d'un message (mono ou poly-alphabétique) ainsi que la longueur probable de la clef.

L'attaque par mot probable consiste à suppposer l'existence d'un mot probable dans le message chiffré. Il est donc possible d'en déduire la clef de chiffrement si le mot choisi est adéquat. Une illustration intéressante en est faite dans une des *Histoires extraordinaires*, d'Edgar Alan POE, *le scarabé d'or*.

L'attaque par dictionnaire consiste à tester tous les mots d'une liste comme clef. Elle est souvent couplée à l'attaque par force brute.

L'attaque par force brute à déjà été évoquée. Elle est le seul moyen de récupérer la clef dans les algorithmes les plus modernes et encore inviolés. Pour d'anciens standards, Diffie & Hellman (1977) et Wiener (1996) montrent des exemples de chiffrements cassés par recherche exhaustive.

L'attaque par paradoxe des anniversaires est une approche probabiliste utilisée contre les fonctions de hachage. L'idée est de calculer combien de messages peuvent donner la même valeur de hachage. On trouvera des éléments théoriques dans l'article de Wendl (2003). De même, Knuth (1998), alors qu'il propose des tests empiriques d'évaluation de générateurs de séquences binaires pseudo-aléatoires, propose deux tests, *par collision* et *par espacement des anniversaires*, dont la nature est proche d'une attaque par paradoxe des anniversaires.

La cryptanalyse linéaire consiste à faire une approximation linéaire de la structure interne de la méthode de chiffrement. Cette méthode est par exemple utilisée contre DES par Matsui (1993).

La cryptanalyse différentielle est une analyse statistique des changements dans la structure de la méthode de chiffrement après avoir modifié légèrement les entrées. Avec un très grand nombre de pertubations, il est possible

d'extraire la clef. Biham & Shamir (1993) présentent l'application de la cryptanalyse différentielle à l'attaque de DES.

4.3 Structures de chiffrement

La sécurité d'un système de chiffrement repose sur une idée-force : l'adversaire connaît l'algorithme de chiffrement. La confidentialité de l'algorithme ne garantit pas la sécurité. Celle-ci repose sur l'utilisation d'une clef de chiffrement. C'est le binôme algorithme–clef de chiffrement qui assure, éventuellement, la sécurité du chiffrement. La figure 4.1 présente le schéma canonique des systèmes de chiffrement.

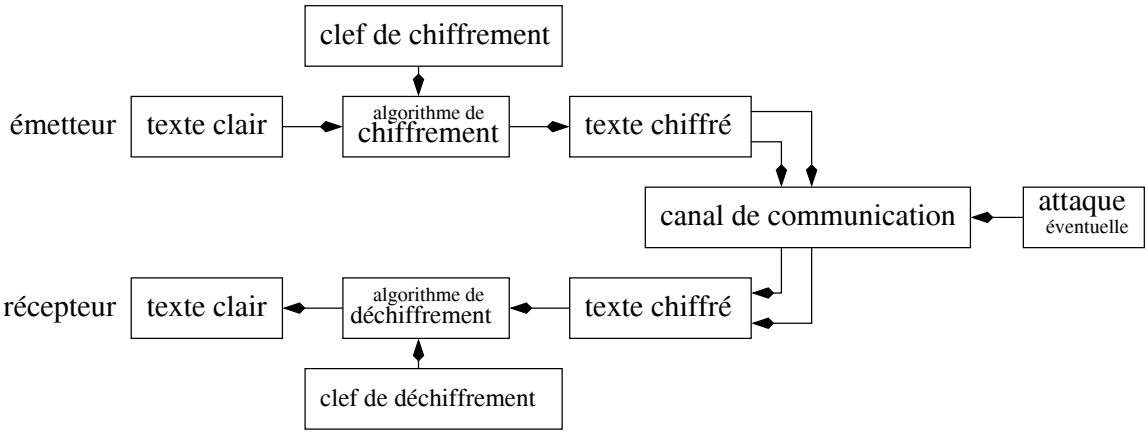


FIG. 4.1: Principe du chiffrement

Deux schémas de chiffrement cohabitent. Le premier suppose la transmission d'une clef secrète. On parle alors de chiffrement à clef privée ou chiffrement symétrique. Le second utilise une clef pour le chiffrement et une autre pour le déchiffrement. On parle de chiffrement à clef publique ou chiffrement asymétrique.

4.3.1 Chiffrement symétrique

Le chiffrement symétrique utilise une même clef pour le chiffrement et pour le déchiffrement. Cette clef doit donc être transmise de manière sûre. Cette clef est donc également unique pour chaque paire (émetteur-récepteur). La figure 4.2 présente une communication à deux utilisant un chiffrement à clef privée, avec un canal assuré pour l'échange de la clef de communication. La clef de déchiffrement doit être aisément calculée à partir de la clef

de chiffrement. Le cas le plus simple est lorsque clef de chiffrement et de déchiffrement sont strictement identiques.

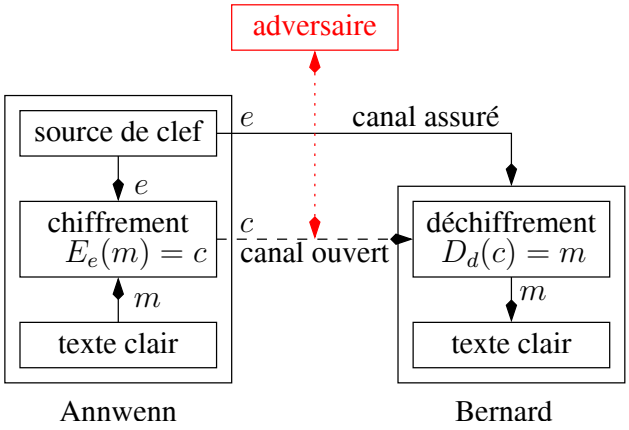


FIG. 4.2: Principe du chiffrement symétrique

La principale propriété attendue d'un cryptosystème symétrique est le caractère aléatoire de sa sortie. Pour simplifier, un chiffrement symétrique est un générateur de nombres binaires pseudo-aléatoires, fournissant une série de bits uniformément distribués.

Le chiffrement symétrique possède quelques avantages :

- le chiffrement symétrique permet un débit binaire élevé. Les mises en œuvre matérielles chiffrent plusieurs centaines de megaoctets par seconde tandis que les débits des solutions logicielles sont de l'ordre du megaoctet par seconde ;
- les clefs pour le chiffrement symétrique sont relativement courtes ;
- le chiffrement symétrique est une brique élementaire dans la construction de nombreux mécanismes cryptographiques comme la génération de nombres pseudoaléatoires, les fonctions de hâchage et les schémas efficaces de signatures numériques, pour n'en nommer que quelques uns ;
- le chiffrement symétrique peut fournir des chiffrements encore plus sûrs par composition. Des transformations simples à analyser et donc plutôt fragiles, peuvent alors autoriser des chiffrements sûrs ;
- le chiffrement symétrique est connu depuis longtemps. Il bénéficie donc d'une longue analyse de ses performances et d'un recul certain sur sa sûreté.

Le chiffrement symétrique présente également des défauts :

- dans une communication bipartite, la clef doit rester secrète de chaque côté ;

– dans un réseau de grande taille, le nombre de paires de clefs, une par canal de communication bipartite, est important. Par conséquent, une gestion efficace des clefs de chiffrement nécessite un tiers de confiance inconditionnel ;
– la pratique cryptographique indique qu'il est nécessaire de changer fréquemment la clef de chiffrement d'une communication bipartite, par exemple pour chaque nouvelle session de communication ;
– les mécanismes de signature numérique reposant sur un chiffrement symétrique nécessitent soit des clefs de grande taille soit l'intervention d'un tiers de confiance.

4.3.2 Chiffrement asymétrique

Le chiffrement asymétrique utilise une clef pour chiffrer et une seconde pour déchiffrer. La clef privée sert à chiffrer le message. La clef publique appariée à la clef privée permet de déchiffrer le message. La figure 4.3 présente le principe de ce chiffrement. Ainsi, la communication pour Bernard passe par le chiffrement du message avec la clef publique de celui-ci. Une fois le message chiffré, seule la clef privée de Bernard permet de déchiffrer le message.

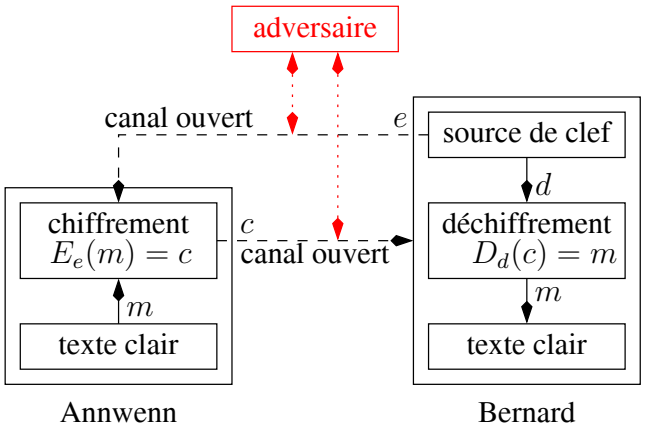


FIG. 4.3: Principe du chiffrement asymétrique

Le chiffrement asymétrique présente les avantages suivants :
– seule la clef privée doit être secrète. Néanmoins, l'authentification des clefs publiques doit être garantie ;
– l'administration des clefs publiques sur un réseau ne requiert qu'un seul tiers de confiance fonctionnel et non inconditionnel ;
– suivant l'usage, la paire clef publique–clef privée n'a pas besoin d'être changée sur une longue période, pour plusieurs sessions par exemple ;

- les chiffrements asymétriques permettent des schémas de signature numérique efficaces. La longueur des clefs pour une vérification publique est souvent plus petite que le cas des clefs symétriques ;
- dans un grand réseau, le nombre de clefs nécessaires est nettement plus faible que si on utilise des schémas de chiffrement symétrique.

D'un autre côté, ses défauts sont :

- les débits binaires du chiffrement asymétrique sont quelques ordres de grandeur plus faibles que ceux du chiffrement symétrique ;
- la taille des clefs publiques est typiquement plus grande que celle des clefs privées pour une sûreté à peu près identique ;
- la démonstration de la sûreté des chiffrements asymétriques reste à faire. Cette sûreté repose sur la difficulté –supposée et communément admise– de quelques problèmes de la théorie des nombres ;
- le chiffrement asymétrique est récent et ne bénéficie pas du même recul que le chiffrement symétrique.

4.3.3 Chiffrement hybride

Le concept de chiffrement hybride fait appel aux deux techniques, symétrique et asymétrique. PGP, ou *Pretty Good Privacy*, inventé par Zimmermann (1995) en est un bon exemple. Il associe la rapidité des méthodes symétriques à la sécurité des algorithmes asymétriques. Dans PGP, une clef secrète est générée automatiquement pour la session. Le message est chiffré de manière symétrique avec cette clef. La clef est ensuite chiffrée de manière asymétrique en utilisant la clef publique du destinaire. Le message chiffré et la clef chiffrée sont concaténés et envoyés au destinataire. Celui-ci utilise sa clef privée pour déchiffrer la clef permettant à son tour de déchiffrer le message. Cette technique évite d'utiliser un tiers de confiance et simplifie la gestion des clefs. Elle est en usage par exemple de l'échange de messages électroniques.

4.3.4 Quelle taille de clef utiliser ?

Le paramètre essentiel influençant la sécurité d'un cryptosystème est la longueur de la clef de chiffrement. À la lumière des remarques faites à propos des avantages et inconvénients des structures de chiffrement, il apparaît que les clefs de chiffrement seront plus longues pour un chiffrement asymétrique que pour un chiffrement symétrique.

Actuellement, les systèmes symétriques sont considérés comme sûrs avec une clef de 128 bits tandis que des clefs de taille supérieure à 2048 bits

sont nécessaires pour les méthodes asymétriques présentant la même sécurité calculatoire.

4.4 Chiffrement par flot

Le chiffrement par flot fonctionne comme une approximation du masque jetable. Il s'agit de générer, à partir de la clef de chiffrement, un flot de clefs combinées au message pour le chiffrer. Dans le cas binaire, la génération se fait avec un générateur binaire pseudo-aléatoire. La combinaison de cette séquence et du message se fait par l'addition (modulo 2), ou, de manière équivalent, un OU-exclusif.

La réalisation d'un chiffrement par flot demande des circuits peu complexes et consomme peu de mémoire. Il est donc adapté aux systèmes à ressources limitées, comme les systèmes embarqués. C'est le cas de la téléphonie GSM[1] et des communications sans fil type WIFI[2].

La manière de générer le flot de clefs permet de distinguer les chiffrements. Si le flot de clefs est construit indépendemment du texte clair, le chiffrement est dit **synchrone**. Dans le cas d'une construction dépendant du texte clair, le chiffrement est appelé **asynchrone**.

La génération de séquences binaires pseudo-aléatoires est un domaine de recherche important, à la fois en mathématique et en informatique. Knuth (1998) y consacre un chapitre entier de son œuvre fondatrice. Parmi les méthodes classiques figurent les générateurs congruentiels linéaires. La clef de chiffrement peut, par exemple, être utilisée pour donner le point de départ de l'algorithme et faire son initialisation. Un excellent et moderne générateur de séquences binaires pseudo-aléatoire est la *tornade de Mersenne*, introduite par Matsumoto & Nishimura (1998).

De manière intéressante, un chiffrement synchrone peut être obtenu en utilisant les modes OFB (*Output FeedBack*) et CTR (*CounTeR*) des chiffrements par blocs. De même, un chiffrement asynchrone peut reposer un chiffrement par bloc en mode CFB (*Cipher FeedBack*). Ces modes propres à chaque type de chiffrement seront développés par la suite.

4.5 Chiffrement par bloc

Le chiffrement par bloc opère sur des blocs de données, souvent de taille 64 ou 128 bits. La clef est unique pour tout le traitement. La plupart des chif-

[1] *Global System for Mobile communications* décrit par Redl et al. (1995)

[2] nom d'usage pour toutes les applications des normes IEEE 802.11.

frements par bloc fonctionnent par itération d'une fonction simple. Chaque étape est appelée une ronde, itérée souvent entre 4 et 32 fois. L'architecture générale de tels chiffrements est appelée réseau de Feistel, ou de façon plus générale, réseaux de substitution–permutation. Ces réseaux permettent d'obtenir à la fois la confusion et la diffusion.

4.5.1 Réseaux de Feistel

Les réseaux de Feistel et les constructions similaires font partie de la famille des réseaux de *substitution-permutation*. Ils sont produits en combinant plusieurs rondes d'opérations comme :

- un mélange de bits ou permutations, boîte-P ;
- une fonction non linéaire simple ou substitution, boîte-S ;
- une opération linéaire de type ou-exclusif.

afin d'obtenir ce que Shannon (1949) appelle *confusion* et *diffusion*. Ici, le mélange de bits fournit la confusion tandis que la substitution introduit la diffusion. Un mécanisme de génération de clefs, appelé *key-schedule* permet de générer, à partir d'une clef principale, un ensemble de clefs utilisé dans les rondes.

Le principal intérêt de cette structure est que le chiffrement et le déchiffrement sont similaires, voire identiques dans certains cas. De plus, leur relative simplicité en fait des outils indiqués pour des applications à faibles ressources. La méthode de chiffrement DES, que nous verrons en 4.6.1 utilise des réseaux de FEISTEL.

4.5.2 Modes de fonctionnement

Les chiffrements par blocs opèrent selon différents modes, précisant la façon d'adapter des textes clairs de taille quelconque à un découpage en blocs de taille fixe.

4.5.2.1 Vecteur d'initialisation

Un vecteur d'initialisation permet d'amorcer tous les modes de chiffrement par bloc, hormis le mode ECB détaillé ci-après. C'est un bloc « blanc » non nécessairement secret, qui permet cependant d'ajouter un peu de désordre au chiffrement.

4.5.2.2 Mode ECB

Le mode ECB, *Electronic Code Book*, ou dictionnaire de codes, est le mode le plus simple. Le texte clair est découpé en plusieurs blocs qui sont

chiffrés indépendemment les uns après les autres. La figure 4.4 montre le chiffrement et le déchiffrement correspondant. Le principal défaut de cette méthode est que deux blocs clairs identiques deviennent deux blocs chiffrés identiques. Ainsi, on peut tirer des informations à partir du texte clair en faisant la recherche des blocs identiques dans le texte chiffré. On obtient donc un dictionnaire de codes des correspondances entre blocs clairs et chiffrés.

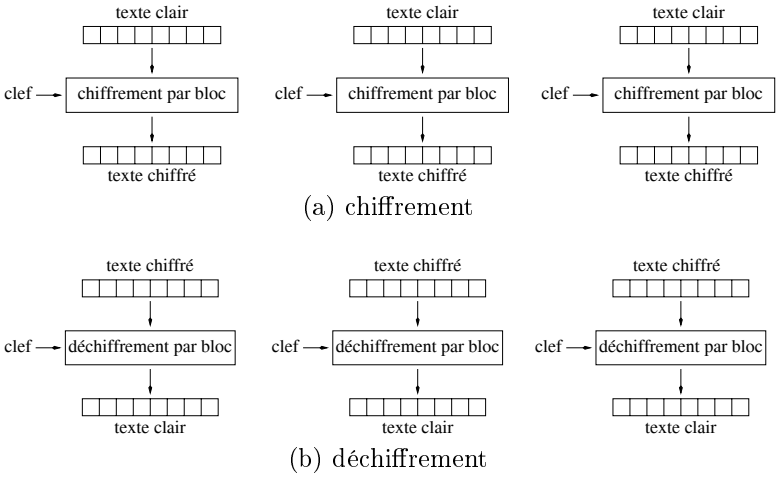


FIG. 4.4: Chiffrement et déchiffrement par bloc : mode ECB.

Ce mode est aussi sensible aux attaques par répétitions qui consistent à réinjecter dans le système des données identiques à celles déjà interceptées. Cela permet soit de modifier le comportement du système soit de répéter des actions. Pour ces raisons, le mode ECB est déconseillé pour des applications cryptographiques. Son principal intérêt est qu'il permet d'accéder à n'importe quel bloc de données, de manière aléatoire, sans avoir à déchiffrer les blocs précédents.

4.5.2.3 Mode CBC

Le mode CBC, *Cipher Block Chaining*, ou enchaînement des blocs, lie le bloc clair à chiffrer avec le bloc chiffré précédent, avant de procéder à son chiffrement, comme montré sur la figure 4.5. La transformation consiste en un ou-exclusif tandis qu'un vecteur d'initialisation permet de démarrer le système.

Le mode CBC est le mode de chiffrement le plus utilisé. Il a néanmoins comme inconvénient d'effectuer un chiffrement séquentiel. Ainsi, pour déchiffrer un bloc de texte chiffré, il faut avoir auparavent déchiffré tous les blocs

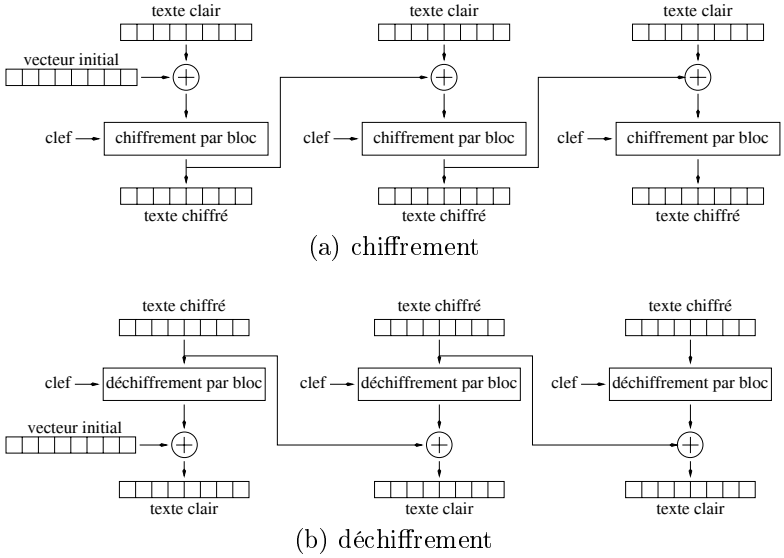


FIG. 4.5: Chiffrement et déchiffrement par bloc, mode CBC.

de texte chiffré placés devant lui. Ce mode de chiffrement ne pourra donc pas être parallélisé.

Le chaînage assure qu'un petit changement dans le texte clair modifie tous les blocs chiffrés suivants. Il permet également de justifier qu'il suffit de deux blocs chiffrés consécutifs pour déchiffrer un bloc de texte clair. Ainsi, contrairement au chiffrement, le déchiffrement peut être parallélisé.

4.5.2.4 Mode CFB

Le mode CFB, *Cipher FeedBack*, est un mode proche du mode CBC. Il fait d'un chiffrement par bloc un chiffrement par flot se synchronisant par lui-même. La figure 4.6 montre que le chiffrement et le déchiffrement utilisent le même chiffrement par bloc.

4.5.2.5 Mode OFB

Le mode OFB, *Output FeedBack*, fait d'un chiffrement par bloc un chiffrement par flot synchrone. Le vecteur d'initialisation, chiffré successivement par la clef et le bloc de chiffrement, fournit un flot de clefs qui sont combinées au texte clair. La symétrie de l'opérateur ou-exclusif explique le fait que, figure 4.7, le même chiffrement par bloc soit utilisé pour le chiffrement et le déchiffrement en mode OFB.

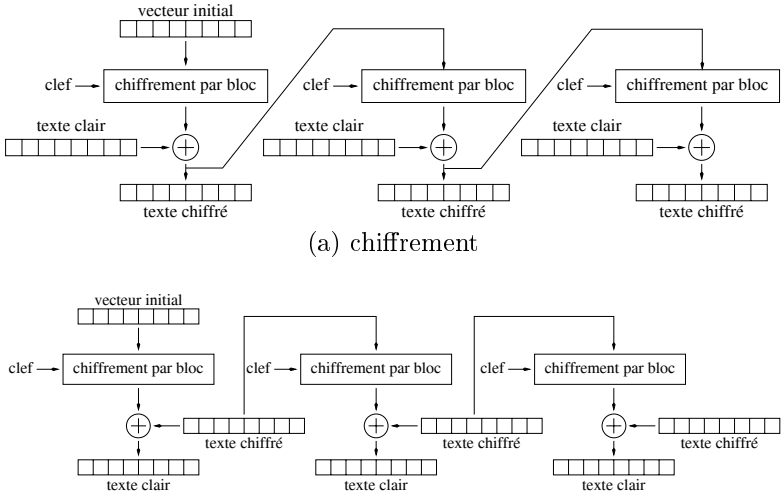


FIG. 4.6: Chiffrement et déchiffrement par bloc, mode CFB.

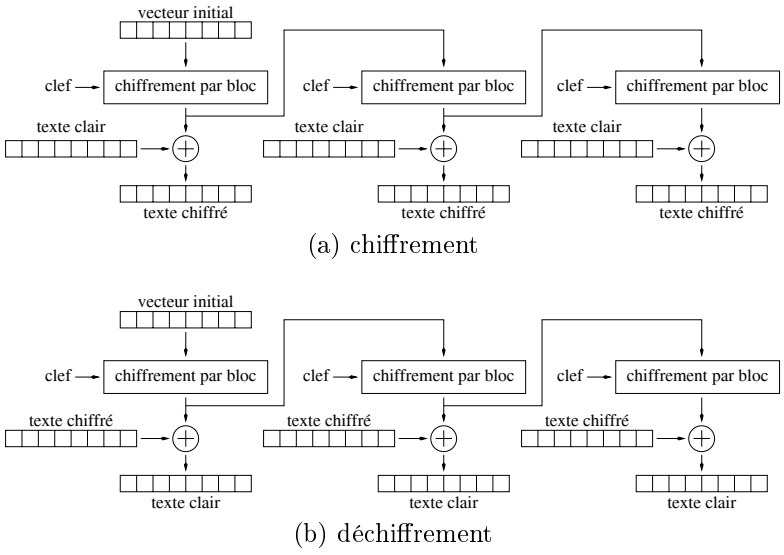
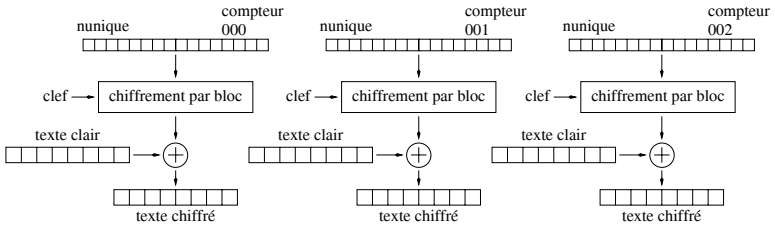


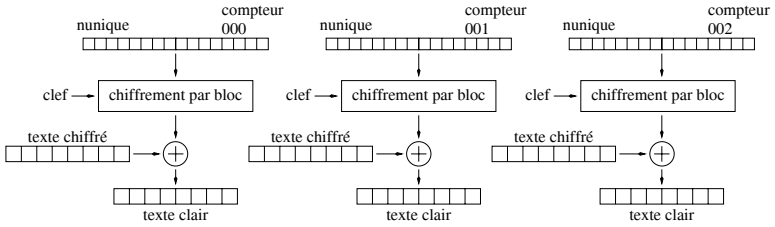
FIG. 4.7: Chiffrement et déchiffrement par bloc, mode OFB.

4.5.2.6 Mode CTR

Le mode CTR, *CounTeR*, fait référence à l'emploi du compteur incrémental pour générer le flot de clefs. Ce mode, tout comme le mode OFB, change un chiffrement par bloc en un chiffrement par flot. Avec un fonctionnement proche du mode OFB, le mode CTR autorise un accès aléatoire aux blocs chiffrés, tout comme le mode ECB.



(a) chiffrement



(b) déchiffrement

FIG. 4.8: Chiffrement et déchiffrement par bloc, mode CTR.

Les chiffrements et déchiffrements peuvent se faire en parallèle, comme le montre la figure 4.8[3]. Ceci autorise une mise en œuvre sur machine multiprocesseurs et donc des vitesses de chiffrement élevées.

4.6 Primitives de chiffrement

Les schémas de chiffrement d'images utilisent des primitives de chiffrement éprouvées. L'interopérabilité est alors assurée par la normalisation des méthodes. Cette normalisation peut être soit formelle, avec la publication d'une norme locale ou internationale, soit *de facto*, avec la publication de documents permettant l'échange de données.

[3]nunique fait référence à un nombre utilisé une seule fois. C'est une francisation de *nonce*, ou *number used once*.

Cette section présente quelques unes des primitives de chiffrement normalisées les plus utilisées. Les chiffrements DES/3DES, AES et RC4 sont des chiffrements symétriques[4] tandis que RSA est un chiffrement à clef publique[5].

4.6.1 Méthode DES/3DES

Le chiffrement DES, ou *Data Encryption Standard* est un chiffrement par bloc de taille 64 bits utilisant une clef de chiffrement de 64 bits. De fait, la taille effective de cette clef est de 56 bits. DES repose sur 16 rondes de FEISTEL après scission du bloc en deux moitiés de 32 bits.

Le DES a été normalisé en 1976 sous la référence *Federal Information Processing Standard* ou *FIPS-1976*, décrit par le document FIPS PUB 46–3 (1999). Cette normalisation lui a valu d'être très utilisé tout en faisant l'objet d'une attention soutenue de la part des cryptanalystes.

Ceux-ci ont trouvé des méthodes théoriques pour casser le chiffre en utilisant moins que les 2^{56} opérations nécessaires pour une attaque de force brute. Cependant, c'est la taille réduite de la clef et la réalisation de circuits spécifiques capables de casser le chiffre par une attaque par force brute en quelques heures qui ont signé l'arrêt de l'utilisation de DES comme primitive de chiffrement.

Pour pallier la faiblesse de la taille de la clef, les concepteurs de DES ont proposé d'utiliser de tripler la taille des clefs en enchaînant trois chiffrements DES. Ainsi, la clef atteint une taille de 168 bits et le nombre de rondes de FEISTEL passe à 48. Le 3DES est considéré comme sûr face aux attaques par force brute. Néanmoins, il tend à être remplacé par le chiffrement AES.

4.6.2 Méthode AES

Le chiffrement AES, *Advanced Encryption Standard*, est le résultat d'un appel à norme de chiffrement lancé par le gouvernement des États-Unis. Daemen & Rijmen (2002) proposent ainsi le chiffrement *Rijndael* qui sera adopté comme norme. C'est un chiffrement par bloc de taille 128 bits, avec des clefs de tailles 128, 192 et 256 bits. N'utilisant pas de rondes de FEISTEL, AES repose sur des réseaux de substitution-permutations de type *SubBytes*, *ShiftRows*, *MixColumns* et *AddRoundKey* pour obtenir les effets de confusion et de diffusion, en effectuant 10, 12 ou 14 rondes de ces réseaux.

Il existe des mises en œuvres rapides pour AES, tant matérielles que logicielles. La publication d'une norme pour AES fait que celui-ci est utilisé

[4] ou à clef privée
[5] ou asymétrique

extensivement et fait l'attention des cryptanalystes. À l'heure actuelle, le chiffrement est considéré comme sûr. Il n'existe pas de méthode plus efficace que la force brute pour casser la clef de chiffrement. Cependant, contrairement à nombre de chiffrements, AES bénéficie d'une description mathématique complète, ce qui pourrait aider les cryptanalystes à le casser dans le futur.

4.6.3 Méthode RC4

Le chiffrement RC4 est le chiffrement symétrique par flot le plus répandu. Il est décrit par exemple dans Kaukonen & Thayer (1999). Le chiffrement génère une séquence binaire pseudo-aléatoire qui est ensuite combinée au texte clair avec un ou-exclusif. Pour générer le flot de clefs, le chiffrement utilise un algorithme de gestion de clefs (*key-scheduling*) dont l'initialisation est fournie par la clef de chiffrement.

La mise en œuvre de RC4 est suffisamment simple pour permettre une vitesse de calcul élevée. Néanmoins, en termes de sécurité, le chiffrement révèle une faille au niveau de son *nunique*. En effet, celui-ci est utilisé sur une longue période de temps et ainsi, son caractère secret s'estompe peu à peu. Ainsi, il est recommandé de ne pas utiliser RC4 pour des applications futures. Néanmoins, ce chiffrement permet un compromis et reste en usage pour les protocoles réseau SSL[6] et WEP[7] par exemple.

4.6.4 Méthode RSA

Le chiffrement RSA fait partie des pierres angulaires du chiffrement public. Il a été inventé par Rivest et al. (1978) et utilise les propriétés de décomposition en facteurs premiers de grands nombres. Cette utilisation de la théorie des nombres est fréquente dans les chiffrements à clef publique.

La grande différence avec les autres méthodes de cette section, est que la clef, pour obtenir un chiffrement asymétrique sûr, doit avoir une longueur supérieure à 1024 bits. Cette valeur est à comparer aux 256 bits au maximum utilisés par AES.

RSA est considéré comme un chiffrement sûr, même si l'on pense qu'un ordinateur quantique pourrait briser le code de manière brutale, en utilisant l'algorithme de SHOR, comme décrit par Shor (1994) lui-même.

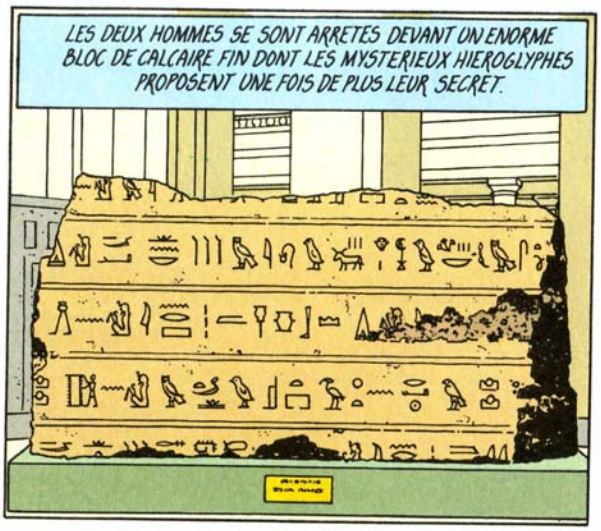
[6] *Secure Sockets Layer* est un protocole de sécurisation des échanges sur internet, de type client-serveur. Thomas (2000) en fait une bonne description.

[7] *Wired Equivalent Privacy* est un protocole de sécurisation des réseaux WIFI, dont les failles présentées par Borisov et al. (2001) sont connues, et qui est supplanté par d'autres méthodes.

4.7 À retenir

Il est souvent nécessaire de cacher le contenu d'un message lors de son transit sur un moyen de communication non sûr. Le message initial est donc chiffré par l'émetteur avant d'être transmis. Cette opération de conversion d'un message clair en un message chiffré est un chiffrement. Ce processus de chiffrement repose à la fois sur un algorithme de chiffrement et sur une clef. Le processus inverse de récupération du message clair à partir du message chiffré et de la clef de chiffrement est appelé déchiffrement. Le même processus de récupération, en l'absence de clef, est appelé cryptanalyse.

Une idée fondamentale de la théorie du chiffrement est que le processus de chiffrement doit être public, tandis que la clef doit être secrète, en tout ou en partie. Dans la suite de notre étude, nous avons sélectionné les primitives AES et RC4 afin de procéder au chiffrement des flots LAR. Cependant, avant de décrire cette méthode, le chapitre suivant s'attache à présenter la problématique spécifique du chiffrement d'image et les méthodes dédiées qui en découlent.



E. P. JACOBS, *Le mystère de la grande pyramide, tome 1*, page 9, case 2.

5

Chiffrement d'images

Plan

5.1 Introduction
5.2 Chiffrement partiel
5.3 Chiffrement sélectif
5.4 Analyse
5.5 Évaluations des performances
5.6 Techniques de chiffrement d'images
5.7 Normes de sécurité pour les images
5.8 À retenir

Il n'est pas aussi simple de chiffrer une image que de chiffrer un texte. Des contraintes nouvelles apparaissent, comme la durée du chiffrement qui peut être importante, la nécessité de conserver un flot binaire décodable et enfin, les corrélations plus fortes existant entre les données chiffrées. Cela amène à considérer des aménagements au chiffrement total de l'image comme, d'une part, le chiffrement partiel et d'autre part le chiffrement sélectif. Le présent chapitre introduit donc les notions importantes et les enjeux du chiffrement d'images. Les propriétés souhaitables sont présentées et, dans l'état de l'art des techniques proposées pour effectuer le chiffrement, une évaluation des performances eu égard à ces critères est effectuée.

5.1 Introduction

Dans des applications de transmission ou de stockage, la compression de l'image est effectuée au préalable afin de réduire la bande passante utilisée. Ensuite seulement, l'image est chiffrée. La figure 5.1 présente un schéma de principe de l'agencement des opérations effectuées.

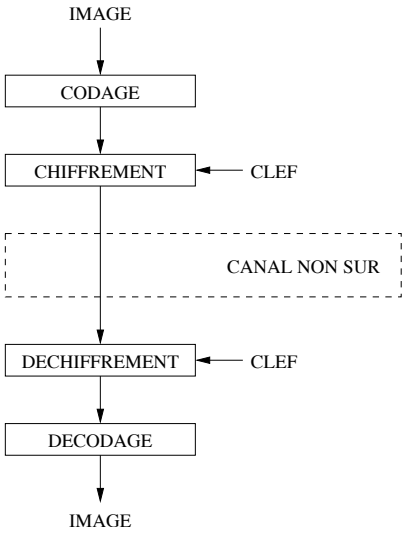


FIG. 5.1: Chiffrement d'une image

La compression réduisant la redondance présente dans l'image augmente la robustesse du chiffrement. En effet, l'absence de redondance élimine des informations sur l'image qui pourraient être utiles au cryptanalyste. Cependant, la compression ajoute des structures nouvelles au message, comme les entêtes et les marques de synchronisation. Ces structures facilitent par contre le travail du cryptanalyste.

Un choix alternatif pourrait être d'intervertir les blocs de chiffrement et de compression dans la figure 5.1. Or, un flot chiffré ne présente que peu de redondance et rend inefficace le schéma de compression qui le suit.

Le chiffrement à proprement parler est souvent effectué par une méthode à clef privée, comme IDEA, introduit par Lai & Massey (1990), de plusieurs ordres de grandeurs plus rapide que les méthodes à clef publique, comme RSA inventé par Rivest et al. (1978). Cependant, l'échange des clefs privées peut se faire à l'aide d'une méthode à clef publique. Pour la suite, on peut considérer que la sécurité de bas niveau du chiffrement est obtenue par l'utilisation de primitives de chiffrement, tant à clef publique que privée, efficaces et adéquates.

En reprenant le schéma de la figure 5.1, on peut faire les remarques suivantes :

- il y a deux types d'informations disponibles : l'image d'un côté, et la clef de chiffrement de l'autre ;
- la signification subjective de l'information contenue dans l'image n'entre pas en ligne de compte. Il n'y a pas, par exemple, de distinction entre les bits de poids fort et de poids faible ;
- tout le flot binaire correspondant à l'image est chiffré. Dans le cas d'images de grandes tailles ou pour une compression sans perte, les volumes à chiffrer sont très importants et les durées de chiffrement ne sont plus négligeables ;
- le récepteur doit déchiffrer tout le flot binaire avant de commencer le décodage de l'image. Il est donc impossible d'obtenir la moindre information sur l'image sans avoir la clef de déchiffrement ;
- le corollaire de la remarque précédente est le suivant : un flot compressé puis chiffré n'est pas décodable. Il ne respecte plus la syntaxe qu'attend le décodeur. Le cryptanalyste peut se servir de ce défaut comme critère de réussite pour une attaque.

Ces défauts permettent d'introduire quatre propriétés souhaitables pour une méthode de chiffrement d'images :

1. prise en compte de l'information image ;
2. réduction de la quantité de données chiffrées ;
3. respect de la syntaxe du codeur d'image ;
4. possibilité de décoder une partie du flot sans avoir la clef.

Les deux premières propriétés sont fournies par des approches de **chiffrement partiel** tandis que les méthodes présentant les quatre propriétés sont qualifiées de **chiffrement sélectif**.

5.2 Chiffrement partiel

Nombreux sont les schémas de compression qui décomposent l'image en plusieurs parties de sémantiques différentes. JPEG, décrit par Pennebaker & Mitchell (1992)par exemple, à travers la TCD, permet de séparer les coefficients DC[1] des coefficients AC[2]. De manière générale, on peut considérer qu'un tel schéma de codage représente une image en différents sous-flots, d'importance sémantique variable. Le fait que les sous-flots sont d'importances différentes autorise à les classer en deux catégories : principale et secondaire. Les sous-flots principaux sont regroupés dans une composante principale tandis que les sous-flots secondaires apparaissent dans une composante secondaire.

Cette dichotomie permet de construire un mécanisme de chiffrement partiel, dans lequel seule la composante principale est chiffrée. Une vue de principe est proposée sur la figure 5.2.

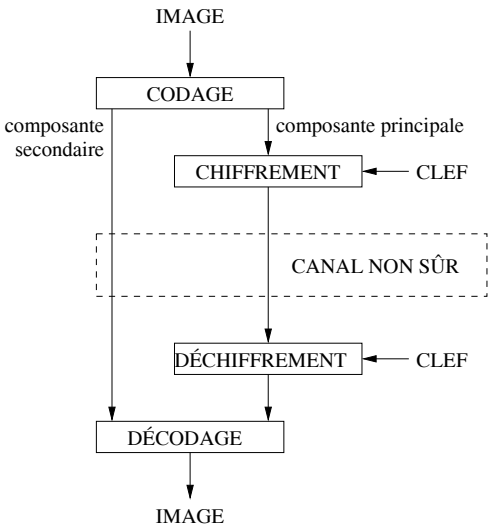


FIG. 5.2: Principe du chiffrement partiel d'image

Si la composante principale est importante, sa taille est souvent de beaucoup inférieure à celle de la composante secondaire. Ainsi, le chiffrement partiel possède-t'il la propriété de réduire fortement la quantité de données chiffrées. Dans certains cas, le volume de données à chiffrer est tellement faible qu'il est envisageable de chiffrer la composante principale par une méthode à clef publique, évitant ainsi le besoin d'échanger des clefs privées.

[1] *Direct Current* c'est-à-dire la valeur moyenne.
[2] *Alternating Current* c'est-à-dire de fréquence non nulle.

Le chiffrement partiel peut également être considéré comme une façon de hiérarchiser l'information, avec la contrainte additionnelle que la composante secondaire n'apporte guère d'information sur l'image originale.

En outre, les schémas de compression cherchent souvent à séparer les données en classes décorrelées afin d'obtenir un codage efficace, en utilisant par exemple une transformation. Si la distinction entre les composantes principale et secondaire repose sur cette décomposition, on peut considérer, avec une bonne confiance, que les deux composantes sont décorrélées. Ainsi, même si la composante secondaire, non chiffrée, est connue, elle ne contient pas assez d'information pour reconstruire la composante principale ou pour déduire la clef de chiffrement. La sécurité fournie par la primitive de chiffrement n'est donc pas remise en cause par le chiffrement partiel.

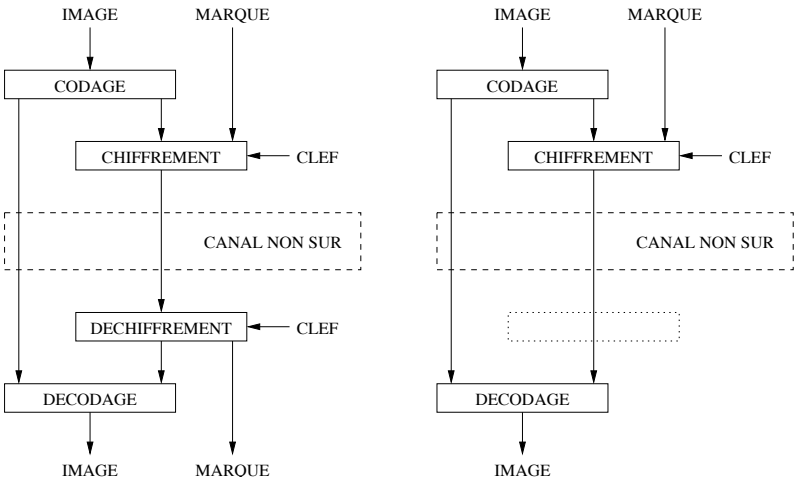
En résumé, le chiffrement partiel prend en compte l'information image à travers la décomposition en deux composantes d'importances sémantiques différentes tout en réduisant la quantité de données chiffrées. Cependant, un flot non déchiffré ne respecte pas la syntaxe admise par le décodeur. Il n'est donc pas décodable.

5.3 Chiffrement sélectif

L'approche du chiffrement sélectif vise à répondre aux quatre propriétés vues à la section 5.1. Son fonctionnement est illustré sur la figure 5.3. L'image est d'abord compressée, puis l'algorithme chiffre seulement une partie du flot binaire avec une méthode de chiffrement éprouvée. Une marque peut également être insérée pendant ce processus. Pour assurer une pleine compatibilité avec le décodeur, le flot binaire est chiffré de manière à respecter la syntaxe de codage. En particulier, les éléments de synchronisation doivent être conservés. De même, on peut aussi se rappeler que les mots d'un code à longueur variable, une fois chiffrés, ne sont pas forcément des mots valides, aptes à être décodés. Ici, le choix des modifications dépend fortement du schéma de codage, JPEG par exemple, et passe par une analyse fine de son fonctionnement. La transposition d'une technique de chiffrement sélectif vers un autre codeur n'est donc pas immédiate.

Le chiffrement sélectif sépare le flot binaire en deux, afin d'en chiffrer une partie. Cependant, à la différence du chiffrement partiel, le chiffrement sélectif place davantage la contrainte en termes de conformité qu'en termes de sémantique sur le contenu de l'image.

Lorsque la clef de chiffrement est connue, comme sur la figure 5.3a, le récepteur déchiffre le flot binaire chiffré et décode l'image. Les seules différences



(a) Fonctionnement nominal (b) Déchiffrement sans clef

FIG. 5.3: Principe du chiffrement sélectif

entre l'image originale et l'image décodée sont dûes à l'utilisation éventuelle d'une marque de tatouage.

Lorsque la clef de chiffrement est inconnue du récepteur, tel qu'illustré sur la figure 5.3b, le récepteur est toujours capable de décoder l'image, mais celle-ci différera notablement de l'image originale. Cependant, cette dégradation peut être telle qu'elle n'empêche pas une visualisation d'une image de mauvaise qualité, à des fins de recherches dans une base de d'images par exemple.

5.4 Analyse

Des spécificités apparaissent à la lumière des différentes approches utilisées dans le chiffrement d'images. D'abord, la conformité avec le schéma ou la norme de codage. Une image compressée puis chiffrée devrait rester une image décodable, même en l'absence de clef. Dans le cas de décodage sans clef, on peut choisir entre une distorsion faible, utile dans l'indexation, ou une distorsion très élevée, afin de masquer le contenu de l'image. Autre spécificité, la nécessité de diminuer le volume de données chiffrées, afin de ne pas allonger la durée du traitement de l'image. Enfin, et en lien avec les deux points précédents, la tendance naturelle pousse à intégrer le chiffrement dans le codeur, pour des raisons évidentes d'efficacité. Ceci n'empêche pas de manipuler les flots binaires compressés hors-ligne, mais il semble que codage et chiffrement joints offrent des perspectives importantes.

5.5 Évaluations des performances

La mesure de la pertinence d'une méthode de chiffrement d'images laisse apparaître deux principales directions d'évaluation des performances : d'un côté les aspects liés à la sécurité, de l'autre les incidences du chiffrement sur les performances de codage, principalement dans le cadre du chiffrement d'image sélectif.

La description et la discussion d'algorithmes présents dans la littérature seront suivies d'une évaluation de chaque technique. Cette évaluation passe par l'analyse des propriétés décrites dans les sections qui suivent.

5.5.1 Coût temporel

La durée additionnelle que nécessite le chiffrement se répartit en deux parties. D'une part, la durée de chiffrement proprement dite, notée $D(E)$, d'autre part, la durée nécessaire à identifier les données effectivement chiffrées, notée $D(P)$. Cette étape peut consister en une analyse syntaxique du flot binaire, ou un pré-traitement d'image. Le coût temporel peut ainsi être noté nul, faible, moyen ou élevé, avec éventuellement la propriété d'être proportionnel à la quantité de données chiffrées. Il s'agit dans ce cas d'un chiffrement dit scalable.

5.5.2 Sécurité

La sécurité d'un système de chiffrement d'images présente deux aspects. D'abord, la sécurité elle-même du chiffrement utilisé, ensuite l'importance et la pertinence des données qui seront chiffrées. La sécurité est vue comme faible, moyenne, élevée, avec la propriété supplémentaire de dépendre de la quantité de données chiffrées. En tenant compte des deux aspects de la sécurité, deux types d'attaques sont à envisager. D'un côté, le chiffrement utilisé peut être en lui-même la cible d'une première attaque. Dans ce cas, les résultats généraux concernant la sécurité du chiffrement utilisé s'appliquent. D'un autre côté, dans le cas d'un chiffrement partiel ou sélectif, il peut être possible de reconstruire le contenu visuel d'une image chiffrée, sans tenir compte des données chiffeés. Selon l'importance des données chiffrées dans la perception visuelle de l'image, le résultat peut varier d'une image complètement incompréhensible à une qualité mauvaise ou réduite. En effet, le bruit hautes fréquences généré par la partie chiffrée des données se propage dans tout schéma de reconstruction directe. Pour éviter ce phénomène, des parades existent. Wen et al. (2002) proposent une attaque par dissimulation des erreurs, Lookabaugh et al. (2003) introduisent une attaque perceptuelle

tandis que Norcen et al. (2003); Norcen & Uhl (2003) suggèrent une attaque par remplacement. Ces attaques procèdent en dissimulant la réduction de qualité provenant des données non décryptées en les considérant comme perdues. Ensuite, elles remplacent les données manquantes par des données minimisant des critères d'erreurs de reconstruction, par des données artificielles reproduisant un contenu simple non structuré ou par des données minimisant l'impact visuel des données incorrectes.

5.5.3 Conformité du flot binaire

Un schéma de chiffrement d'images fixes est dit conforme si le flot binaire chiffré est conforme à la définition d'un flot binaire du schéma de compression utilisé. La conformité du flot binaire est implicite dans tout schéma de compression reposant sur une manipulation des coefficients obtenus dans le schéma de codage. D'un autre côté, les schémas de chiffrement par flot ne tiennent pas compte de cette propriété de conformité. Cette propriété est évaluée par oui ou non.

5.5.4 Traitement dans le domaine compressé

Une propriété importante d'un schéma de chiffrement d'image est caractérisé par le fait qu'il s'applique directement à un flot binaire ou par le fait que le flot binaire doit être décodé avant d'appliquer le chiffrement. Cette propriété est notée oui (domaine compressé) ou non (domaine non compressé).

5.5.5 Effet sur les performances de compression

Les schémas de chiffrement d'images augmentent la taille des flots binaires générés, comparée aux méthodes de compression sans chiffrement. Cette propriété influence les courbes de type débit–distorsion et elle est évaluée oui, modérée ou non.

5.6 Techniques de chiffrement d'images

Le chiffrement d'image, sélectif ou partiel, repose principalement sur une décomposition possible du flot binaire d'une image en plusieurs composantes d'importances diverses. Différents types de décomposition amènent différentes mises en œuvre de chiffrement. Le type de décomposition peut donc servir de critères de classification des méthodes de chiffrement existantes. On retrouve là la dichotomie présente dans les schémas de codage d'images,

avec d'une part les méthodes dans le domaine image, et d'autre part les méthodes reposant sur une transformation. À ceci s'ajoutent les méthodes manipulant directement les flots binaires générés par les codeurs d'images. Les performances des différentes techniques présentées seront résumées dans le tableau 5.1 à la section 5.6.4.

5.6.1 Techniques dans le domaine image

Les techniques de chiffrement dans le domaine image sont principalement des méthodes qui modifient les données brutes de l'image, avant toute transmission ou codage. Ainsi, s'il y a compression, le flot binaire résultant reste décodable. Seulement, l'image est soit complétement aléatoire, soit dégradée. Évidemment, le chiffrement, en ajoutant du désordre dans les images, diminue les corrélations existantes et diminue donc les performances des schémas de codage.

De nombreuses approches concernent le chiffrement des différents plans de bits comme celles présentées par Van Droogenbroeck & Benedett (2002) et Norcen et al. (2003). Le chiffrement est d'autant plus visible que le poids du plan binaire est grand. La dégradation des images est donc réglable par les choix du nombre de plans binaires chiffrés. Ici, chiffrer un plan binaire équivaut à le remplacer par une image binaire aléatoire. Ceci permet de montrer que le PSNR décroît linéairement avec le nombre de plans binaires chiffrés, de poids de plus en plus grand. Même si aucun auteur ne le mentionne, la technique de chiffrement de plan binaire autorise un chiffrement par région d'intérêt, par l'utilisation d'un masque binaire adapté.

5.6.2 Techniques dans le domaine transformé

5.6.2.1 Principe des techniques appliquées aux normes de compression

Dans le cadre de la norme JPEG, Van Droogenbroeck & Benedett (2002) chiffrent les coefficients TCD, en tenant compte du codage entropique de HUFFMAN et en essayant de conserver les performances du codeur. Dans JPEG, le codeur de HUFFMAN agrège les suites de coefficients nuls en des plages de zéros. Pour atteindre l'entropie, le codeur utilise des symboles qui combinent la longueur de la plage avec une catégorie précisant la gamme de l'amplitude du premier coefficient non nul. Des codes sur 8 bits sont donnés par le codeur de HUFFMAN à de tels symboles. Ils sont suivis par les bits précisant le signe et l'amplitude du coefficient non nul. Dans leur méthode, les auteurs laissent les mots code inchangés mais chiffrent les bits ajoutés.

Leur motivation est de conserver la synchronisation fournie par les mots code tout en conservant une bonne efficacité de codage. De même, ils ne modifient pas les coefficients DC à cause de leur grande importance visuelle. De plus, en choisissant combien de bits sont chiffrés, ils obtiennent une progressivité dans le chiffrement d'image. Le chiffrement est effectué hors ligne en temps réel.

Les transformées en ondelettes peuvent également servir à chiffrer l'image, avec, éventuellement, un chiffrement à différentes résolutions. Pommer & Uhl (2002) utilisent une décomposition en paquets d'ondelettes comme SPIHT décrit par Said & Pearlman (1996). L'approche consiste alors à ne chiffrer que l'information concernant la structure en sous-bandes du *zero-tree*. Contrairement à une approche qui priviligierait le taux de compression en adaptant l'arbre à l'image, la structure des sous-bandes utilisées est aléatoire afin d'avoir une sécurité maximale. Dans ce cas, les performances en compression sont dégradées. Les mêmes auteurs exploitent la structure du flot JPEG-2000 pour fournir un chiffrement. Dans Norcen et al. (2003), ils utilisent deux codes marquant les débuts et fins de paquets pour chiffrer des paquets données sans être obligés de décoder complétement l'image. De plus, les paquets chiffrés peuvent être sélectionnés pour obtenir des effets visibles soit pour les différentes résolutions de l'image, soit pour les différents niveaux de qualité.

5.6.2.2 Techniques fondées sur la TCD

Algorithme de permutation zig-zag Tang (1996) propose une méthode fondée sur un algorithme de permutation zig-zag. L'idée principale est de remplacer le balayage zig-zag permettant de passer d'un bloc de taille 8x8 à un vecteur de taille 64 par une permutation aléatoire. Cette permutation peut être différente pour chaque bloc. Ce schéma offre une bonne sécurité, en échange d'un coût de codage plus élevé. En effet, la corrélation existant entre les coefficients parcourus dans l'ordre zig-zag est en partie perdue. Par conséquent, le codeur entropique qui suit ne permettra pas d'obtenir le même taux de compression. Toujours dans le même article, d'autres mécanismes permettent d'augmenter la sécurité de cette technique :

- Le coefficient DC codé sur 8 bits est scindé en deux moitiés de 4 bits. La moitié de poids fort remplace le coefficient DC tandis que la moitié de poids faible remplace le coefficient AC de plus petite valeur. En effet, les coefficients DC peuvent être identifiés immédiatement par leur taille révélant une approximation basses fréquences de l'image ;
- Avant leur scission, les coefficients DC de huit blocs 8x8 sont concaténés, chiffré par DES et réécrit à l'envers, octet par octet ;

– Au lieu d'utiliser une seule liste de permutations, un générateur de bits pseudo-aléatoire cryptographiquement robuste sélectionne une parmi deux liste de permutations pour chaque bloc de 8x8 pixels.

Shin et al. (1999) proposent un système très proche, pour lequel la scission des coefficients DC est remplacée par le chiffrement des bits de signe des coefficients DC par DES tandis que les coefficients DC ne sont pas sujets à la permutation.

La littérature présente quelques lacunes dans cette méthode. Elles sont décrites dans les paragraphes suivants.

Sécurité

– Attaque à texte connu et à texte chiffré choisi : les permutations sont vulnérables aux attaques à texte connu. À partir d'une image connue et de sa version chiffrée, la liste de permutation peut être retrouvée. Même si deux listes sont en jeu, celle utilisée peut être déterminée en appliquant les deux, à partir du bloc avec le plus de coefficients non nuls dans le coin supérieur gauche pris comme bloc décrypté, comme le montrent Nahrstedt & Qiao (1997) et Qiao & Nahrstedt (1998). Si un décodeur est disponible pour un attaquant, Ueraha & Safavi-Naini (2000) montrent une faille analogue face à une attaque à texte chiffré choisi ;
– Attaque à texte chiffré : cette attaque est l'attaque la plus faible disponible à l'adversaire. L'attaque de la permutation des coefficients zig-zag repose sur le fait que les coefficients AC non nuls tendent à se répartir dans le coin supérieur gauche du bloc considéré. Une fois que les coefficients non nuls sont identifiés, ils sont déplacés dans le coin supérieur gauche. Étant donné leur faible nombre, les combinaisons à explorer sont relativement peu nombreuses. À partir d'une analyse statistique, Nahrstedt & Qiao (1997) expliquent comment approcher l'image originale par cette méthode.

Baisse des performances de compression Le balayage zig-zag ordonne les coefficients par fréquence croissante et par amplitude décroissante. Il génère donc de longues plages de zéros. Or, les tables de HUFFMAN du codage entropique tiennent compte de cette propriété. Ainsi, le changement du balayage des coefficients introduit une baisse des performances de compression. Par exemple, Kailasanathan (2002) rapporte des baisses jusqu'à 20% pour JPEG.

Avec la même idée de départ, Shi & Bhargava (1998b) proposent une méthode de même nature mais non équivalente, pour le chiffrement des images

I dans MPEG. La table de HUFFMAN telle que définie dans la norme est permutée à l'aide d'une clef secrète. Elle est ensuite utilisée à la compression et à la décompression. Afin d'empêcher la dégradation des performances de compression, seules les permutations conservant la longueur des codes sont admissibles. Cependant, cette solution réduit de manière importante le nombre des permutations possibles, et donc limite la taille de l'espace des clefs et par conséquent la sécurité offerte par le système. On peut néanmoins remarquer qu'à la différence de cette approche, la permutation zig-zag entraîne éventuellement une modification du nombre de mots de la table de HUFFMAN utilisés.

Mélange des coefficients fréquentiels par bande Pour limiter la chute de l'efficacité de compression observée par la méthode des permutations zig-zag, Zeng & Lei (1999) proposent non pas une permutation à l'intérieur d'un simple bloc 8x8 pixels, mais de grouper les coefficients d'un ensemble de blocs et d'effectuer les permutations de coefficients TCD correspondants à une bande de fréquences donnée. Cette stratégie réduit le surcoût introduit par le chiffrement. Néanmoins, Zeng & Lei (2003) indiquent qu'une augmentation de la taille des fichiers de 10 à 20% est observable. Alors que les problèmes de sécurité liés à la permutation zig-zag restent valides en principe, la situation est améliorée car la taille de l'espace des clefs augmente. Par exemple, le choix des blocs utilisés pour la permutation rend plus difficile une attaque par texte chiffré uniquement.

Chiffrement scalable des coefficients Dans le cadre de JPEG, Cheng & Li (1996) proposent de chiffrer uniquement les coefficients TCD de basses fréquences et de laisser tels quels les coefficients hautes fréquences, ceci pour JPEG. Cette approche correspond donc à un chiffrement d'image partiel. Les auteurs mentionnent eux-mêmes que la sécurité de ce mécanisme est discutable. En effet, en appliquant cette méthode à tous les blocs d'une image, il persiste néanmoins de l'information concernant les contours présents. Kunkelmann & Reinema (1997); Kunkelmann & Horn (215–218) appliquent cette idée pour les coefficients TCD issus de MPEG, en utilisant DES ou IDEA pour faire le chiffrement. Les auteurs suggèrent d'utiliser plus ou moins de coefficients pour les images I, P ou B. Prendre plus ou moins de coefficient permet d'avoir un aspect scalable en sécurité. Cette technique pose la question de la conformité des coefficients après chiffrement. En effet, leur amplitude doit rester telle que le flot binaire reste décodable.

Wu & Kuo (2000, 2001) indiquent également le même problème de sécurité en montrant des exemples visuels et en proposant une méthode pour

récupérer les informations sur les contours à partir des données chiffrées. Les auteurs précisent que la compaction de l'énergie sur quelques coefficients, résultat de la transformation décorrélante, n'implique pas nécessairement que cela est vrai pour l'intelligibilité de l'image. Ainsi, les coefficients de faible énergie contiennent-ils encore suffisamment d'information pour rendre l'image intelligible. Néanmoins, même si cette remarque s'observe pour les techniques de chiffrement partiel des coefficients TCD, elle sera moins vraie pour les approches par ondelettes qui présentent d'autres caractéristiques vis-à-vis de ce problème.

Chiffrement du bit de signe des coefficients Zeng & Lei (1999, 2003) suggèrent de ne chiffrer que le bit de signe de chaque coefficient TCD, ce qui revient à faire un chiffrement partiel. En effet, la séquence des bits de signe présente une forte entropie, et, par conséquent, une hausse supplémentaire de l'entropie due au chiffrement n'est pas attendue. Ceci permet de réduire l'impact du chiffrement sur les performances de compression. Appliquée sur des coefficients TCD issus d'un codec H-263, cette méthode a même réduit la taille du flot binaire généré. Shin et al. (1999) propose également d'ajouter à la permutation zig-zag, déjà vue précédemment, le chiffrement des bits de signes des coefficients DC.

Shi & Bhargava (1998a); Bhargava et al. (2004) proposent de changer aléatoirement les bits de signes de tous les coefficients TCD, en appliquant cette méthode directement sur le flot binaire. En effet, les bits de signe sont souvent codés à part des amplitudes de coefficients codés avec une table de HUFFMAN.

La réduction de la charge de calcul par rapport à un chiffrement complet est significative car seulement de 13 à 20% de toutes les données doit être chiffré. Le chiffrement des bits de signe peut se faire de plusieurs façons : une possibilité est d'effectuer un OU-exclusif entre les bits de signe et les bits issus d'un flot binaire fourni par une source cryptographiquement sûre. Une autre possibilité est d'utiliser un chiffrement par bloc, en regroupant les bits de signe. Cependant, dans les deux cas, Wu & Kuo (2000, 2001) montrent des exemples d'attaques produisant des résultats visuels très bons et donc rendent discutables l'utilisation de telles méthodes.

5.6.2.3 Transformée de FOURIER à domaine secret

Cette approche diffère beaucoup des précédentes car elle consiste en la dissimulation du domaine vers lequel l'image est transformée dans le schéma de compression. L'idée sous-jacente est que si l'attaquant ne connait pas le domaine transformé, il lui sera impossible de décoder l'image. Pour cela, il

faut utiliser une transformation prenant un paramètre indiquant le domaine transformé. Almeida (1994) introduit une telle transformation, la transformation de Fourier fractionnaire. Djurovic et al. (2001) l'ont utilisée pour insérer une marque dans un domaine transformé secret. De même, Unnikrishnan & Singh (2000) proposent d'utiliser cette transformation pour chiffrer des données visuelles. Plus précisement, le domaine image, le domaine chiffré et le domaine sortie sont liés les uns aux autres par une transformée de Fourier fractionnaire. L'article discute en détails la taille de l'espace des clefs disponibles et la sécurité qui en découle, sans préciser la complexité de calcul de la méthode. Les auteurs suggèrent également une mise en œuvre à l'aide de méthodes de FOURIER optiques. Néanmoins, il semble que la charge de calcul rende difficile la réalisation numérique de cette technique. De plus, aucun aspect de compression n'est pris en compte. Cependant, cette approche présente un attrait théorique évident, avec une utilité restreinte à des applications spécifiques.

5.6.2.4 Codage entropique secret

Cette technique repose sur le fait qu'un codeur entropique et un procédé de chiffrement transforment les données originelles en un flot binaire sans redondance qui ne peut être décodé ou déchiffré sans information additionnelle. Wu & Kuo (2000) discutent l'idée de changer un codeur entropique en outil de chiffrement. Dans le cas du codeur entropique, l'information requise est le modèle statistique, tandis que pour le chiffreur, c'est la clef. En effet, il est très difficile, voire impossible, de décoder un flot binaire codé entropiquement avec un code à longueur variable de HUFFMAN sans la connaissance de la table utilisée. En considérant la table de HUFFMAN comme étant la clef, on obtient là une technique de chiffrement. Les auteurs proposent de choisir une table parmi plusieurs, avec une sélection aléatoire reposant sur une clef. Néanmoins, pour éviter de dégrader les performances de compression, les différentes tables utilisées doivent être calculées à partir d'ensembles d'images d'apprentissage différents. Des techniques de mutation d'arbres de HUFFMAN peuvent aussi permettre de générer des tables différentes à partir de 4 tables originales. L'analyse des auteurs montre que leur approche est vulnérable face à une attaque à texte clair choisi mais pas face à une attaque à texte clair connu et à texte chiffré uniquement. Dans des travaux plus récents, Wu & Kuo (2001) suggèrent d'améliorer la sécurité de leur approche en insérant des bits additionnels à des positions aléatoires, en le faisant de manière différente suivant la position dans le flot de données.

Sur le même principe, Shi & Bhargava (1998b) proposent une permutation des mots codes avec un espace de clef réduit. Des travaux plus anciens existent également pour des codeurs arithmétiques.

La principale critique de ces méthodes vient des baisses de performances que l'on observera. En particulier, si l'image à coder et chiffrer possède un contenu très éloigné de la base d'images ayant servi à générer les différentes tables de codes à longueurs variables, les performances de compression seront forcément dégradées. Cet effet néfaste peut limiter l'emploi de cette technique de chiffrement.

5.6.3 Techniques orientée flot binaire

5.6.3.1 Chiffrement d'entêtes

Chiffrer l'entête de fichier d'image est la manière la plus directe de chiffrer l'image elle-même. La conformité du flot binaire est immédiatement perdue et l'image ne peut être décodée de manière usuelle. Cependant, les attaques contre cette manière de chiffrer reposent sur l'évaluation des données de l'entête directement à partir de la syntaxe du flot binaire. Par conséquent, la sécurité d'un tel mécanisme dépend de la nature des données d'entête chiffrées. La méthode sera donc sûre si l'analyse des données restantes ne permet pas d'estimer les données chiffrées. Dans tous les cas, le chiffrement de données d'entête est une approche intéressante nécessitant peu de ressource de chiffrement.

5.6.3.2 Permutations

Les permutations à l'intérieur d'un flot binaire forment une classe de chiffrement bien adaptée au chiffrement d'image. Cependant, elles sont sensibles à une attaque à texte clair connu, comme vu à la section 5.6.2.2. Les méthodes existantes diffèrent sur la nature des données permutées.

Octets Qiao & Nahrstedt (1998) discutent un algorithme de permutation pure pour lequel des octets du flot binaire sont permutés. Suivant le degré de sécurité souhaité, la taille de la liste de permutations varie. C'est une approche très rapide tant pour le chiffrement que pour l'analyse du flot binaire à chiffrer. L'inconvénient majeur est que la conformité du flot binaire est perdue.

Mots du code à longueur variable Prenant comme point de départ le mélange de coefficients dans les bandes de fréquences de Zeng & Lei (1999,

2003) et Wen et al. (2002) proposent de mélanger les mots du codage des longueurs de plage correspondants aux coefficients DC non nuls et isolés. Les mots de différents blocs de 8x8 pixels sont regroupés selon leur indice et permutés à l'aide d'une liste de permutations. Le nombre de groupes et la plage des mots de codes dans un groupe peuvent être ajustés aux contraintes de sécurité. La principale difficulté de la méthode est que les blocs 8x8 diffèrent dans le nombre de coefficients DC non nuls et isolés. La solution réside dans le contrôle du dernier champ de chaque bloc. La conformité du flot binaire peut être garantie par la troncature de mots de code en cas de dépassement des 64 coefficients par bloc 8x8. Le surcoût de traitement, lié à l'identification et au regroupement des mots du code à longueur variable, est plus important que pour la méthode de permutation pure. De manière indépendante, Kankanhalli & Hau (2002) utilisent la même idée, augmentant la sécurité du système en inversant de manière aléatoire le dernier bit des mots et en appliquant des corrections si le préfixe du code qui suit est modifié.

Blocs Dans le même article, Kankanhalli & Hau (2002) discutent également de la permutation de blocs 8x8. Dans ce cas, la conformité du flot binaire résultant est garantie. Néanmoins, il reste un important problème de sécurité. En effet, cette méthode revient à permuter des blocs images dans le domaine spatial et il est communément admis qu'une telle méthode est vulnérable à une attaque à texte chiffré seul. Il suffit, par exemple, de regrouper les blocs avec des frontières correspondantes ou similaires pour obtenir une bonne approximation de l'image.

Amélioration de la sécurité des permutations La vulnérabilité des schémas de compression reposant sur les permutations face aux attaques à texte clair connu est contrée par le changement fréquent des permutations utilisées. Ceci ajoute une charge de travail significative aux mécanismes de gestion et de distribution de clefs. Ceci peut être coûteux (en temps et en puissance de calcul) car mettant en œuvre des chiffrements à clef publique. Pour éviter cette charge, Wen et al. (2002) proposent de générer les listes de permutations *au vol* à partir de bouts du flot binaire non impliqués dans les permutations. Par exemple, si les mots du code à longueur variable sont permutés, les bits de signe des coefficients TCD chiffrés par DES peuvent être utilisés. En effet, les bits de signe ne sont pas permutés de par leur entropie importante. Une mise en œuvre peut être comme suit :

- chiffrer les bits de signe des coefficients TCD avec la clef K_F ;
- générer une séquence binaire aléatoire R_L de longueur L (avec un chiffrement par flot de clef K_L), avec L de taille suffisante pour chiffrer

tous les mots du code à longueur variable. Si K est le nombre de mots du code, la contrainte $L > K \log_2(K)$ est une bonne estimation ;
- pour chaque ensemble de mots à permuter, concatener les bits de signe chiffrés en un flot R' ;
- R' est chiffré avec K_F pour obtenir R, qui est répété pour obtenir un flot R_r de même longueur que le flot R_L ;
- calculer $R_C = \overline{R_L + R_r}$, application d'un ou-exclusif ;
- R_C est découpé en K segments, chacun de longueur b_L, b_L étant la longueur moyenne d'un mot du code, soit environ $\log_2(K)$. La liste de permutation fait le lien entre les indices i, avec $0 \leq i \leq K-1$ et le i-ème segment de R_C.

Il est intéressant de noter que ce mécanisme possède des propriétés bénéfiques cachées. En effet, dans le domaine de la transmission video sans fil, Tosun & Feng (2001b) montrent que les permutations peuvent servir de support pour un schéma de réduction de l'effet des erreurs. L'idée est qu'après permutation, les données transmises de manière erronées voient leur influence réduite à la longueur du segment dans lesquelles elles se produisent. Il n'y a plus d'effet d'avalanche des erreurs. Les permutations apparaissent alors comme des marques de synchronisation. Ceci est une propriété importante dans la transmission sans fil, dont le taux d'erreur binaire est élevé. De plus, la localisation des erreurs aidant, il est plus facile à un mécanisme de récupération des erreurs de fonctionner.

5.6.3.3 Masque jetable

L'application du masque jetable à un flot binaire revient à scinder le flot en deux. Une première moitié, chiffrée, sert de masque jetable pour le chiffrement de la seconde. Qiao & Nahrsted (1997); Qiao & Nahrstedt (1998) appliquent cette idée sur des flots MÊG, considérés octet par octet. Ceci est parfaitement transposable à un flot binaire JPEG. Ici, la scission du flot se fait en séparant d'un côté les octets numérotés pairs des octets numérotés impairs pour obtenir deux listes, une dite paire, une dite impaire. La liste paire est chiffrée en utilisant un chiffrement de flot, robuste, comme DES, proposé par les auteurs. Cette liste chiffrée sert ensuite de masque pour une opération de ou-exclusif avec la liste impaire. L'existence d'une faible corrélation, confirmée expérimentalement, entre octets et paire d'octets dans le flot binaire MPEG garantit la sécurité du mécanisme. Cette faible corrélation vient directement de l'utilisation d'une transformation qui décorrèle les coefficients et fournit une compaction de l'énergie. Pour augmenter la sécurité de la méthode, les auteurs proposent les améliorations suivantes :

- la sélection fixe des octets pour déterminer les listes *paire* et *impaire* est remplacée par l'utilisation de deux listes aléatoires générées à l'aide d'une clef de 128 à 256 bits ;
- chaque ensemble de 32 octets est permuté à l'aide de 8 différentes listes de permutations utilisées dans un ordre prédéterminé.

L'idée sous-jacente de ces améliorations est d'augmenter la taille de l'espace de recherche d'une éventuelle attaque brutale. Ainsi, cet algorithme présente une grande sécurité, pour un coût de calcul élevé. Les auteurs indiquent que ce dernier représente 47% d'un chiffrement DES de l'ensemble du flot binaire. De plus, la méthode opérant au niveau octet, la sémantique du flot binaire est complétement perdue et il ne respecte plus la syntaxe prévue par la norme de compression.

Tosun & Feng (2001a) suggèrent d'appliquer cette idée récursivemenet pour diminuer la charge de chiffrement par moitié à chaque récursion. La liste *paire*, normalement sujette au chiffrement dans la méthode originale, est à son tour scindée en deux listes, une liste *impaire2* et une liste *paire2*, listes qui sont combinées avec un ou-exclusif. Ensuite, la liste *paire2*, dont la taille est égale au quart des données originales, est soit chiffrée, soit à nouveau scindée en deux. Cette stratégie réduit bien l'effort de chiffrement au détriment de la sécurité, qui est plus faible que précédemment, la méthode n'étant plus une technique de masque jetable, comme le font remarquer Lookabaugh et al. (2003). Évidemment, comme pour la méthode précédente, la conformité du flot binaire est perdue.

5.6.3.4 Chiffrement d'octets par destruction

Griwotz (1998); Griwotz et al. (1998) proposent de détruire aléatoirement des octets dans le flot binaire, en l'occurence MPEG[3], pour une distribution sans contrôle, tandis que les données manquantes sont chiffrées et transmises pour les utilisateurs légitimes. Cela revient en fait à chiffrer des octets à des positions aléatoires. Les auteurs montrent que le chiffrement d'un pour-cent des données est suffisant pour rendre la video au moins non regardable voire non décodable. Néanmoins, il manque une cryptanalyse de la méthode. Par exemple, la répartition des destructions est importante. Le pire cas apparait lorsque seules les données de l'entête sont modifiées. En effet, dans ce cas, il est possible de reconstruire l'entête du fichier à partir des données restantes. De plus, le scenario d'utilisation de la méthode proposé ne s'intéresse qu'à la lecture de la video sur un lecteur conforme et ne tient pas compte d'éventuelles attaques. Ainsi, pour garantir un niveau de sécurité correct, un

[3]Cette méthode est extensible au cas des images fixes.

pourcentage plus élevé de données doivent être chiffrées et les positions des octets détruits doivent être soigneusement choisies. Évidemment, la quantité de données à transmettre à l'utilisateur autorisé devient grande et cela complique inévitablement la gestion des clefs.

Wen et al. (2002) décrivent une approche plus générale, toujours pour de la video, toujours extensible aux cas des images fixes, appelée *Syntax Unaware Runlength-based Selective Encryption*. L'idée est de chiffrer d'abord X bits, de laisser les Y bits suivants en clair, de chiffrer les Z bits suivants, et ainsi de suite. Il y a donc une information concernant le chiffrement, et une autre concernant les tailles des ensembles de bits qui seront chiffrés successivement. Les propriétés de la méthode précédente persistent, avec également la destruction de la conformité du flot binaire.

5.6.3.5 Chiffrement de mots-code

La section 5.6.3.4 l'a montré, les chiffrements d'octets dans les flots binaires cassent la conformité de ceux-ci au regard des schémas de compression. Par opposition, le chiffrement des mots-code du code à longueur variable tient lui compte de cette conformité. Ceci amène une difficulté. Si les mots-code sont chiffrés, il est nécessaire que la concaténation des mots-codes chiffrés restent un flot binaire valide. Par exemple, prenons les mots-code 0, 10, 110 et 111. La séquence 010 est une séquence valide et décodable. Son chiffrement peut être 001, qui n'est plus un flot binaire conforme au code à longueur variable. Wen et al. (2002) proposent une solution simple à ce problème, en utilisant une table d'indices. L'idée n'est pas de chiffrer les mots-code, mais de chiffrer leur position dans la table. Ces indices de position ont tous la même taille et pour des tables ayant une puissance de 2 comme dimension, si l'on chiffre un indice de position, on retrouve une position valide dans la même table. Il suffit donc de remplacer le mot-code d'indice i par le mot-code correspondant à l'indice chiffré i'. Le résultat sera une concaténation différente de mots-code valides. Si la table n'est pas de taille une puissance de 2, la méthode est toujours utilisable en décomposant la table en sous-tables de taille une puissance de 2 plus petite. Le flot binaire obtenu est donc toujours conforme mais il présente une variation légère en taille. En effet, le nombre de mots-code est conservé mais leur taille diffère. Les auteurs indiquent par exemple des variations de taille jusqu'à 9% en sus. La charge de calcul pour le chiffrement inclut la détermination des mots-code, la construction de la table des indices, leur chiffrement, et la réinsertion des données dans le flot binaire.

5.6.4 Comparaison des différentes techniques de chiffrement d'images

Le tableau 5.1 présente maintenant les aspects clefs des différentes techniques de chiffrement présentées dans cet état de l'art. Il apparaît qu'il n'existe pas de méthode dont les propriétés soit bonnes dans toutes les dimensions. Certains aspects, la compression ou la compatiblité des flots binaires générés, sont quelques fois laissés de côté au profit des aspects de sécurité pure. Il reste donc encore des pistes à explorer, en particulier celles concernant le chiffrement effectué au niveau du codeur, qui permettent de conserver une compatibilité de flots avec un surcoût espéré faible. La section 5.7 présente à ce propos l'effort de normalisation de la sécurité des flots JPEG-2000 qu'est JPSEC.

5.7 Normes de sécurité pour les images

La norme JPEG-2000 est connue surtout pour ses aspects de compression d'image et les aspects de scalabilité qui sont présentés. Néanmoins, la partie 8 de la norme s'intéresse à la sécurité des images et introduit JPSEC. La présente section présente les objectifs de la norme, précise le *framework* utilisable et indique quelques techniques appliquées.

La partie sécurité de JPEG-2000 défini un *framework* pour fournir des services de sécurité comme la confidentialité, le contrôle d'accès, la protection de l'intégrité, l'authentification et la protection de la propriété des images. Cet objectif est atteint en utilisant des techniques de chiffrement, de génération et gestion de clefs, de génération et vérification de signatures numériques, de brouillage et de tatouages. De plus, un des objectifs de JPSEC est de faire qu'un flot binaire JPSEC, c'est-à-dire un flot binaire JPEG-2000 protégé par JPSEC, peut toujours être utilisé comme un flot JPEG-2000 ordinaire. Ainsi, un flot JPSEC va conserver la structure riche d'un flot JPEG-2000 comme les tuiles, les couches, les flots d'améliorations de la résolution, par exemple.

On le voit, l'objectif principal de la norme est de fournir des services de sécurité tout en garantissant le respect de la compatibilité des flots binaires générés. À ce propos, JPSEC est devenu la norme internationale ISO/IEC 15444–8 en juin 2006. Quelques techniques adaptées commencent à voir le jour. Zhang et al. (2004) proposent l'intégration d'une méthode de chiffrement et de tatouage dans JPEG-2000, tandis que Fang et al. (2006) se concentrent sur le chiffrement de sous-flots binaires. Enfin, Deng et al. (2005) et Wu et al. (2004) s'intéressent à la fois à la sécurité des images et à leur dissémination.

méthode	$D(E)$	$D(P)$	sécurité	conformité binaire	traitement de flot	influence débit distorsion
Permutation	faible	nulle	faible	oui	non	oui
mélange de coefficients par bande	faible	faible	faible	oui	non	modéré
chiffrement scalable	moyenne et scalable	0	faible et scalable	oui	oui	non
chiffrement des bits de signe	moyenne	moyenne	faible	oui	oui	non
TF à domaine secret	élevée	élevée	élevée	non	non	oui
codage entropique secret	faible	nul	élevée	non	non	modérée
chiffrement d'entête	faible et scalable	faible	faible et scalable	non	oui	non
Permutation d'octets	faible	0	faible	non	oui	non
Permutations de mots-code	faible	moyenne	faible	oui	oui	non
Permutations de blocs et macroblocs	faible	faible	faible	oui	oui	non
masque jetable	élevé	faible	élevé	non	oui	non
destruction d'octets	moyenne et scalable	faible	faible et scalable	non	oui	non
chiffrement de mots-code	moyenne et scalable	élevé	moyenne et scalable	oui	oui	modérée

TAB. 5.1: Tableau comparatif des propriétés des différentes techniques de chiffrement d'images présentées à la section 5.6

5.8 À retenir

Le chiffrement d'image introduit une grande variété de solutions innovantes à la fois d'un point de vue conceptuel, avec le chiffrement sélectif et le chiffrement partiel, et d'un point de vue technique, avec un foisonnement des méthodes. Celles-ci combinent souvent des aspects de partitionnement des données pour choisir celles qu'il faudra effectivement chiffrer et des aspects de chiffrement à proprement parler utilisant des primitives de chiffrement. De même, l'évaluation des performances prend en compte la conformité des flots binaires, conformité plus difficile à atteindre eu égard à la complexité croissante des normes de compression et de la description de flot binaires. Il apparait ainsi que les bonnes méthodes de chiffrement reposent sur l'exploitation fine de la structure des codeurs utilisés et que cet emploi joint permet d'obtenir des performances de sécurité bonnes avec un coût calculatoire réduit.

Cette piste sera suivie par la suite. En effet, dans le chapitre suivant, nous présentons une méthode de chiffrement à coût nul, avec un bon niveau de sécurité, tandis que dans le chapitre 7 nous introduisons une méthode de chiffrement sélectif du flot binaire LAR IS+P.

E. P. Jacobs , *Le mystère de la grande pyramide, tome 1*, page 22, case 14.

6

Chiffrement partiel à coût nul

Plan

6.1 Principe du chiffrement à coût nul

6.2 Du nombre de partitions LAR

6.3 Du passage d'un niveau au suivant dans la partition LAR

6.4 Discussion des résultats

6.5 Sécurité

6.6 À retenir

Parmi les méthodes de chiffrement d'images présentées dans le chapitre 5, certainent pratiquent par chiffrement des entêtes des fichiers contenant les données compressées. La section 5.6.3.1 montre une telle technique. La motivation de ce type de solution repose sur le fait qu'il est difficile de reconstruire l'image en l'absence de ces données partielles mais cruciales.

Une extension de cette idée pour les images compressées par les codeurs LAR utilise l'importance de la partition quadtree dans le processus de compression. En effet, la section 3.3.1 a montré la difficulté de reconstruire l'image sans la connaissance de la partition quadtree construite selon l'activité dans l'image.

Ce chapitre présente donc un mécanisme de chiffrement partiel d'images obtenu en enlevant tout ou partie des données relatives à la partition quadtree. Après une présentation de principe et l'énoncé des principales questions soulevées par cette idée, nous verrons des éléments théoriques propres à étayer cette méthode. Ensuite, des évaluations expérimentales valident les résultats théoriques. Motsch et al. (2006a,b) introduisent les concepts utilisés dans ce chapitre.

6.1 Principe du chiffrement à coût nul

La partition LAR peut être utilisée pour masquer le contenu d'une l'image codée, éventuellement par niveau. Le schéma 6.1 détaille le principe de cette méthode. L'idée est d'ôter du flot binaire tout ou partie du flot correspondant à la partition LAR.

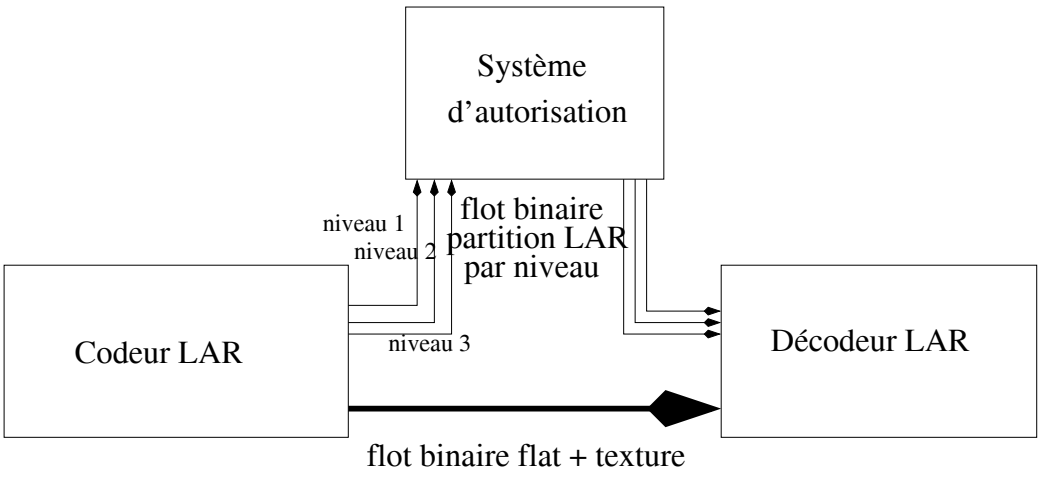


FIG. 6.1: Proposition de schéma de masquage d'image par niveaux

Cette méthode diffère des techniques de chiffrement habituellement proposées. Le schéma de masquage intégré au codeur LAR permet d'autoriser la récupération d'une image à différents niveaux de résolution. Ainsi, l'image peut être reconstruite au niveau le plus haut, et l'accès aux meilleures résolutions peut être contrôlé de manière fine. Quant à la transmission de la partition quadtree, elle se fait en utilisant un chiffrement à clef publique, par exemple.

Le principe du chiffrement est donc simple : il s'agit de ne pas transmettre tout ou partie de la partition. Justifier l'intérêt de cette méthode revient à démontrer que la reconstruction de l'image sans connaître la partition LAR est difficile. Ainsi, plusieurs éléments sont à considérer. Comme pour toutes les méthodes de chiffrement, il faut d'abord montrer que dans le cas d'une attaque *par force brute*, le nombre de combinaisons à essayer est tel que la durée de l'attaque est trop grande pour être réalisée en pratique. Ensuite, il s'agira d'évaluer la possibilité d'extraire des données (hors partition) permettant de reconstruire la partition. Enfin, il faudra mener une investigation des potentiels défauts dans le mécanisme même qui empêcheraient le chiffrement d'être sûr.

Les interrogations précédentes peuvent se décliner en quelques pistes d'investigations :

- montrer que le nombre de partitions quadtree est grand ;
- montrer que, connaissant la partition à un niveau de résolution donné, il est difficile de trouver celle du niveau suivant ;
- montrer que la corrélation restant entre le flot binaire de la partition et les flots binaires de l'image *flat* et de texture est faible.

La section 6.2 se penche sur le nombre de partitions possibles. Le passage d'un niveau à un autre est étudié en section 6.3. Enfin, la sécurité intrisèque de la méthode est brièvement discutée à la section 6.5.

6.2 Du nombre de partitions LAR

6.2.1 Approche théorique

Soit une image I de M lignes et N colonnes avec $S = M \times N$ pixels. Chaque pixel est représenté par q bits et peut donc prendre $Q = 2^q$ valeurs différentes.

$\mathrm{QP}^{\lfloor L_T..L_B \rfloor}(I)$ est une partition quadtree de l'image I avec des tailles de blocs allant de $L_T \times L_T$ pour le niveau le plus haut (premier niveau), à $L_B \times L_B$ pour le niveau le plus bas (dernier niveau). $\left|\mathrm{QP}^{\lfloor L_T..L_B \rfloor}(I)\right|$ est le nombre de partitions quadtree par blocs possibles pour une image I.

Soient $l_T = \log_2 L_T$, $l_B = \log_2 L_B$ et $l = l_T - l_B + 1$ le nombre de niveaux dans la partition quadtree, ou partition LAR. On note $\Omega^{[L_T..L_B]}$ le nombre de partitions LAR possibles sur un bloc de taille $L_T \times L_T$. $\omega^{[L_T..L_B]} = \log_2 \Omega^{[L_T..L_B]}$ est donc le nombre de bits nécessaires pour coder cette partition quadtree.

Pour illustrer notre propos, nous considérons une image 256×256 sur 8 bits avec une partition LAR $\mathrm{QP}^{[32..2]}$ sur 5 niveaux.

Le nombre de partitions quadtree $\Omega^{[L_T..L_B]}$ est lié à la fonction récursive $\Phi(n)$. Cette fonction est définie récursivement comme suit :

$$\begin{cases} \Phi(0) = 1 \\ \Phi(n) = 1 + \Phi^4(n-1) \end{cases} \tag{6.1}$$

En fait, $\Omega^{[L_T..L_B]} = \Phi(l-1)$. Le tableau 6.1 donne les premières valeurs de $\Phi(n)$. Il est à remarquer que $\log_2 \Phi(n)$ est le nombre de bits nécessaires pour représenter la valeur de $\Phi(n)$. Pour $n > 0$, $\Phi(n)$ peut être approchée grossièrement par :

$$\Phi_a(n) = 2^{(4^{n-1})} \tag{6.2}$$

Si toutes les partitions étaient équiprobables, ceci établirait un coût de codage du quadtree $C(\mathrm{QP}^{[L_T..L_B]})$ à 4^{l-2} bits par bloc $L_T \times L_T$.

n	$\Phi(n)$	$\log_2 \Phi(n)$	$\Phi_a(n)$
0	1	0	1
1	2	1	2
2	17	4.09	2^4
3	83522	16.35	2^{16}
4	4.86×10^{19}	65.40	2^{64}

TAB. 6.1: $\Phi(n)$ et $\Phi_a(n)$ pour les premières valeurs de n

Pour notre exemple, $\Omega^{[32..2]} = 4.86 \times 10^{19}$ et il faut jusqu'à 65.40 bits pour coder la partition quadtree sur un bloc 32×32.

Le nombre de partitions LAR pour l'image entière I est simplement calculé en considérant que chaque bloc de taille $L_T \times L_T$ contient une partition quadtree $\mathrm{QP}^{[L_T..L_B]}$. $\left|\mathrm{QP}^{[L_T..L_B]}(I)\right|$ est alors donné par :

$$\left|\mathrm{QP}^{[L_T..L_B]}(I)\right| = \Phi(l-1)^{\frac{MN}{L_T^2}} \tag{6.3}$$

Le coût de codage est souvent exprimé en bit par pixel (bpp). Ce coût $c(QP)$ est donné par :

$$c(\mathrm{QP}^{[L_T..L_B]}) = \frac{\log_2 \Phi(l-1)}{L_T^2} \tag{6.4}$$

Si on utilise l'approximation $\Phi_a(l-1)$, il vient la relation approchée :

$$c(\mathrm{QP}^{[L_T..L_B]}) \approx \frac{1}{4L_B^2} \tag{6.5}$$

Ainsi, le coût de codage approché ne dépend que de la taille des plus petits blocs de la partition quadtree.

En continuant avec notre exemple, $\left|\mathrm{QP}^{[32..2]}(I)\right| \approx 2^{4186}$. Cela indique que le coût de codage sera au maximum de 4186 bits soit 6.39 cbpp[1].

6.2.2 Validation expérimentale

Les images utilisées pour l'expérimentation sont *lena*, *airplane*, *baboon*, *goldhill*, *man*, *pepper* et *woman*, toutes de taille 512×512 et codées sur 8 bits. Les partitions LAR sont de type $\mathrm{QP}^{[64..2]}$ soit au moins 6 niveaux dans la pyramide. Dans ce cas, l'entropie maximale pour la partition LAR est de 6.387 cbpp.

La figure 6.2 montre l'influence du seuil d'activité sur l'entropie de la partition LAR. En effet, ce seuil pilote le processus de découpage en quadtree. Plus précisement, lorsque le seuil augmente, le nombre de blocs diminue, avec des blocs de taille plus grande. Par ailleurs, Déforges (2004) s'est intéressé à trouver l'influence du seuil pour obtenir un bon compromis entre coût de codage de la partition quadtree et coût de codage de l'image. Il y apparaît que le meilleur compromis pour des images naturelles correspond à une valeur de seuil autour de 30, sur une échelle de 256 niveaux de gris. Par conséquent, la valeur du seuil est fixée par défaut à 30.

Pour les images de l'expérimentation, l'entropie de la partition LAR reste au-dessus de 3 cbpp (environ la moitié du coût de codage) pour une large plage de seuils. Par exemple, pour une image de taille 512×512, l'entropie de la partition LAR est d'environ 7864 bits pour toute l'image. Ainsi, le nombre de partitions LAR possible est de l'ordre de $2^{7864} \approx 10^{2367}$. Une attaque par brute force ne serait pas envisageable.

À l'inverse, si la protection de l'image devait durer 100 ans, c'est-à-dire $3.156 \times 10^9 s$, et qu'un décodage ne durait qu'une microseconde, l'entropie

[1] centième de bit par pixel.

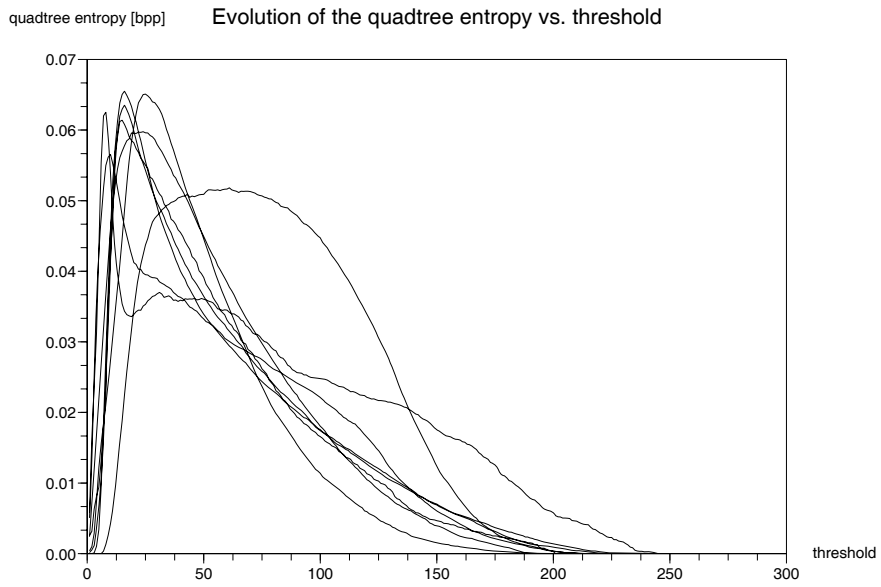


FIG. 6.2: Entropie de la partition LAR en fonction du seuil d'activité

que devrait avoir la partition LAR ne devrait être que de 51.5 bits, soit 0.02 cbpp, toujours pour une image 512×512.

La figure 6.3 montre l'influence du nombre de blocs sur l'entropie de la partition LAR. Le graphe a l'allure de la fonction entropie binaire $H_2(p)$ comme sur la figure 6.4. Cela démontre bien que l'entropie de la partition LAR est maximale lorsque le nombre de blocs est égal à la moitié du nombre maximal de blocs possible.

Comme la partition LAR est transmise progressivement comme vu à la section 3.1, l'entropie de chaque niveau de la partition est calculée et le tableau 6.2 présente les résultats obtenus lorsque le seuil vaut 30. L'entropie du premier niveau est nulle en l'absence de blocs de taille 64×64 dans la partition LAR. Le second niveau possède une entropie d'environ 50 bits qui peut être suffisante pour fournir une protection raisonnable de l'image. De plus, l'entropie augmente avec la diminution de la taille des blocs, rendant la reconstruction de l'image sans la partition LAR plus difficile. L'image *airplane* fait exception, les 17 bits d'entropie étant insuffisants. Dans tous les cas, le niveau le plus haut de la partition LAR peut être décodé, ne fournissant qu'une image de très basse résolution avec une grande distorsion. En effet, une partition $\mathrm{QP}^{[64..2]}$ ne donnera qu'une imagette 8×8 à partir d'une image 512×512.

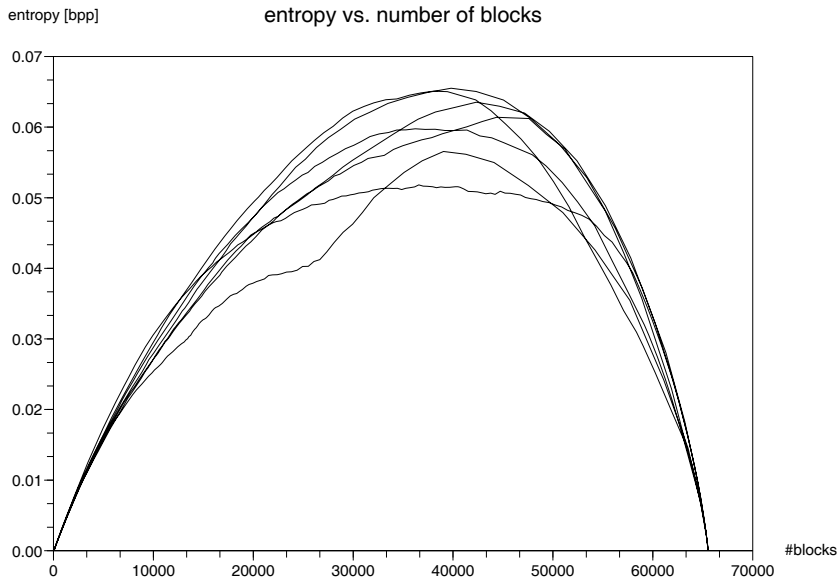


FIG. 6.3: Entropie de la partition LAR en fonction du nombre de blocs

image	taille des blocs				
	64×64	32×32	16×16	8×8	4×4
lena	0	61	656	2916	8886
airplane	17	175	601	1922	7243
baboon	0	0	21	1647	10356
goldhill	0	65	218	2659	12218
man	0	35	466	2824	11014
pepper	0	41	667	3148	8245
woman	0	86	560	2810	9483

TAB. 6.2: Entropie des niveaux de la partition LAR en [bits] pour des images de 512×512 en fonctions des tailles de blocs.

6.2.3 Analyse

La section 6.2.1 montre que le nombre de partitions quadtree sur une image est très grand. Néanmoins, deux remarques viennent nuancer ce résultat.

La première concerne le résultat de l'équation (6.3). Ce résultat repose sur l'hypothèse que les blocs d'une image sont indépendants statistiquement. Ce qui implique que le coût de codage fourni par (6.2) est un coût maximal. La présence d'une corrélation entre blocs fera baisser l'entropie du flot binaire généré et donc le coût de codage. L'expérimentation présentée montre que l'entropie restante est toujours importante.

La deuxième remarque indique que l'espace des images est toujours beaucoup plus grand que l'ensemble des partitions LAR. Dans l'exemple utilisé, chaque pixel est représenté par 8 bits, mais le coût maximum de codage de la partition LAR est de 6.4 cbpp. Cette remarque pose le problème de l'utilisation de la partition LAR comme identifiant unique d'une image, comme clef de hachage par exemple. Cette utilisation ne serait justifiée que si le nombre d'images partageant la même partition quadtree était réduit.

6.3 Du passage d'un niveau au suivant dans la partition LAR

Il s'agit de répondre à la question suivante : *connaissant la partition* LAR *à un niveau donné, sachant le nombre de blocs à décomposer, combien y-a-t'il de partitions* LAR *de niveau suivant ?*

La partition initiale, en blocs de taille maximale $L_T \times L_T$ est toujours la même. Il y a $b = \frac{MN}{L_T^2}$ blocs différents. Parmi ces blocs, d seront découpés en blocs de taille inférieure. Le nombre de partitions de passage, P_{L_T} est donc une combinaison :

$$P_{L_T} = C_b^d \approx 2^{bH_2(\frac{d}{b})}$$

avec

$$H_2(p) = -p\log_2(p) - (1-p)\log_2(1-p) \tag{6.6}$$

l'entropie binaire. La figure 6.4 présente le tracé de $H_2(p)$. En tenant compte de $0 \leq H_2(x) \leq 1$, de $0 \leq \frac{d}{b} \leq 1$ et $H_2(x)$ maximale pour $x = 0.5$, il vient :

$$1 \leq P_{L_T} \leq 2^b$$

et P_{L_T} maximale pour $d = \frac{b}{2}$. En effet, si aucun ou tous les blocs sont à découper, il n'y a aucune difficulté à retrouver la partition suivante. En

revanche, si le nombre de blocs à découper est proche de la moitié des blocs, le nombre de partitions possibles est de l'ordre de 2^b. On constate également que si le nombre de blocs à découper est compris en 10 et 90 %, le nombre de partitions possibles est d'environ 2^{b-1} qui reste grand.

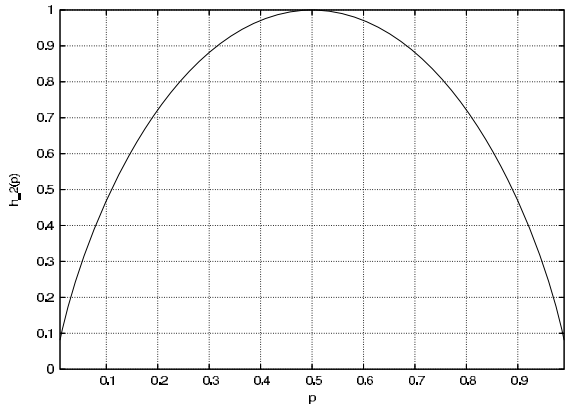


FIG. 6.4: Fonction entropie binaire $H_2(p)$.

En reprenant notre exemple canonique, $b = 512^2/32^2 = 256$ et $1 \leq P_{32} \leq 2^{256}$. Ainsi, l'espace de recherche pourrait être aussi grand que celui correspondant à une clef de chiffrement de 256 bits.

Un calcul similaire peut être conduit pour déterminer le nombre de possibilités de transition entre deux niveaux quelconques de la partition LAR.

Le tableau 6.3 montre le nombre P_{32} de partitions LAR possibles à partir du décodage du niveau de taille 32×32. Mis à part l'image `baboon` qui présente beaucoup de petits blocs, comme l'avait montré le tableau 6.2, P_{32} est assez élevé pour toutes les autres images.

image	P_{32}
lena	2.79×10^{17}
airplane	2.23×10^{51}
baboon	1
goldhill	6.24×10^{18}
man	8.81×10^{9}
pepper	3.69×10^{11}
woman	1.01×10^{25}

TAB. 6.3: Nombre de partitions P_{32} du passage du niveau du haut au suivant.

6.4 Discussion des résultats

Les expérimentations mettent en évidence deux choses. La première est que le nombre de partitions LAR possibles pour des images naturelles est très grand. La seconde est que, même en connaissant la partition LAR pour un niveau donné, il reste très difficile, pour ne pas dire impossible, de déterminer la partition LAR pour le niveau suivant. Ainsi, la partition LAR apparaît comme un outil intéressant pour effectuer une protection du contenu de l'image, avec un fonctionnement par niveaux. De plus, l'adaptation du seuil d'activité semble être le moyen de choisir le niveau de protection souhaité. Il est alors possible d'élaborer une stratégie de choix du seuil permettant d'obtenir un compromis entre entropie de la partition (et donc sécurité) et coût de codage (et donc efficacité).

6.5 Sécurité

La méthode de chiffrement partiel à coût nul que nous proposons procède par dissimulation d'une partie des données. Sa sécurité repose sur le fait qu'une attaque *par force brute* est trop coûteuse pour être envisageable. Pour illustrer cet aspect, prenons une petite image, de taille 128×128 avec une partition $\mathrm{QP}^{[16..2]}$. Cela correspond à un coût maximal de codage de 16.35 bits pour chaque bloc 16×16, soit au total, environ 1046 bits. Si, en réalité, l'entropie du quadtree est moindre, disons un quart de sa valeur maximale, il resterait environ 256 bits à coder, soit un espace de recherche exhaustive de $2^{256} \approx 1,16 \times 10^{77}$ partitions différentes. Si, lors de l'attaque, le décodage d'une partition dure 10^{-15} s, une femtoseconde, la durée totale de cette attaque serait de l'ordre de 10^{62} s, ou bien, 2.78×10^{58} heures, soit encore 3×10^{54} années. Ainsi, même pour une image de taille réduite, une attaque exhaustive est inenvisageable pour retrouver la partition.

La sécurité de notre méthode de chiffrement est une sécurité de type inconditionnelle. Il s'agit donc d'envisager les attaques possibles permettant de retrouver l'image codée à partir des données restantes, à savoir le flot plat et la texture. Certains aspects de la sécurité des codeurs reposant sur les quadtrees ont été étudiés par Cheng & Li (2000) et nous les exposons dans ce qui suit.

Les auteurs traitent d'un codeur d'image utilisant une décomposition quadtree. La structure de données associée est donc un quadtree, avec comme feuilles les niveaux de gris de chaque bloc. L'approche de chiffrement est un chiffrement partiel ne concernant que la structure quadtree. Deux types de parcours de l'arbre sont pris en compte. Le premier est un parcours de type

infixe, tandis que le second est un parcours en *largeur*. À ce propos, la section 3.1 rappelait les différents parcours d'un quadtree.

La sécurité inconditionnelle de cette méthode réside sur l'absence de corrélation entre le quadtree et les niveaux de gris des blocs représentés. Les auteurs montrent alors que le parcours *infixe* induit des corrélations locales qui permettent de reconstruire des morceaux de la structure du quadtree à partir des valeurs de niveaux de gris proches. D'un autre côté, ils montrent qu'un parcours *en largeur* du quadtree est nécessaire pour garantir un niveau de sécurité suffisant.

Cette conclusion peut être reprise à notre compte. En effet, dans le codeur LAR, la construction de la partition quadtree se fait de haut en bas, sans redondance. De plus, la seule information récupérable dans les niveaux de gris est le nombre de blocs de chaque niveau. La section 6.3 a déjà montré que cette information, même si elle réduit la taille de l'espace à explorer, n'est pas suffisante pour reconstruire l'image chiffrée.

6.6 À retenir

Ce chapitre a présenté quelques résultats sur l'intégration dans un codeur d'images fixes avec et sans pertes d'un mécanisme simple de chiffrement du contenu. La dissimulation d'une partie du flot binaire généré par le codeur suffit, sans coût additionnel, à fournir un chiffrement efficace. Il autorise des niveaux différenciés d'accès au contenu d'images codées avec un outil dont les performances, tant en sans perte que avec pertes, sont au-dela de l'état de l'art.

Comme pour toute méthode de chiffrement, l'évaluation des niveaux de sécurité n'est pas une tâche aisée. À côté des éléments théoriques et expérimentaux présentés, quelques pistes restent encore à explorer. D'abord, il est nécessaire de continuer à rechercher les vulnérabilités de la méthode, en imaginant de nouvelles attaques possibles. Ensuite, il s'agit d'évaluer l'information mutuelle existant entre la partition quadtree et les niveaux de gris codés. Il y a là matière tant théorique qu'expérimentale.

E. P. Jacobs, *Le secret de l'espadon, tome 2*, page 36, case 4.

7

Chiffrement sélectif du flot LAR iS+P

Plan

7.1 Position du problème et étude préliminaire
7.2 Chiffrement de la partition
7.3 Chiffrement combiné : partition et *flat*-LAR
7.4 Chiffrement combiné : partition, flat et vecteur d'initialisation
7.5 À retenir

Les chapitres précédents se sont attachés à présenter les algorithmes de chiffrement spécifiquement appliqués aux images. Dans le cadre du projet ANR TSAR, des usages adaptés à la mise en ligne de la base de données EROS ont été identifiés. Ils mettent en avant le besoin d'une transmission hiérarchiquement sécurisée. Il apparait donc évident de profiter de la hiérarchie intrinsèque du codeur LAR, en particulier celle du codeur LAR IS+P, afin de proposer une solution de chiffrement sélectif. Ainsi, après la description du problème et une étude préliminaire, nous proposons une méthode de chiffrement d'images par niveau pilotée par la partition quadtree servant de support à la compression. Pour améliorer les aspects de sécurité de cette méthode de chiffrement, d'autres portions du flot LAR sont utilisées, comme le *flat* LAR. L'initialisation des primitives de chiffrement utilisées sera évoquée ainsi que leur mise en œuvre. Nous montrerons l'efficacité d'un tel schéma ainsi que sa progressivité.

7.1 Position du problème et étude préliminaire

Cette partie a pour objet l'étude d'un chiffrement sélectif mais aussi hiérarchique des données issues du codec LAR Interleaved S+P. En se plaçant dans un contexte d'images de grandes tailles, la sélectivité du chiffrement permet de réduire la complexité et la durée du chiffrement. De plus, la hiérarchisation du chiffrement permet de préciser à quel niveau de détails et/ou de distorsions celui-ci a lieu. Ainsi, cela permet un réglage fin des configurations d'accès aux images.

La constitution du flot binaire, comme décrit en 3.2, fait apparaître deux composantes importantes : la partition quadtree et le flat LAR, adaptés au contenu sémantique de l'image. Le chapitre 6 a déjà montré l'intérêt de la partition dans le cadre d'un système de sécurisation de contenu.

Le flot *flat*-LAR du LAR IS+P est lui aussi constitué de 2 parties. D'une part, la partition quadtree, d'autre part, une construction multi-résolution correspondant aux blocs de la partition. Malgré le caractère supposé grossier de cette image flat, sa qualité visuelle est bonne. En suivant une démarche similaire au chiffrement de la partition, il sera intéressant de chiffrer le flot binaire flat LAR d'une manière hiérarchique.

Enfin, un raffinement de la méthode de chiffrement est considéré en prenant en compte les 4 valeurs initiales servant à la reconstruction des bords de l'image. Le chiffrement de ces 4 valeurs comme vecteur d'initialisation permettra d'augmenter l'espace des clefs.

7.2 Chiffrement de la partition

Cette section présente le chiffrement de la partition, en utilisant comme primitives de chiffrement RC4 et AES. RC4 est un chiffrement par flot décrit en 4.6.3 tandis qu'AES est un chiffrement par blocs présenté en 4.6.2. Ces chiffrements sont répandus, bien étudiés et leurs avantages et inconvénients sont connus. Ils sont normalisés et bien décrits, ce qui permet un usage aisé. Chacun des deux chiffrements est représentatif des chiffrements par flots et par blocs, respectivement. De plus, le chiffrement des flots binaires, quoique reposant sur des primitives de chiffrement, est relativement indépendant de la mise en œuvre de ces primitives. Ainsi, il serait facile de remplacer RC4 par un autre chiffrement par flot.

7.2.1 Principe

Il s'agit d'insérer un bloc fonctionnel de chiffrement entre le codeur et le décodeur sur les flots que l'on souhaite chiffrer. Dans notre cas, seuls les flots correspondants à la partition quadtree sont concernés.

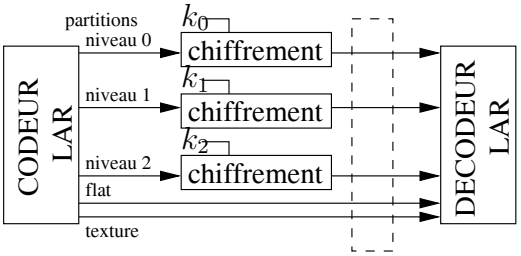
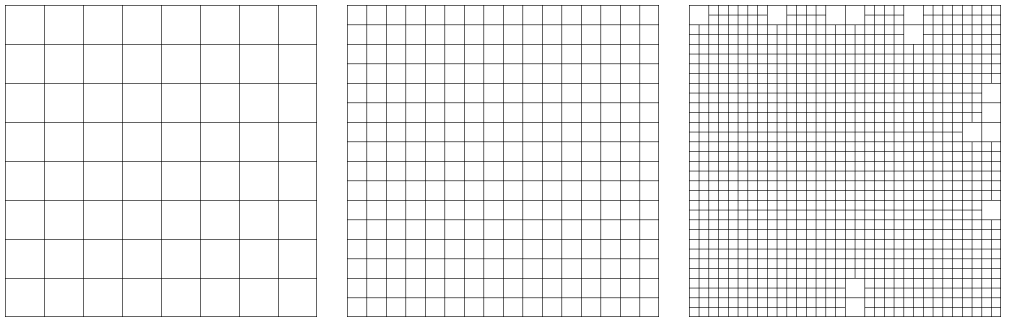


FIG. 7.1: Flots binaires LAR chiffrés

La figure 7.1 présente la réalisation d'un chiffrement sélectif des différents flots de la partition et permet d'étudier l'influence du niveau chiffré sur l'image finale obtenue en sortie du décodeur. En effet, si la protection des images de faible résolution et faible qualité n'est pas nécessaire, seul le chiffement des niveaux à partir du niveau requis est nécessaire. Les premiers niveaux de la partition ne seront donc pas chiffrés. L'utilisateur aura donc accès aux imagettes, mais il devra posséder une clef pour pouvoir visualiser et utiliser les images de résolution supérieure. La figure 7.2 présente les différents niveaux de la partition pour l'image *lena*.

Les clefs de chiffrement utilisées ont des longueurs de 16, 24 ou 32 octets (soient, respectivement, 128, 192 et 256 bits) pour AES, et jusqu'à 256 octets pour RC4. Les longueurs de ces clefs sont suffisantes pour obtenir une sécurité satisfaisante. Ensuite, le flot binaire chiffré, de même taille que le flot binaire



(a) Niveau 6 (b) Niveau 5 (c) Niveau 4

(d) Niveau 3 (e) Niveau 2 (f) Niveau 1 (base de la pyramide)

FIG. 7.2: Partitions issues des flots des niveaux en sortie du codeur pour l'image *lena*.

clair, est transmis en entrée du décodeur. Le résultat est une image modifiée suivant la nature du ou des flots de partition chiffrés.

7.2.2 Modifications visuelles

D'un point de vue visuel, les figures 7.3 et 7.4 montrent que plus le chiffrement est fait tôt dans le flot, c'est-à-dire dans les niveaux les plus hauts de la pyramide ou encore les images de faible résolution, et plus la dégradation constatée est importante sur l'image pleine résolution reconstituée à la sortie du décodeur. Si le chiffrement a lieu sur les niveaux hauts de la pyramide, l'image finale devient méconnaissable. Par contre, plus le chiffrement se fait à un niveau bas et plus l'image obtenue est reconnaissable. En effet, un niveau supérieur est utilisé pour la reconstruction du niveau inférieur (de plus grande résolution) : les dégradations générées sur les blocs par le chiffrement au niveau supérieur se propagent donc aux niveaux inférieurs.

Ces résultats sont obtenus après un chiffrement de type RC4 et AES avec une clef de 32 octets et pour l'image *lena*. Cependant, même si l'image est reconnaissable après chiffrement du dernier ou de l'avant-dernier niveau, l'image finale pleine résolution n'est pas utilisable du fait de qualité vraiment médiocre.

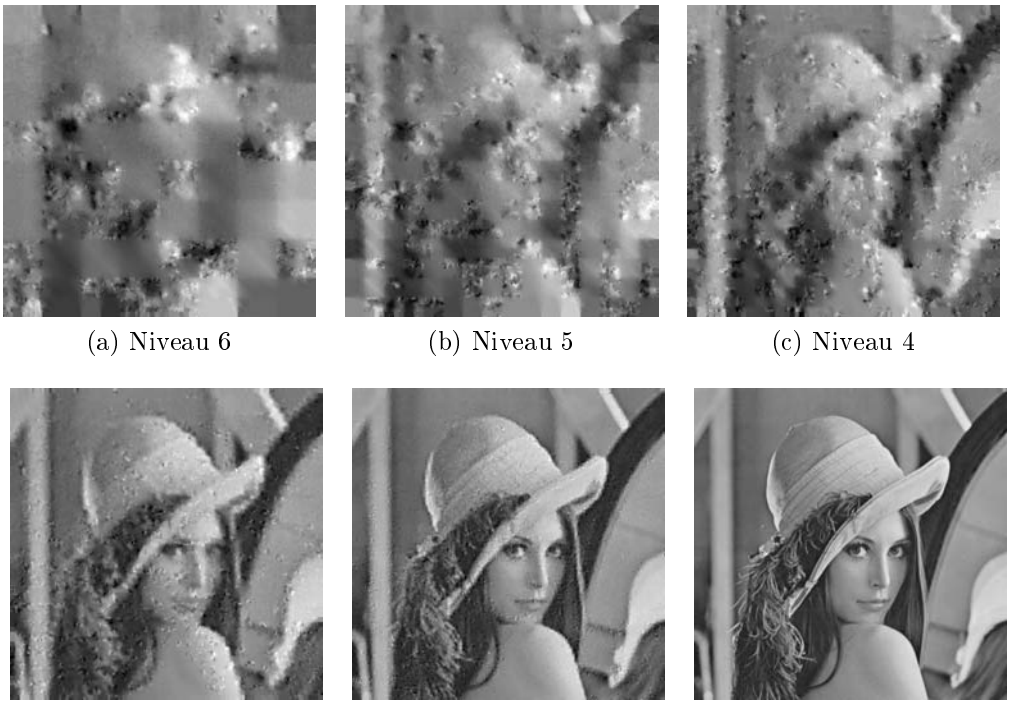
À un niveau de chiffrement donné, la taille de la clef utilisée pour le chiffrement n'a pas de réelle influence sur la qualité finale de l'image. En effet, les algorithmes RC4 et AES génèrent leur propre clef interne avant de chiffrer. De même, les images obtenues pour différentes clefs de même taille présentent des dégradations de même nature. Dans ces deux cas, seule apparaît une modification légère du bruit apparent : la quantité de bruit reste de même ordre de grandeur, seule la position des dégradations change, comme le montre la figure 7.5.

7.2.3 Résultats

L'évaluation des résultats se fait à l'aide de trois mesures, à savoir le PSNR, l'indice *Komparator* et le *gain composé*.

KOMPARATOR est une métrique objective perceptuelle introduite par Barba & Callet (2003)[1]. Elle modélise le système visuel humain et permet de comparer deux images entre elles. Cette métrique présente deux notes : une mesure objective, d'abord, évaluant les différences entre les deux images comparées. Une note subjective, ensuite, est calculée à partir de la mesure

[1] `http ://autrusseau.florent.club.fr/Komparator/index.html`



(a) Niveau 6 (b) Niveau 5 (c) Niveau 4

(d) Niveau 3 (e) Niveau 2 (f) Image originale

FIG. 7.3: Images *lena* reconstruites en sortie du décodeur après chiffrement (RC4, clef de 256 bits) d'un niveau donné de la partition.

(a) Niveau 5 (b) Niveau 4

(c) Niveau 3 (d) Niveau 2

FIG. 7.4: Images *lena* reconstruites en sortie du décodeur après chiffrement (AES, clef de 256 bits) d'un niveau donné de la partition.

(a) RC4, clef 128 bits (b) RC4, clef 256 bits (c) RC4, clef 512 bits

(d) AES, clef 128 bits (e) AES, clef 192 bits (f) AES, clef 256 bits

FIG. 7.5: Image *lena* chiffrée au niveau 3 par RC4 et AES avec des clefs de tailles différentes.

objective et d'une table de corrélation entre mesure objective et note subjective obtenue expérimentalement. Il faut remarquer que cette note ne donnera qu'une indication pour le codeur LAR. En effet, les tables de corrélation proviennent de résultats obtenus avec d'autres codeurs d'images.

Le GAIN COMPOSÉ (*compound gain*)[2], introduit par Garcia et al. (2001), est une métrique objective utilisant les statistiques des points saillants des images sous inspection. Son principe repose sur une généralisation du gain informationnel joint de KULLBACK-LEIBLER pour plusieurs variables aléatoires. L'idée est de mesurer la distance entre les histogrammes locaux autour de certains points saillants des deux images. Ainsi, le gain composé est la mesure du gain informationnel entre deux images satisfaisant certains critères naturels et utiles.

L'utilisation des métriques présentées pour évaluer des images codées par le LAR est discutable. En effet, la délivrance d'une note, ou d'une mesure, nécessite la connaissance du modèle psychovisuel associé aux déformations propres à la compression LAR. Ainsi, pour notre application, la mesure objective ou la note subjective corrélée n'ont d'intérêt que dans la comparaison relative de la qualité des images, mais en aucun cas, dans l'évaluation des performances du codeur relativement à un autre. Une étude liée à l'évaluation des modèles relatifs au codeur LAR déborde largement du cadre de ce document.

Les mesures sont faites pour différentes images, et pour l'image *lena*, avec différentes tailles de clef, avec un chiffrement sur chaque niveau.

Le tableau 7.1 récapitule les résultats obtenus avec *lena*.

Tout d'abord, la faible influence de la longueur de la clef se remarque : les valeurs du PSNR en particulier demeurent relativement stables. La mesure objective (Mobj) et la note objective (Nobj) délivrées par KOMPARATOR sont difficilement interprétables dans l'absolu. Cependant, la valeur très élevée de la mesure objective et une note subjective proche de 1 indiquent une importante différence visuelle entre l'image de référence et l'image chiffrée. Cette constatation corrobore les résultats du PSNR qui indiquent déjà une mesure d'environ 26 dB pour le chiffement le plus bas dans la pyramide. Or, un PSNR de 25 dB est considéré comme présentant une dégradation très élevée. De plus, en diminuant à chaque niveau de la partition, le PSNR confirme bien le fait qu'un chiffrement effectué plus tôt dégrade davantage l'image obtenue. Les résultats sont en accord puisque l'augmentation du gain composé traduit elle aussi le même comportement. Un chiffrement avec l'algorithme AES conduit aux mêmes observations, comme le montre le tableau 7.2.

[2] `http ://decsai.ugr.es/cvg/CG/`

	lena (clef 16 octets)			
	PSNR en dB	CG	KOMPARATOR	
			Mobj	Nobj
Niveau 2	25.94	14 093	2 987	1.56
Niveau 3	22.92	21 856	7 602	1.00
Niveau 4	20.12	33 900	10 558	1.00
Niveau 5	17.85	44 796	13 319	1.00
Niveau 6	15.46	51 947	12 420	1.00

	lena (clef 32 octets)			
	PSNR en dB	CG	KOMPARATOR	
			Mobj	Nobj
Niveau 2	25.94	14 219	3 228	1.38
Niveau 3	22.93	22 057	8 051	1.00
Niveau 4	20.17	33 813	11 043	1.00
Niveau 5	17.91	41 941	13 054	1.00
Niveau 6	15.58	59 057	12 670	1.00

	lena (clef 64 octets)			
	PSNR en dB	CG	KOMPARATOR	
			Mobj	Nobj
Niveau 2	25.95	14 074	3 277	1.34
Niveau 3	22.83	21 599	8 657	1.00
Niveau 4	20.18	33 704	11 883	1.00
Niveau 5	18.00	42 247	12 593	1.00
Niveau 6	15.60	55 903	12 514	1.00

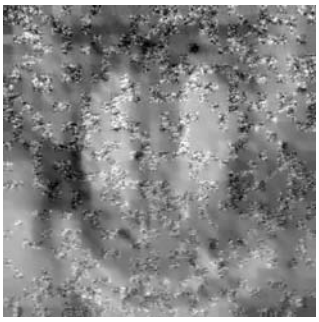
TAB. 7.1: Chiffrement RC4 de la partition : résultats sur image *lena* pour différentes tailles de clef. CG est le gain composé, KOMPARATOR fournit une mesure objective et une note subjective.

	lena (clef 16 octets)			
	PSNR en dB	CG	KOMPARATOR	
			Mobj	Nobj
Niveau 2	25.94	14 256	3 426	1.23
Niveau 3	22.77	21 386	8 461	1.00
Niveau 4	20.14	34 215	11 663	1.00
Niveau 5	18.08	41 062	11 079	1.00

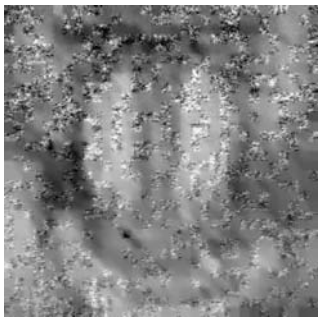
	lena (clef 24 octets)			
	PSNR en dB	CG	KOMPARATOR	
			Mobj	Nobj
Niveau 2	25.95	14 094	3 243	1.36
Niveau 3	22.90	21 924	8 661	1.00
Niveau 4	20.07	32 665	11 329	1.00
Niveau 5	17.85	48 659	12 175	1.00

	lena (clef 32 octets)			
	PSNR en dB	CG	KOMPARATOR	
			Mobj	Nobj
Niveau 2	25.92	14 292	3 310	1.31
Niveau 3	22.90	21 612	7 825	1.00
Niveau 4	20.21	35 974	10 411	1.00
Niveau 5	17.75	42 700	13 020	1.00

TAB. 7.2: Chiffrement AES de la partition : résultats sur image *lena* pour différentes tailles de clef.

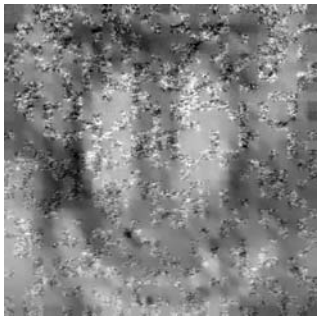


(a) Omission d'un caractère

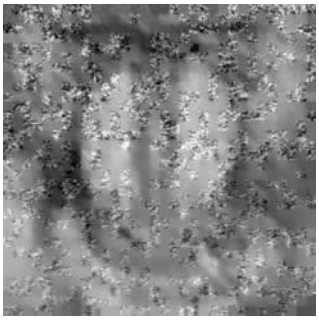


(b) Ajout d'un caractère

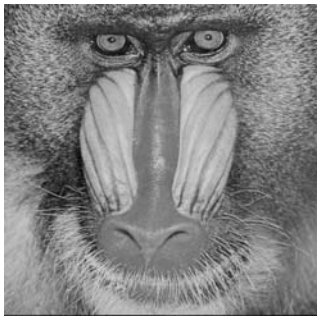
(c) Substitution d'un caractère



(d) Décalage circulaire



(e) Permutation



(f) Image originale

FIG. 7.6: Image *baboon* avec le niveau 4 chiffré, RC4 et clef de 512 bits, et déchiffrée avec une clef erronée.

Concernant la robustesse des algorithmes lors du déchiffrement, pour une très faible variation de la clef comme :

- l'omission d'un caractère ;
- l'ajout d'un caractère ;
- la substitution d'un caractère ;
- le décalage circulaire d'un caractère ;
- la permutation de deux caractères consécutifs ;

les résultats indiquent que les algorithmes RC4 et AES sont résistants. Les tests visuels et numériques effectués confirment en effet la fiabilité des algorithmes RC4 et AES dans le sens où l'on obtient une image dégradée similaire à celle obtenue après chiffrement s'il n'y avait pas eu de déchiffrement (PSNR du même ordre), et ce pour chaque niveau de chiffrement (voir figure 7.6).

Ainsi, combiné à l'aspect hiérarchique du codec LAR, le chiffrement des flots de la partition permet à la fois de protéger efficacement l'image décodée puisque même la dégradation la plus légère rend l'image inutilisable pour une quelconque reproduction, et de laisser la possibilité d'un accès aux imagettes de faible résolution grâce au chiffrement sélectif.

7.3 Chiffrement combiné : partition et *flat*-LAR

Nous allons décrire ici le principe du chiffrement du *flat*-LAR et étudier l'influence d'un chiffrement RC4 puis AES effectué à la fois sur la partition et sur le flat pour différentes tailles de clefs. Nous détaillerons en particulier les résultats obtenus pour les images *lena*, *baboon* et l'image médicale *abdomen*.

7.3.1 Principe

De même que pour le chiffrement de la partition, on applique les algorithmes RC4 et AES aux flots du flat, codant l'image *flat*, basse résolution du LAR, récupérés en sortie du codeur. On chiffre donc à un niveau donné le flot flat, modifiant de fait les valeurs moyennes pour les blocs des quadtrees. Le principe est le même que celui de la section 7.2, mais on combine cette fois le chiffrement de la partition avec celui du flat comme indiqué sur la figure 7.7. Là aussi, les taille de clefs sont les mêmes, à savoir 16, 32 ou 64 octets pour l'algorithme RC4 et 16, 24 ou 32 octets pour AES.

7.3.2 Modifications visuelles

Bien évidemment, le fait de rajouter le chiffrement du flat ne change en rien le comportement hiérarchique constaté précédemment : on dispose tou-

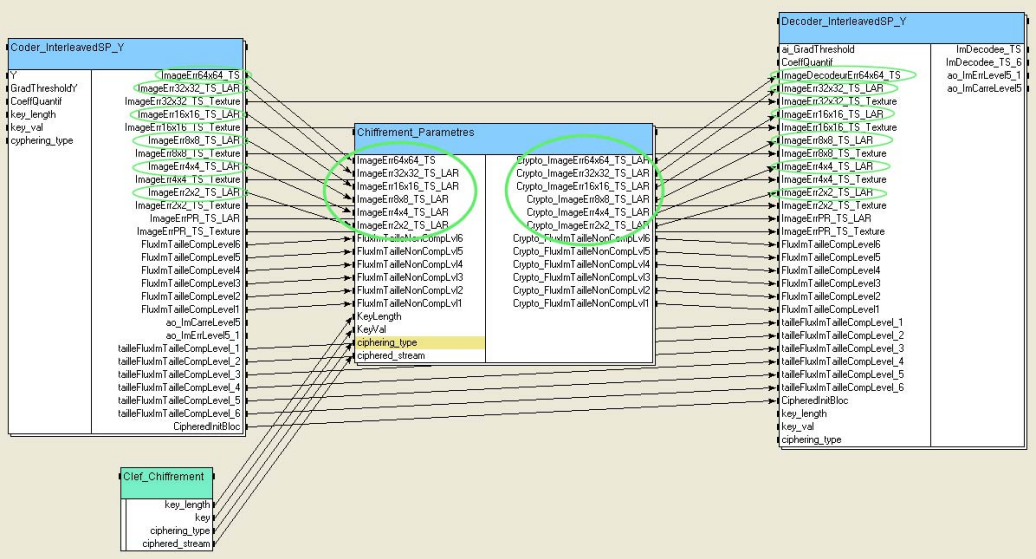


FIG. 7.7: Vue de SynDEX, fenêtre principale, présentant l'ajout du chiffrement des flots du LAR flat.

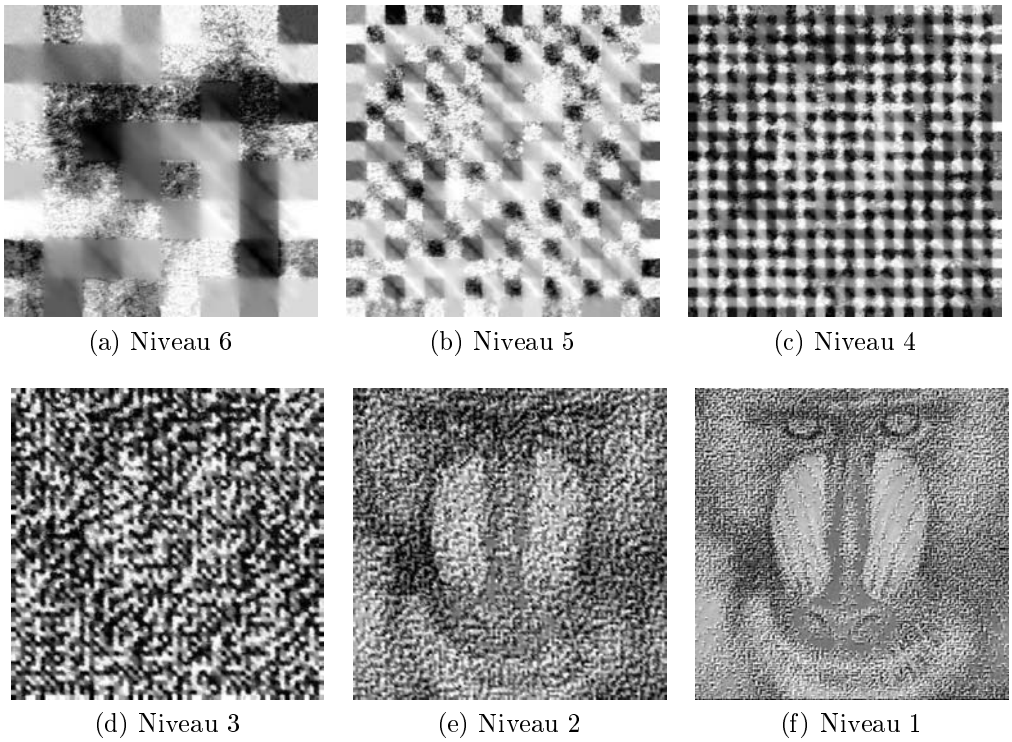
jours d'une dégradation progressive de l'image lors des chiffrements niveaux après niveaux. Cependant, la modification du flot du flat par chiffrement entraîne un changement de la valeur moyenne codée par ce flot, et on obtient alors visuellement un nouveau niveau de gris pour le bloc du quadtree concerné. De manière sporadique, il apparaît des blocs de la plus petite taille du niveau chiffré près de certains contours de l'image. Cumulée au chiffrement de la partition, on obtient une diffusion de ces blocs à travers toute l'image. Sur la figure 7.8, on observe très bien les effets sur l'image *baboon* très texturée.

La dégradation obtenue est bien plus importante pour un chiffrement combiné partition et flat que pour un chiffrement unique de la partition, et ce quel que soit le niveau chiffré.

Pour un niveau donné, en comparant les résultats visuels, comme ceux obtenus sur la figure 7.9, on constate bien une dégradation nettement plus importante de l'image que le chiffrement utilisé soit RC4 ou AES. Cette remarque s'avère également valable pour les images peu texturées.

7.3.3 Résultats

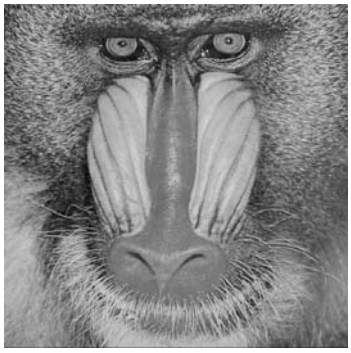
Pour un chiffrement combiné partition et flat, de même que pour l'étude réalisée sur le chiffrement de la partition, la longueur de la clef définie par l'utilisateur n'a pas de réelle importance sur l'image finale même si les dé-

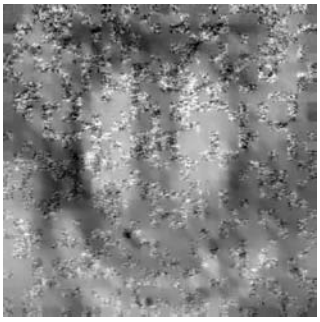


(a) Niveau 6 (b) Niveau 5 (c) Niveau 4

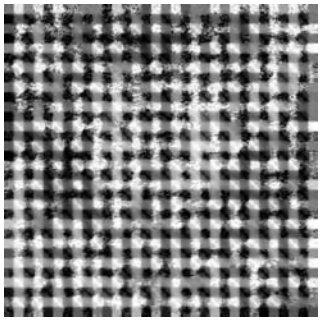
(d) Niveau 3 (e) Niveau 2 (f) Niveau 1

FIG. 7.8: Chiffrement RC4 de la partition et du flat de l'image *baboon* pour différents niveaux.

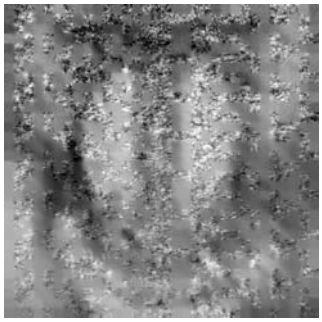




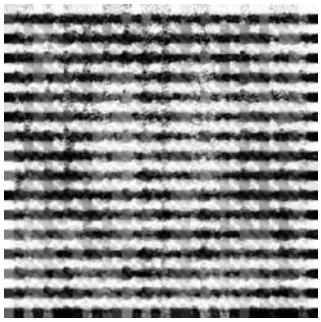
(a) RC4 : partition



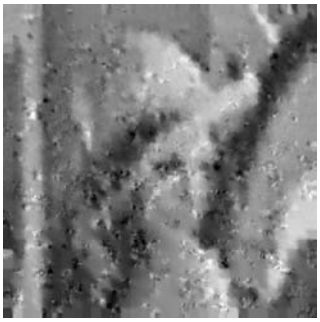
(b) RC4 : partition et flat



(c) AES : partition



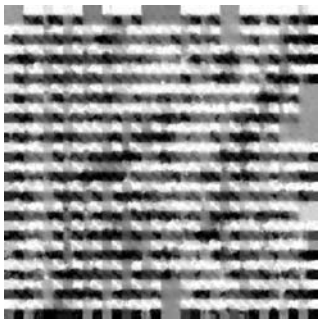
(d) AES : partition et flat



(e) RC4 : partition

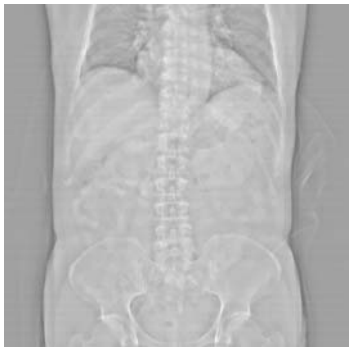
(f) RC4 : partition et flat

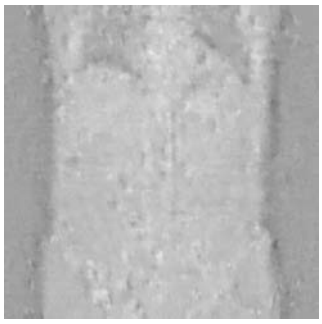
(g) AES : partition



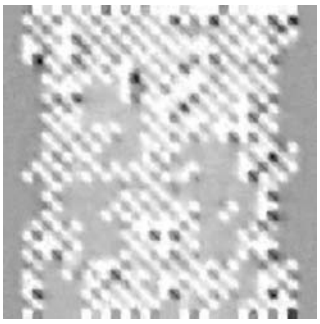
(h) AES : partition et flat

FIG. 7.9: Comparaison du chiffrement de la partition et du chiffrement combiné de la partition et *flat* pour le niveau 4

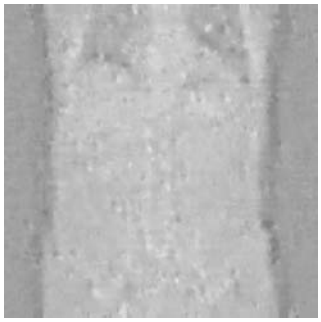




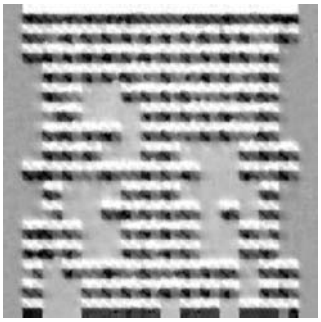
(i) RC4 : partition



(j) RC4 : partition et flat



(k) AES : partition



(l) AES : partition et flat

FIG. 7.9: Comparaison du chiffrement de la partition et du chiffrement combiné de la partition et *flat* pour le niveau 4 (suite et fin).

gradations constatées sont légèrement plus élevées dans le cas où la clef est de grande taille. Étant données deux clefs de tailles différentes, on remarque que le PSNR et Mobj varient légèrement mais restent proches pour une image et un niveau donné, comme le montrent les tableaux 7.3.

Par contre, on remarque que la mesure perceptuelle KOMPARATOR retourne une note Nobj égale à 1.00, la pire qui soit, même sur le niveau 1 le plus bas. Nobj caractérise bien le fait que l'image obtenue est très différente de l'image originale en attribuant cette note. Il est surtout intéressant de constater la forte diminution du PSNR entre le chiffrement simple de la partition et le chiffement combiné de la partition et du flat (voir les tableaux 7.1 pour la partition et 7.3 pour la partition et le flat). En effet, on remarque en moyenne sur les tableaux 7.4 une forte diminution de l'ordre de 10 dB pour chaque niveau sur les chiffrements RC4 et AES. L'image médicale Abdomen va même jusqu'à perdre 18dB entre le chiffrement AES de la partition et celui combiné partition et flat.

Tout en conservant la propriété de chiffrement hiérarchique, le chiffrement combiné des flots de la partition et du flat apporte une dégradation bien plus importante sur l'image décodée, sécurisant plus ainsi l'image pleine résolution pour un chiffrement sur le dernier niveau.

7.4 Chiffrement combiné : partition, flat et vecteur d'initialisation

De manière à ajouter une sécurité supplémentaire, on va chercher à augmenter l'espace des possibilités concernant l'image finale décodée en chiffrant la structure appelée vecteur d'initialisation. Nous allons décrire la structure et analyser ici le principe du chiffrement du vecteur d'initialisation. Les résultats visuels et numériques seront présentés ensuite.

7.4.1 Principe

Toujours en se servant des algorithmes RC4 et AES, nous testons ici le chiffrement des quatre octets formant le vecteur d'initialisation. Ces quatre octets servent de valeur initiale pour le prédicteur de niveau le plus haut de la pyramide du LAR IS+P.

7.4.2 Modifications visuelles

Le chiffrement du vecteur d'initialisation provoque un décalage appliqué à l'ensemble des pixels de l'image, comme le montre la figure 7.10. En effet,

	lena (clef 16 octets)			
	PSNR en dB	CG	KOMPARATOR	
			Mobj	Nobj
Niveau 1	16.64	-1 451	8 608	1.00
Niveau 2	15.35	383	14 944	1.00
Niveau 3	11.51	-3 867	18 685	1.00
Niveau 4	9.68	3 460	17 334	1.00
Niveau 5	9.81	19 852	15 019	1.00
Niveau 6	6.57	65 355	15 144	1.00

	lena (clef 32 octets)			
	PSNR en dB	CG	KOMPARATOR	
			Mobj	Nobj
Niveau 1	19.34	844	7037	1.00
Niveau 2	12.95	-3004	19479	1.00
Niveau 3	11.80	-1562	19154	1.00
Niveau 4	11.99	7968	18759	1.00
Niveau 5	9.51	20958	13511	1.00
Niveau 6	6.51	82762	14759	1.00

	lena (clef 64 octets)			
	PSNR en dB	CG	KOMPARATOR	
			Mobj	Nobj
Niveau 1	16.83	-698	8751	1.34
Niveau 2	13.78	2256	12390	1.00
Niveau 3	13.95	-117	18115	1.00
Niveau 4	10.30	3442	13830	1.00
Niveau 5	10.35	23107	14367	1.00
Niveau 6	10.35	81941	14947	1.00

TAB. 7.3: Chiffrement RC4 de la partition et du flat : mesure de la distorsion (PSNR), du gain composé (CG) et des notes de KOMPARATOR (Mobj et Nobj) pour l'image lena pour différentes tailles de clef

	PSNR (*lena*)			
	RC4		AES	
	partition	partition + flat	partition	partition + flat
Niveau 1	-	19.34	-	16.24
Niveau 2	25.94	12.95	25.92	14.19
Niveau 3	22.93	11.80	22.90	11.09
Niveau 4	20.17	11.99	20.21	9.87
Niveau 5	17.91	9.51	17.75	10.82
Niveau 6	15.58	6.35	-	-

	PSNR (*baboon*)			
	RC4		AES	
	partition	partition + flat	partition	partition + flat
Niveau 1	-	11.64	-	10.81
Niveau 2	18.35	12.84	18.36	10.42
Niveau 3	17.16	9.78	17.16	9.25
Niveau 4	16.35	10.31	16.34	9.14
Niveau 5	15.57	9.63	15.77	9.44
Niveau 6	14.74	8.95	-	-

	PSNR (*abdomen*)			
	RC4		AES	
	partition	partition + flat	partition	partition + flat
Niveau 1	-	25.57	-	23.56
Niveau 2	37.43	25.02	37.41	19.26
Niveau 3	33.01	17.94	32.96	16.11
Niveau 4	29.35	16.40	29.58	11.85
Niveau 5	27.14	10.38	27.36	9.17
Niveau 6	26.12	12.67	-	-

TAB. 7.4: Comparaison des PSNR entre le chiffrement simple de la partition et celui combiné partition et *flat*.

l'image correspondant au niveau le plus haut de la pyramide est obtenue par une modulation d'impulsion codée en différence. Le prédicteur partant des valeurs d'initialisation, une modification de celles-ci vient changer la valeur moyenne de l'ensemble de l'image.

(a) N=196 (b) N=128 (c) N=54

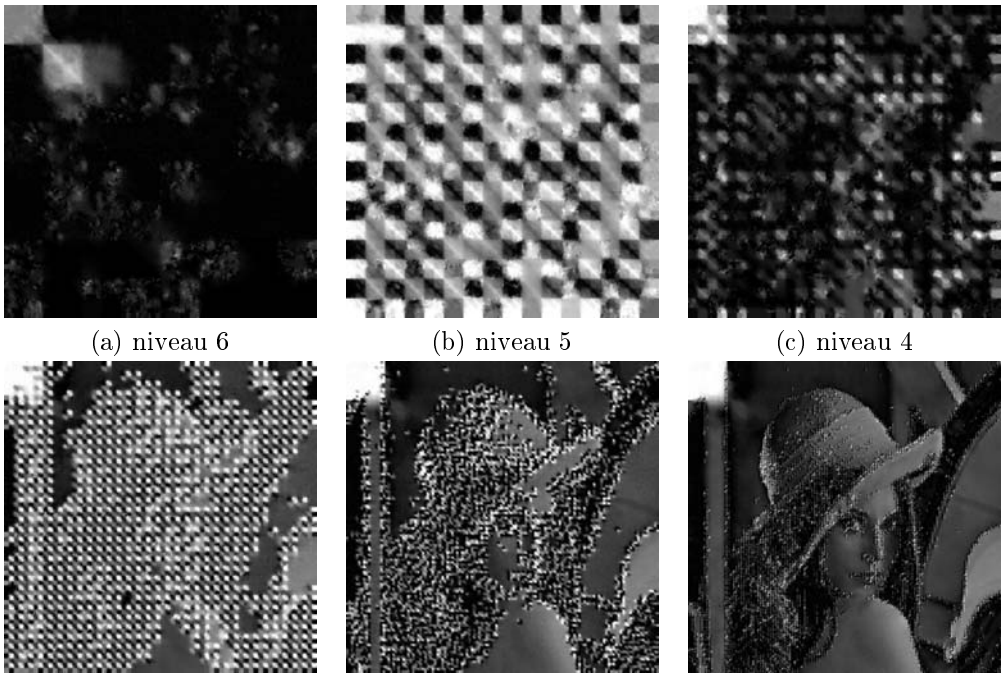
FIG. 7.10: Image *lena* déchiffrée (RC4) après modification au décodeur d'une valeur initiale.

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

FIG. 7.11: Visualisation de régions sombres ou claires sur l'image *lena* déchiffrée (RC4) après modification d'un des paramètres au décodeur.

On obtient donc une image plus claire ou plus sombre mais certaines régions codées initialement par une valeur proche de 0 ou 255 n'ont pas le même décalage que les autres régions. En effet, les pixels chiffrés prennent alors la valeur limite 0 dans le cas d'un décalage négatif et 255 dans le cas d'un décalage positif. Cela crée des régions de saturées, illustrées par la figure 7.11.

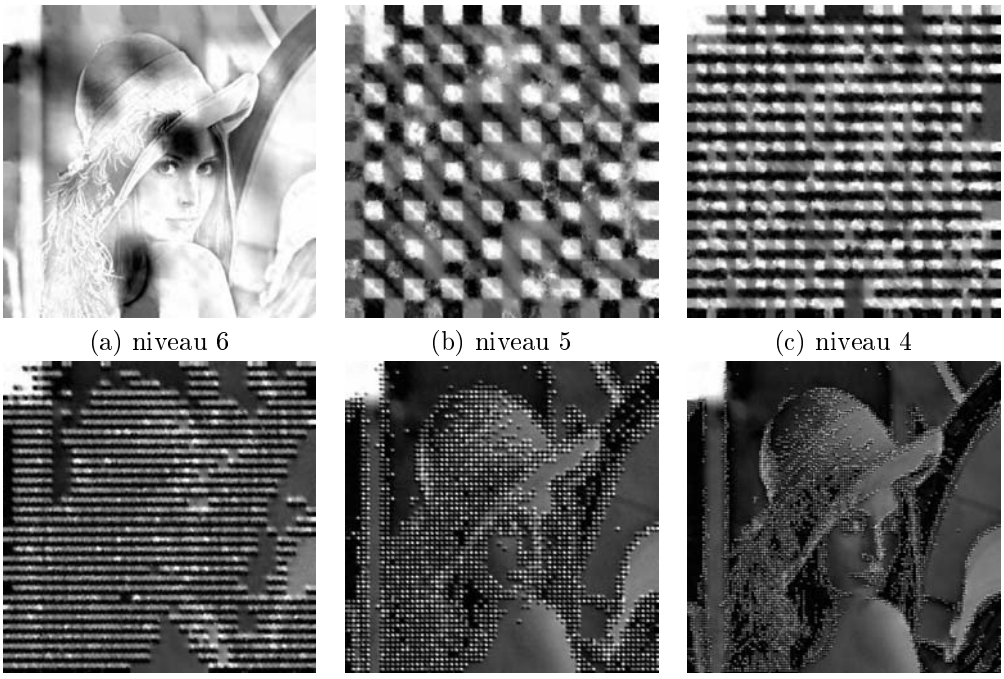
Ces déformations peuvent être cumulées avec les chiffrements réalisés précédemment sur la partition et le *flat*. On constate alors sur les figures 7.12 et 7.13 que le chiffrement combiné de ces trois structures apporte encore un meilleur résultat visuel en termes de dégradations.



(a) niveau 6 (b) niveau 5 (c) niveau 4

(d) niveau 3 (e) niveau 2 (f) niveau 1

FIG. 7.12: Chiffrement RC4 combiné partition, flat et vecteur d'initialisation sur image lena pour différents niveaux



(a) niveau 6 (b) niveau 5 (c) niveau 4

(d) niveau 3 (e) niveau 2 (f) niveau 1

FIG. 7.13: Chiffrement AES combiné partition, flat et vecteur d'initialisation sur l'image *lena* pour différents niveaux

7.4.3 Résultats

Le chiffrement combiné partition, flat et vecteur d'initialisation obtient des résultats en termes de PSNR encore meilleurs, ce qui traduit bien une dégradation encore plus flagrante. On obtient des valeurs très faibles de PSNR : de l'ordre de 10 dB et ce dès le niveau le plus bas. Comme le montre le tableau 7.5, le chiffrement additionnel du vecteur d'initialisation permet d'assurer une forte dégradation à tous les niveaux.

	PSNR *lena*					
	RC4			AES		
	partition	partition + flat	partition + flat + vecteur	partition	partition + flat	partition + flat + vecteur
Niveau 1	-	19.34	9.91	-	16.24	9.72
Niveau 2	25.94	12.95	9.94	25.92	14.19	9.64
Niveau 3	22.93	11.80	10.34	22.90	11.09	9.55
Niveau 4	20.17	11.99	8.21	20.21	9.87	9.30
Niveau 5	17.91	9.51	9.86	17.75	10.82	10.09
Niveau 6	15.58	6.35	6.46	-	-	8.90

TAB. 7.5: Comparaison des PSNR entre le chiffrement simple de la partition, combiné partition et flat et combiné partition, flat et vecteur d'initialisation pour l'image *lena*

Bien que le chiffrement des deux premières structures étudiées, partition et flat, proposent déjà une solution fiable en dégradant efficacement l'image décodée, le chiffrement additionnel du vecteur d'initialisation permet encore d'augmenter la dégradation de l'image et donc sa sécurisation.

7.5 À retenir

Dans ce chapitre, nous avons montré l'intérêt qu'il y a à chiffrer une partie du flot généré par le codeur LAR. Sa structure hiérarchique est utilisée pour effectuer un chiffrement sélectif, progressif tant dans la quantité de données chiffrées que dans la non intelligibilité des images chiffrées. Ainsi, le coût du chiffrement est-il parfaitement maîtrisé. De plus, les aspects purement cryptographiques sont fournis par des primitives de chiffrement efficaces, fonctionnant par flot ou par bloc. La méthode repose sur l'efficacité de ces primitives et non sur leur structure. Il serait donc aisé d'en changer, pour tenir compte de l'évolution des techniques de chiffrement à proprement parler. Après la pierre angulaire de la compression efficace du codeur LAR, nous disposons donc désormais d'une méthode de chiffrement efficiente, propre à servir dans le contexte de l'archivage sécurisé d'images.

Troisième partie

Insertion de données

E. P. Jacobs, *SOS météores*, page 25, case 8.

8

Dissimulation de données dans les images

Plan

8.1 Généralités
8.2 Tour d'horizon
8.3 Propriétés de l'insertion de données
8.4 Les attaques
8.5 Techniques d'insertion de données
8.6 Conclusion

La précédente partie de cette étude s'est attachée à présenter l'intégration de mécanismes de chiffrement dans le codeur LAR iS+P, afin de permettre une sécurisation de l'accès aux images. Ainsi, il est désormais possible de chiffrer de manière hiérarchique les données issues du codeur LAR iS+P . Cet aspect est une des facettes liées à l'archivage sécurisé d'images.

Un autre aspect concerne l'ajout de meta-données aux images. Ces données peuvent concerner l'indexation des images d'une part, facilitant ainsi les fonctions de recherche dans une grande base de données images. D'autre part, des données propres à l'images, fournissant par exemple les conditions de prises de vue, ou tout autre donnée pertinente, ont intérêt à être stockées avec l'image elle-même, cela pour éviter d'éventuelles incohérences.

Ainsi, ce chapitre présente les différentes facettes de la dissimulation de données dans les images. Après quelques précisions de vocabulaire, il effectue un tour d'horizon des domaines scientifiques concernés. La section 8.3 introduit les bonnes propriétés attendues d'un système d'insertion de données, en particulier sa capacité d'insertion et les dégradations qui sont introduites dans les images tatouées. D'un autre point de vue, la section 8.4 présente les principales attaques visant à s'affranchir des données insérées. Ensuite, un état de l'art des techniques d'insertion est fait à la section 8.5, tant dans le domaine spatial que dans le domaine transformé.

8.1 Généralités

Dissimulation de données, *watermarking*, tatouage, stéganographie, filigrane : les termes sont nombreux pour décrire des activités proches. L'idée majeure consiste à insérer des données dans un support constitué d'autres données. La discrétion de la méthode, sa résistance aux transformations, son objectif économique, entre autres, permettent de faire la distinction entre les méthodes.

La difficulté du vocabulaire vient de l'utilisation de termes anglo-saxons, dans une discipline dont les origines sont lointaines, mais dont la mise en œuvre scientifique est récente. Le *watermarking* est la vision moderne de la dissimulation de données. En anglais, le *watermark* est, à l'origine, le filigrane placé dans le papier des billets de banque pour en assurer l'authenticité. Étymologiquement, il traduit l'image d'une marque dans l'eau, indiquant à la fois la discrétion de l'insertion et la résilience de la marque. Depuis peu, le *watermarking* englobe toutes les disciplines concernant la dissimulation de données. En langue française, à côté de l'anglicisme *watermarking*, le terme d'usage est le tatouage (numérique). Ce terme englobe souvent le tatouage à

proprement parler (visible) et les autres techniques comme la stéganographie et le filigrane (invisibles).

Quelle que soit la branche de la dissimulation de données, deux aspects sont présents. D'une part, une technique d'insertion de données dans l'image, avec pour objectif la discrétion, et d'autre part, une technique de synchronisation, avec pour objectif la résilience. Ces deux aspects peuvent être réalisés successivement, ou bien conjointement.

Ce document ne se veut pas exhaustif et ne rentre pas dans tous les détails de l'insertion de données. Son objectif est d'assurer au lecteur les concepts nécessaires à la compréhension du schéma d'insertion de données décrit dans le chapitre 9. Pour davantages de détails, les livres de Davoine & Pateux (2004) et Cox et al. (2007) sont deux références assez complètes du domaine.

8.2 Tour d'horizon

8.2.1 Stéganographie

La stéganographie est, étymologiquement, l'écriture (graphie) cachée (stégano). L'idée est de se servir d'un message anodin comme couverture d'un message caché. Il s'agit de faire passer le message inaperçu.

Utilisée depuis les Grecs, la stéganographie a longtemps reposé sur la manipulation des lettres ou de la peinture. Son usage était restreint et la quantité d'information dissimulée, en rapport avec la quantité d'information dissimulante, était faible.

L'époque contemporaine, avec des données numériques en volumes croissants, permet d'augmenter de manière importante la quantité de données (cachées) transportées. Très souvent, afin de renforcer la sécurité du message en cas de découverte, les données à dissimuler sont chiffrées avant insertion. Cette remarque vaut pour toutes les techniques de dissimulation.

8.2.2 Tatouage numérique

Dans le cadre du tatouage numérique, la quantité d'information insérée est relativement faible. Il s'agit principalement d'assurer la gestion des droits numériques, pour lutter contre les reproductions non autorisées et assurer la protection des droits de propriété intellectuelle. Ces données peuvent identifier l'ordinateur ou l'appareil qui tente de reproduire le medium protégé et déterminer si cette utilisation est autorisée. Le tatouage est normalement imperceptible et indétectable par tout système ignorant son mode d'insertion. La quantité de données brutes est habituellement très faible, et c'est plus la

façon dont elles sont insérées (et avec quelle clef) qui fait la marque et qui définit l'authenticité du média.

8.2.3 Prise d'empreinte numérique

La prise d'empreinte numérique, en anglais *fingerprinting*, consiste à générer un numéro unique identifiant l'acheteur, le possesseur, l'utilisateur final d'un médium et à l'insérer dans le medium lui-même. La faible quantité d'informations nécessaire à cela permet de coder aisément ce *numéro de série*. Cette technique vise à suivre les media et leur éventuelle reproduction non autorisée et leur auteur.

8.2.4 Insertion de (meta)-données

Certaines applications nécessitent des données jointes à une image dans un fichier. Les données EXIF[1], décrit dans JEITA (2002), ou les données XMP[2] précisées dans XMP specification (2005), concernent les conditions de la prise de vue d'une photo numérique, avec des données photographiques et de positionnement par exemple. Dans ce cadre, la solution naïve est d'insérer ces informations dans l'entête du fichier contenant l'image. Cette solution ne résiste pas au transcodage et ne permet pas de chiffrer l'information.

Une solution plus élégante, robuste et discrète est de placer les metadonnées directement dans l'image, à l'aide d'une technique d'insertion de données. Dans ce cas, la quantité d'information à insérer est plus importante, avec une redondance moindre, et dont l'objet principal n'est pas la protection des droits liés à l'image.

8.2.5 Pour la suite

Afin d'être consistant dans le vocabulaire, le terme d'*insertion de données* sera utilisé par la suite. Il reflète bien le fait que le message est inséré dans l'image servant de support. Cette insertion peut se faire plus ou moins discrètement.

Une vision davantage *traitement du signal* considère que le message couvrant est un canal de communication, avec un débit déterminé et un taux d'erreur estimé, servant à transmettre le message caché. À ce propos, la lecture de la thèse de Furon (2002), en particulier le chapitre 3, montre cet aspect traitement de signal de manière très intéressante.

Après l'insertion de données, deux questions peuvent se poser :

[1] EXchangeable Image File format
[2] eXtensible Metadata Platform

– Y-a-t'il des données insérées dans l'image ?
– Quel est le message inséré dans l'image ?

À la première question correspond la mise en œuvre d'un *détecteur* tandis qu'à la seconde répond un *décodeur*. Bien sûr, les tâches sont différentes et très souvent, il est beaucoup plus simple de détecter le message caché que de le déchiffrer correctement.

8.3 Propriétés de l'insertion de données

Les propriétés principales des schémas d'insertion de données sont la capacité d'insertion, la complexité de la méthode, son inversibilité, sa transparence, sa robustesse, sa sécurité et sa vérification. Ces notions s'avèrent indispensables à l'élaboration et la caractérisation d'une méthode relative à l'insertion de données.

8.3.1 Capacité d'insertion

La capacité d'insertion est la quantité d'information qu'il est possible d'introduire dans le message couvrant. Elle est soit mesurée de manière absolue en nombre de bits, soit de manière relative en bpp.

Deux mesures peuvent être faites : d'une part, la capacité d'insertion brute, évaluée à l'émission du message et la capacité d'insertion résiduelle, évaluée à la réception du message. Cette dernière est donc mesurée en tenant compte d'éventuelles attaques ou erreurs de transmission.

8.3.2 Complexité de la méthode

Cette mesure précise l'effort calculatoire nécessaire à l'insertion des données et la difficulté de l'attaque ou la détection ainsi que le coût pour récupérer le message inséré. Il est souvent souhaitable d'avoir une complexité faible pour l'émission et la réception et une complexité élevée pour l'attaque ou la détection.

8.3.3 Inversibilité

Une insertion de données est dite inversible s'il est possible d'enlever complètement les données insérées pour récupérer l'image originale sans distorsion. Notre étude se place naturellement dans ce cadre.

8.3.4 Robustesse

La robustesse caractérise le comportement de la méthode face aux erreurs de transmission. Pour simplifier, la robustesse indique s'il est possible de récupérer tout ou partie du message caché à la réception. Elle est liée à la capacité d'insertion résiduelle.

Un schéma d'insertion est *fragile* si le message ne peut être récupéré après le moindre changement. Cela peut être utile pour détecter une modification du message.

L'insertion est dite *semi-fragile* si le schéma résiste aux petites transformations mais pas aux attaques malicieuses.

L'insertion sera *robuste* si elle résiste à une classe déterminée de transformations.

8.3.5 Sécurité

Le comportement d'une méthode d'insertion de données face à des attaques montre le niveau de sécurité de celle-ci. Il s'agit d'évaluer, par exemple, la quantité d'information cachée qui peut être soit effacée, soit récupérée par un utilisateur non autorisé.

8.3.6 Transparence

L'insertion de données est dite transparente lorsque les modifications de l'image sont négligeables. Les mesures de distorsion sont les mêmes que pour les techniques de compression. Si les données insérées se voient, elles sont *perceptibles* autrement, elles seront dites *imperceptibles*.

8.3.7 Type de vérification

Quelle est l'information nécessaire pour récupérer le message caché ? S'il ne faut rien, l'insertion est dite *aveugle*. S'il faut une information de bord, comme une clef de déchiffrement, l'insertion est dite *informée*. Enfin, s'il faut le message original de couverture, l'insertion est appelée *non-aveugle*.

Évidemment, les difficultés et les performances des schémas d'insertion seront fortement dépendantes du type de vérification. Par exemple, les détecteurs sont plus simples lorsque le message de couverture est utilisable.

8.4 Les attaques

Une attaque est une modification exercée sur une image contenant une marque ou des données insérées. Les attaques sont multiples, avec des objectifs différents et sur différents supports. Un classement simpliste peut distinguer les attaques involontaires des attaques malicieuses.

Les attaques involontaires sont faites sans vouloir causer de préjudice aux données insérées. Un exemple est le transcodage voire même le changement de taux de compression à l'intérieur d'un même schéma de codage.

Les attaques malicieuses visent directement à éliminer ou réduire les données insérées ou à les récupérer. Dans ce cas, elles utilisent éventuellement les informations disponibles sur la méthode d'insertion et son fonctionnement.

Le distinction entre les deux types d'attaques ne dépend pas principalement des techniques mises en œuvre mais de l'objectif poursuivi par l'assaillant.

8.4.1 Attaque par filtrage

L'application de filtres peut, avec une faible dégradation de l'image, altérer considérablement une marque qui serait concentrée dans une bande de fréquences spatiales réduite. On rencontre, entre autres, des filtres median et des filtres passe-bas.

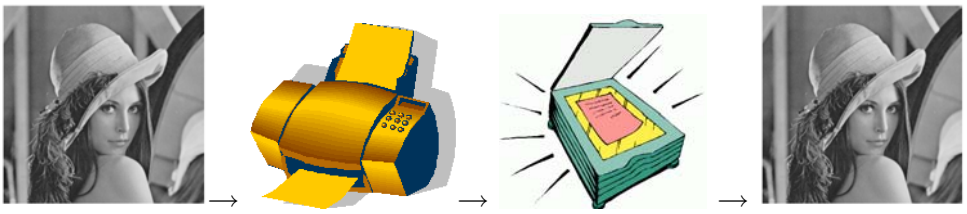
8.4.2 Attaques géométriques

Parmi les traitements effectués souvent de manière non malicieuse, les transformations géométriques sont les plus fréquentes. Une symétrie par rapport à l'axe vertical, par exemple, modifie légèrement le sens de l'image, avec une qualité constante. Cependant, elle peut avoir des effets néfastes sur les données insérées.

Les transformations géométriques peuvent consister en :

- une translation ;
- une rotation ;
- un recadrage ;
- une symétrie ;
- un zoom ;
- un changement d'échelle.

Une transformation géométrique parfaitement malicieuse est une déformation imperceptible en barillet, immédiatement après compensée par une déformation en coussinet. L'effet est invisible à l'œil mais présente la caractéristique d'être redoutable pour l'insertion de données.



(a) image originale (b) impression (c) numérisation (d) image attaquée

FIG. 8.1: Attaque par impression & numérisation

8.4.3 Attaques additives

Ajouter un bruit à l'image, bruit blanc ou autre, est une attaque courante. Néanmoins, le rapport signal à bruit doit rester suffisant pour ne pas trop dégrader l'image. Les schémas d'insertion de données travaillant sur les bits de poids faibles sont souvent peu résistants à ce type d'attaque. Cet ajout peut également se faire de manière involontaire : certaines méthodes de requantification y ont recours pour améliorer la qualité résultante.

8.4.4 Attaque par compression

La compression, avec pertes, d'une image fixe est l'attaque la plus fréquente, quelle soit involontaire ou malicieuse. Dès qu'il y a archivage ou transmission à bas débit, il y a forcément une compression mise en œuvre. Le corollaire est que les données insérées y résistent difficilement et le nombre de schémas de compression différents ne facilite pas la tâche.

8.4.5 Attaque par impression et numérisation

La mise en série d'une conversion numérique-analogique et d'une conversion analogique-numérique est une attaque tant involontaire que malicieuse. Pour une image, une mise en œuvre de cette attaque peut consister à imprimer l'image sur papier, puis à la numériser dans la foulée. La figure 8.1 décrit la succession des étapes. Les schémas récents d'insertion de données lui opposent une bonne résistance.

8.4.6 Attaque par ré-échantillonnage

Cette attaque procède en faisant successivement un sur-échantillonnage puis un sous-échantillonnage de l'image attaquée. Cette attaque peut s'assimiler à l'ajout d'un bruit de conversion, dont les caractéristiques statistiques

sont difficiles à représenter. Cette attaque est redoutable pour les méthodes d'insertion de données. Elle est amenée à se répandre par sa présence dans les schémas de compression multirésolution comme JPEG-2000.

8.4.7 À retenir

Au vu des nombreuses attaques existantes, il faut reconnaître que la mesure de la robustesse des méthodes d'insertion de données n'est pas une tâche aisée. Pour comparer les différentes méthodes entre elles, la robustesse face aux attaques peut être évaluée en utilisant des outils synthétiques, qui fournissent un indice de robustesse. L'outil le plus utilisé s'appelle STIRMARK[3]. Son origine provient de l'idée d'une comparaison *fair-play* des méthodes d'insertion de données, telle que présentée par Kutter & Petitcolas (1999). STIRMARK effectue un grand nombre de transformations, de nature diverses, avec des modifications mineures de l'image attaquée. Petitcolas et al. (1998) et Petitcolas (2000) décrivent plus en détails les attaques mises en œuvre ainsi que la manière d'évaluer la robustesse des méthodes attaquées.

8.5 Techniques d'insertion de données

Les méthodes d'insertion de données suivent les grands axes des méthodes de traitement d'images. Ainsi, les méthodes reposant sur les pixels, dans le domaine direct ou spatial, sont apparues avant les méthodes utilisant les coefficients de domaines transformés. Néanmoins, des méthodes proposées, aucune n'a vocation à répondre parfaitement à toutes les attentes liées aux propriétés évoquées à la section 8.3. Les méthodes ciblent plutôt un nombre limité de propriétés, tout en assurant une bonne résilience à certaines classes d'altérations ou d'attaques.

8.5.1 Techniques du domaine spatial

Les techniques d'insertion de données dans le domaine spatial modifient directement les pixels de l'image servant de couverture. Afin de minimiser l'impact visuel de l'insertion, seule la modification des bits de poids faibles est utile. Les méthodes de compression dans le domaine image, comme le codage linéaire prédictif et les représentations fractales, sont également utilisées. Dans ce cas, ce ne sont pas les valeurs des pixels qui sont directement modifiées, mais les paramètres de codage, avec pour objectif de dissimuler les données à insérer dans le bruit lié à la compression.

[3] `http ://www.petitcolas.net/fabien/watermarking/stirmark/`.

8.5.1.1 Modification du ou des bits de poids faibles

Cette technique fait partie des premières méthodes de l'insertion de données. Elle consiste à remplacer directement le bit de poids faible ou LSB (*Least Significant Bit*) des pixels par le bit de la donnée à insérer. Ceci assure une bonne capacité et invisibilité, mais ne procure aucune robustesse.

van Schyndel et al. (1994); van Schyndel (2001) et Tirkel et al. (1995) proposent de pratiquer l'insertion de données dans les LSB par manipulation de plans binaires ou par addition (modulo 2) de l'image et des données à insérer. La détection se fait par corrélation. Matsui & Tanaka (1994) utilisent un codeur linéaire prédictif et cachent les données en les faisant ressembler au bruit de quantification. Celik et al. (2005) généralisent l'emploi des LSB en utilisant plusieurs plans binaires et une quantification adaptée.

Bender et al. (1996, 2000) proposent une technique additive à l'aide de *patchwork* qui modifie les pixels en suivant une approche statistique. L'image est partitionnée en deux ensembles et chaque pixel est modifié différemment suivant son appartenance à l'un ou l'autre des ensembles. Ce partitionnement induit le fait que la quantité d'information insérée est réduite à seul bit, et seule une détection d'insertion de données est possible. Celle-ci s'effectue en calculant la différence entre les moyennes des pixels des deux ensembles. Le tatouage, car s'en est un dans le cas présent, résiste à la plupart des transformations exceptées les modifications géométriques et les attaques par compression.

8.5.1.2 Modification du code fractal

Une technique décrite dans Bas et al. (1998); Bas (2000) établit un code fractal de l'image, une carte des blocs similaires de l'image en quelque sorte, et exploite le fait qu'il est rare de trouver un bloc totalement similaire à un autre dans une image ordinaire. Il est à noter qu'en tant que telles, les images fractales ne sont donc pas considérées comme ordinaires. La méthode ajoute alors des similarités invisibles en contrôlant le code fractal, comme l'illustre la figure 8.2. L'image est partitionnée en blocs 8x8 (comme JPEG) et les blocs à marquer sont choisis en fonction de leurs voisins afin d'éviter les effets de bords. Ainsi, si $\bar{R}$ est la moyenne du bloc destination original R, $\hat{R}$ est le nouveau bloc destination, S l'intensité de l'insertion, et D le domaine de départ, alors :

$$\hat{R} = \delta S \frac{D}{\max(D)} + \bar{R} \tag{8.1}$$

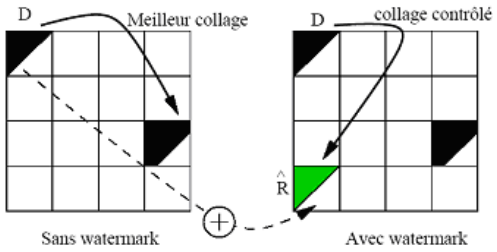


FIG. 8.2: Principe de modification du code fractal

avec

$$\delta = \begin{cases} +1 \text{ si le bit inséré est 1} \\ -1 \text{ si le bit inséré est 0} \end{cases} \tag{8.2}$$

Plus l'intensité de l'insertion est grande, plus les modifications sont importantes, et donc meilleure sera la détection.

Cette méthode est globalement peu robuste, en particulier lors des attaques par compression (JPEG) ou des attaques géométriques et a une faible capacité. Une autre version travaillant dans le domaine transformé (TCD) donne une meilleure résistance à JPEG. Les auteurs parlent aussi de travailler avec le partitionnement triangulaire de Delaunay qui résiste mieux aux modifications géométriques.

Les fractales sont toujours sous investigation. Récemment, Pi et al. (2006) proposent l'insertion d'une séquence binaire pseudo-aléatoire permutée dans les moyennes des ensembles de blocs servant à la transformation. Les auteurs montrent des expérimentations indiquant une bonne résistance face aux transformations géométriques et aux attaques par compression.

8.5.2 Les techniques dans le domaine transformé

Les techniques d'insertion de données dans le domaine transformé sont nombreuses. En fait, il existe au moins autant de méthodes d'insertion qu'il y a de transformations. Chaque technique d'insertion utilise de manière adéquate les propriétés d'invariance propres à chaque transformation pour être robuste face aux attaques. Les transformées les plus utilisées sont les transformées de type fréquentielles. Très souvent, elles sont associées à des techniques d'étalement de spectre, souvent utilisées en télécommunications. Cox et al. (1997) proposent, par ailleurs, d'insérer les données dans les coefficients transformés perceptuellement importants, pour réduire les effets des modifications involontaires et des attaques.

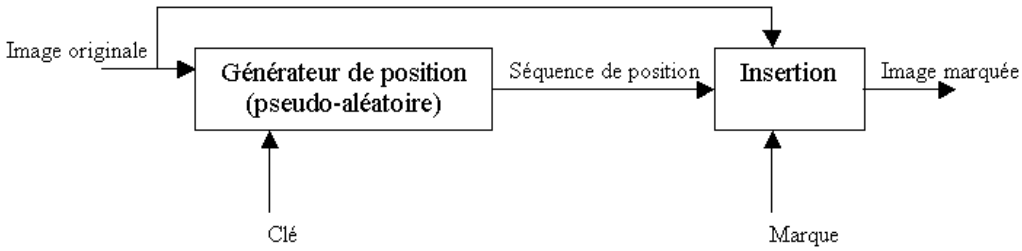


FIG. 8.3: La méthode de Zhao & Koch (1995) par inversion de coefficients

8.5.2.1 Spectre obtenu par transformée en cosinus discret

La transformée en cosinus discret (TCD) fait passer du domaine spatial au domaine fréquentiel réel. Elle effectue une décorrélation du signal et une classification implicite des coefficients transformés. Le fait que JPEG utilise la TCD explique qu'elle serve de base à des techniques d'insertion. Ces techniques peuvent reposer sur des inversions de coefficients, l'insertion d'une séquence binaire pseudo-aléatoire ou de l'étalement de spectre.

Inversion de coefficients TCD : Zhao & Koch (1995) introduisent cette méthode qui consiste à faire des échanges, un seul par bloc à l'origine, de coefficients TCD de façon à les mettre dans un ordre voulu suivant l'information à insérer. Les auteurs appliquent la méthode sur les coefficients de fréquences moyennes, soient au plus 8 coefficients par bloc pour respecter au mieux le critère d'invisibilité et de robustesse. En effet, la modification des coefficients basses fréquences diminue l'invisibilité, tandis que la modification des coefficients hautes fréquences, sensibles au bruit, diminue la robustesse. Les coefficients moyennes fréquences sont un compromis permettant de ménager les deux propriétés. La figure 8.3 présente l'arrangement des données à insérer selon une séquence binaire pseudo-aléatoire.

Cette méthode génère rapidement des distorsions même pour une faible quantité d'information insérée et le compromis invisibilité/robustesse est difficile à trouver. En revanche, la connaissance des coefficients marqués, elle, permet l'extraction des données de manière aveugle.

Intégration d'une séquence pseudo-aléatoire : Cette méthode aveugle développée par Barni et al. (1998) insère une marque pseudo-aléatoire dans des coefficients TCD choisis à l'aide d'un modèle de système visuel humain. Il en résulte une bonne invisibilité et une bonne résistance à la compression. La TCD peut être remplacée par la transformée de Fourier Discrète (TFD)

pour utiliser ses propriétés d'invariance pour obtenir une méthode robuste aux attaques géométriques.

Etalement de spectre TCD Cox et al. (1995) introduit l'étalement de spectre dans la TCD de l'image considérée en entier. La TCD a donc la même taille que l'image. La marque va être répartie sur le spectre entier de l'image, mais avec une très faible amplitude en chaque point. La marque nécessite donc une adaptation avant insertion : elle est sur-échantillonnée pour être de même dimension que le support, puis modulée selon une séquence binaire pseudo-aléatoire, construite à partir d'une clef, pour lui assurer une meilleure indétectabilité et protection. Elle est alors ajoutée au support. La détection est non aveugle et se fait par corrélation entre l'image originale et celle marquée. Cette technique est particulièrement robuste à la plupart des attaques non-géométriques (JPEG qualité à 5%, tramage[4], impression–numérisation, insertions successives) et à quelques attaques géométriques (zoom, changement d'échelle, recadrage) mais ne fait que de la détection. Piva et al. (1997) utilisent le même principe, tout en utilisant des masques psychovisuels pour déterminer les coefficients à utiliser pour insérer les données.

8.5.2.2 Transformée de Fourier Discrète

La TFD fournit une représentation fréquentielle complexe de l'image. Elle possède des propriétés d'invariance par rapport à quelques transformations géométriques, comme le changement d'échelle, la rotation et la translation. Ruanaidh & Pun (1997) donnent davantage de détails. Suivant l'invariance souhaitée, l'intérêt et l'insertion porteront sur le module ou la phase de la TFD. Le spectre obtenu est donc souvent manipulé par étalement de spectre ou par marquage de la phase.

Etalement de spectre Ruanaidh & Pun (1997) pratiquent une insertion par étalement des données insérées sur les points invariants du spectre de l'image couvrante. Ces points sont révélés en utilisant une transformation de FOURIER-MELLIN[5]. La marque peut être détectée sous réserve de posséder la clé de la séquence pseudo-aléatoire qui l'a codée.

La résistance aux transformations telles que rotation, changement d'échelle, translation est conforme à la théorie. L'ajout de bruit, les distorsions non-linéaires d'amplitude, le recadrage jusqu'à 50% et JPEG qualité à 75% n'empêche pas la détection non plus.

[4] *dithering*, en anglais

[5] Une transformation de MELLIN calculée sur le module de la TFD.

Marquage de phase de la TFD Hayes (1992) montre que l'aspect intelligible d'une image dépend davantage de la phase de son spectre que de son amplitude. Forts de ce constat, Ruanaidh et al. (1997) proposent une méthode insérant les données dans la phase du spectre de l'image. Les auteurs tiennent également compte des symétries présentes dans le spectre afin de les conserver. L'intérêt de marquer les zones importantes est que les attaques ne pourront dégrader la marque sans dégrader notablement l'image elle-même. Seule la robustesse à la compression JPEG a été testée, et la technique est résistante jusqu'à un facteur de compression de 15.

8.5.2.3 Distribution de Radon-Wigner

La distribution de Radon-Wigner (DRW) est un outil de outil de détection de données insérées. En effet, le changement de plan effectué par la DRW fait ressortir les signaux de faible amplitude du support qui le contient.

La transformation effectue une corrélation entre l'image marquée et des marques possibles (dépendantes des plans de projection). Le plan correspondant au maxima de la transformée est considéré porteur de la marque.

Derrière le type de marque se cache une idée d'étalement spectral où la détection reste aisée après un recadrage *raisonnable*, c'est-à-dire que les points d'intérêts sont conservés. Le rééchantillonnage entraîne un changement de position des maxima dans la projection. L'idée est alors d'insérer au moins deux signaux. Leurs positions absolues vont changer, mais pas leurs positions relatives.

La méthode est aveugle dans la mesure où les coefficients du plan sont trouvés *sans aide extérieure*. Stanković et al. (2001) proposent deux techniques pouvant être vues comme se situant à la fois dans le domaine spatial et fréquentiel. Dans la première, la marque est composée de deux signaux de type cosinusoïdal 2D (et dont la fréquence varie dans l'espace) qui sont ajoutés à l'image originale. Les propriétés d'une telle marque sont intéressantes, car elle résiste entre autre aux filtres stationnaires. La position relative des deux signaux dans l'image est la clef de l'information cachée.

La seconde technique est une extension de la première, puisqu'elle insère plusieurs signaux d'amplitudes et de positions variant selon une séquence pseudo-aléatoire, codant ainsi l'information. La robustesse dépend de la marque insérée. Avec le type de signaux cité précédemment, une bonne résistance au filtrage stationnaire est obtenue. De plus, les maxima de la DRW et de ses projections ne changent pas après des transformations géométriques telles que translation et rotation.

8.5.2.4 Transformée de Laguerre Discrète

La transformation de Laguerre discrète (TLD) utilise comme base de fonctions les fonctions de Laguerre, fonctions orthonormales. Les fonctions de Laguerre sont construites de manière itérative à partir des polynômes de Laguerre et d'un terme exponentiel.

Gilani & Skodras (2000) décrivent une méthode semblable à celles basées sur les coefficients TCD. La TLD est appliquée à l'image entière ayant subie une pré-transformation spatiale, à savoir un décalage. L'insertion est additive puisque la marque, un signal de type gaussien, est ajoutée à l'image transformée. Le processus de détection est totalement non-aveugle car il requiert non seulement l'image originale mais aussi la marque. Comparativement à la TCD, les performances en robustesse du tatouage basé TLD sont globalement identiques : les tests révèlent une meilleure résistance au bruit mais une moins bonne au sous-échantillonnage. Il faut également préciser que le calcul de la TLD prend davantage de temps qu'une simple TCD.

8.5.2.5 Transformée en ondelettes discrète

Xia et al. (1997) proposent d'insérer par addition une marque pseudo-aléatoire dans les sous-bandes de détails de la transformée en ondelettes discrètes (TOD) de l'image. Cette technique non-aveugle, qui nécessite l'image originale, ne permet qu'une détection de la marque, et ce par intercorrélation des sous-bandes de l'image originale et de l'image marquée, comme illustré sur la figure 8.4.

La technique est robuste à l'ajout de bruit et à toutes les distorsions et transformations que contient la compression par ondelettes. Inoue et al. (1999) présentent une méthode très similaire qui nécessite la marque originale et non l'image. Cette méthode est tout autant robuste que la précédente face à JPEG, JPEG-2000, les filtres passe-bas, les attaques par bruits additifs, le recadrage et les insertions successives.

Toujours dans un contexte de TOD, Zhu et al. (1999) insèrent les données dans tous les coefficients ondelettes hautes fréquences tandis qu'un schéma de détection multirésolution est mis en œuvre.

8.5.2.6 Transformée de Fresnel

Kang & Aoki (1999) proposent d'utiliser les différents plans résultants de la transformation de Fresnel. La marque subit la transformation, et un des plans est inséré dans l'image originale. Seule la phase, élément le plus important pour l'apparence de la marque, est prise en compte, comme le montre la figure 8.5.

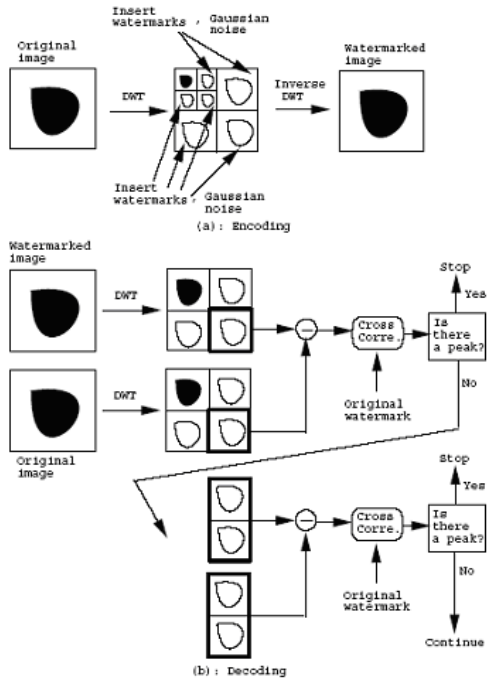


FIG. 8.4: Principe d'insertion et de détection d'après Xia et al. (1997)

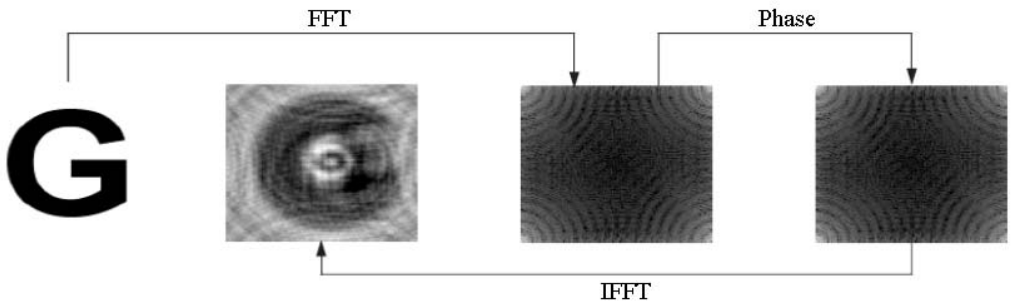


FIG. 8.5: Transformation de Fresnel sur la phase de la marque

La transformée de Fresnel de l'image s est définie comme suit :

$$F(x_2, y_2) = \iint_{R^2} s(x_1, y_1) e^{-\frac{j\pi\left((x_2 - x_1)^2 + (y_2 - y_1)^2\right)}{D}} \, dx_1 dy_1 \qquad (8.3)$$

Il s'agit d'une transformée qui prend en compte la distance et D est une constante non nulle. Cette transformée prend sa source dans l'étude de la diffraction. À partir des propriétés de la convolution et de sa transformation de Fourier, la transformation de Fresnel est simplement la convolution de s avec un *propagateur* $p(x, y)$ donné par :

$$p(x, y) = e^{-\frac{j\pi(x^2 + y^2)}{D}} \qquad (8.4)$$

avec x et y les coordonnées cartésiennes du plan. Ainsi, il vient :

$$F = s * p \qquad (8.5)$$

L'implémentation sur machine est assez simple et repose sur le calcul de TFD uniquement, en utilisant le résultat précédent :

$$F = \mathcal{F}^{-1}[\mathcal{F}[s]\mathcal{F}[p]] \qquad (8.6)$$

L'intérêt de cette transformée est qu'elle génère plusieurs plans de projection qui peuvent être utilisés pour la détection et pas uniquement un plan spécifique. De plus, avec des variations de la distance D, il est possible d'avoir un système multicanal. Les propriétés de la transformée étant proches de celles de la TFD, cette méthode montre une bonne résistance à la rotation, au changement d'échelle, au filtrage passe-bas et médian, ainsi qu'au recadrage.

8.5.2.7 Transformée en S ou extension de la différence

Après avoir appariés les pixels de l'image, Tian & Wells (2004) appliquent une transformée en S (équation 8.7) puis insèrent un bit dans le gradient de chaque paire (équation 8.8). La tranformée inverse est alors appliquée avec la nouvelle valeur du gradient (équation 8.9) pour avoir les nouvelles valeurs des pixels.

Ainsi, soit x et y deux pixels appariés :

$$l = \left\lfloor \frac{x + y}{2} \right\rfloor, h = x - y \qquad (8.7)$$

avec l la moyenne arrondie à l'entier inférieur, et h le gradient. On exprime par h' la nouvelle valeur du gradient :

$$h' = 2 \times h + b \tag{8.8}$$

avec $b \in \{0, 1\}$ le bit à insérer. Les valeurs reconstruites x' et y' sont alors données par :

$$x' = l + \left\lfloor \frac{h' + 1}{2} \right\rfloor, y' = l - \left\lfloor \frac{h'}{2} \right\rfloor. \tag{8.9}$$

L'insertion d'une carte de position des paires marquées, en plus de l'information, permet non seulement une détection mais encore une extraction de données de manière aveugle. De plus cette technique est réversible, c'est-à-dire que l'image originale peut être intégralement restaurée, et semble bénéficier d'un très bon rapport capacité/distorsion. Les auteurs montrent que la capacité maximale d'insertion est d'environ 0.5 bpp, soit une grande capacité d'insertion.

Elle nous intéressera particulièrement dans la suite de l'étude, et sera explicitée plus en détail dans le chapitre 9.

8.6 Conclusion

Les méthodes d'insertion de données sont nombreuses et les résultats obtenus dépendent principalement de la structure choisie ou de la transformée utilisée. Les principales propriétés à surveiller sont la capacité, la robustesse et l'invisibilité.

Ainsi, notre besoin d'ajouter des méta-données aux images nous pousse à choisir une méthode permettant d'insérer une grande quantité de données, avec, néanmoins, une qualité visuelle correcte. De plus, les contraintes de complexité placées à la fois sur le codeur, chargé de réaliser l'insertion, et sur le décodeur, chargé de faire l'extraction, amènent à choisir une méthode de mise en œuvre peu coûteuse. Enfin, nous choisirons une méthode qui s'insère de manière élégante dans le codeur LAR, en particulier en profitant des structures présentes.

Ces trois conditions nous conduisent à considérer la méthode d'insertion de données par extension de la différence pour la suite de nos travaux. En effet, elle s'avère peu gourmande en ressources, manipule des paires de pixels qui pourraientt être prises sur la diagonale du codeur LAR iS+P et permet d'insérer un volume important de données.

Les détails de l'extension de la différence, son adéquation au LAR et les résultats expérimentaux obtenus font l'objet du chapitre suivant.

E. P. JACOBS, *Le secret de l'espadon, tome 2*, page 32, case 5.

9

Extension de la Différence au LAR iS+P

Plan

9.1 Extension de la différence et LAR iS+P
9.2 Intégration de ED au LAR iS+P
9.3 Validation de la méthode
9.4 À retenir

Les images haute résolution de la base d'images EROS du C2RMF nécessitent la manipulation conjointe de metadonnées renseignant entre autres les conditions de la prise de vue. Le codeur LAR IS+P a déjà montré son intérêt, au-dela du codage, pour des services de protection des erreurs et de chiffrement, sélectif, de niveaux de résolution de l'image. Le présent chapitre expose ainsi la mise en œuvre du service d'insertion de données dans les données issues du codeur LAR. Nous justifions l'utilisation de la technique d'insertion de données dénommée *extension de la différence* par l'utilisation de la même transformée en S que le LAR IS+P. La question de l'adaptation ou l'adéquation entre les deux techniques sera suivie par la présentation et l'interprétation de résultats expérimentaux. Nous ferons alors l'analyse des performances en terme de courbes débit-distorsion ainsi que l'évaluation de la qualité visuelle après insertion. De plus, nous verrons en particulier l'évaluation du surcoût de codage lié à l'insertion de données, paramètre particulièrement important pour un codeur d'images fixes. Motsch et al. (2007) présente cette technique et quelques résultats.

9.1 Extension de la différence et LAR IS+P

9.1.1 Similarités entre ED et LAR IS+P

Il s'agit d'intégrer un schéma d'insertion de données dans le codec LAR IS+P. La section 8.5 présente de nombreuses techniques existantes dans ce domaine. Cepdendant, l'application directe d'une de ces méthodes peut nuire aux bonnes propriétés du codeur LAR, à savoir :

- qualité de l'image ;
- faible complexité du codec ;
- faible coût entropique.

L'idée première est donc d'utiliser une technique d'insertion de données opérant de manière compatible avec le codeur. Le codeur LAR Interleaved S+P utilise la transformée en S pour décorréler les deux diagonales de blocs 2×2. Parmi les méthodes d'insertion de données, la méthode d'extension de la différence procède également à partir de la transformée en S des niveaux de gris d'une paire de pixels. De plus, cette méthode présente l'intérêt d'une insertion de grande capacité (jusqu'à 0.5 bpp pour une passe) et l'avantage d'une assez faible complexité algorithmique.

La section 9.1.2 présente donc l'extension de la différence avec davantages de détails.

9.1.2 L'extension de la différence

9.1.2.1 Principe et exemple

Comme il a été vu à la section 8.5.2.7, l'extension de la différence ou ED, est une méthode réversible d'insertion de données. Elle repose sur la transformée en S qui est une transformée réversible. Prenons ainsi un exemple complet de réalisation de cette méthode.

Soient les deux pixels appariés de luminance respective $x = 206$ et $y = 201$. Leur transformation en S donne :

$$l = \left\lfloor \frac{206 + 201}{2} \right\rfloor = \left\lfloor \frac{407}{2} \right\rfloor = 203, h = 206 - 201 = 5. \tag{9.1}$$

Soit le bit $b = 1$ à insérer, la valeur du gradient devient alors :

$$h' = 2 \times h + b = 2 \times 5 + 1 = 11. \tag{9.2}$$

La transformation inverse donne les nouvelles valeurs des deux pixels

$$x' = 203 + \left\lfloor \frac{11 + 1}{2} \right\rfloor = 209, y' = 203 - \left\lfloor \frac{11}{2} \right\rfloor = 198. \tag{9.3}$$

La transformée est, par définition, appliquée aux deux pixels :

$$l' = \left\lfloor \frac{209 + 198}{2} \right\rfloor = 203, h' = 209 - 198 = 11. \tag{9.4}$$

Le bit b est naturellement extrait du gradient. On peut alors obtenir la valeur originale de celui-ci par les opérations suivantes :

$$b = \mathrm{LSB}(h') = 1, \quad h = \left\lfloor \frac{h'}{2} \right\rfloor = 5. \tag{9.5}$$

Il ne reste plus qu'à appliquer la transformation inverse pour obtenir les pixels originaux :

$$x = 203 + \left\lfloor \frac{5 + 1}{2} \right\rfloor = 206, y = 203 - \left\lfloor \frac{5}{2} \right\rfloor = 201. \tag{9.6}$$

L'opération réversible d'insertion $h' = 2 \times h + b$ est l'*extension de la différence* ou ED.

Il convient tout de même d'assurer que l'insertion du bit n'engendre pas de dépassement de capacité. En effet, sur une image dont les pixels sont codés sur 8 bits, il faut respecter la condition :

$$0 \leq l + \left\lfloor \frac{h' + 1}{2} \right\rfloor \leq 255, \text{et } 0 \leq l - \left\lfloor \frac{h'}{2} \right\rfloor \leq 255. \tag{9.7}$$

l et $h' = 2 \times h + b$ étant des entiers, la condition 9.7 est équivalente à :

$$\begin{cases} |h'| \leq 2(255 - l), \text{ si } 128 \leq l \leq 255 \\ |h'| \leq 2l + 1, \text{ si } 0 \leq l \leq 127 \end{cases} \tag{9.8}$$

et ce quelle que soit la valeur de $b \in \{0, 1\}$.

Pour prévenir le dépassement de capacité, la condition 9.7 revient ainsi à dire que :

$$|2 \times h + b| \leq \min(2(255 - l), 2l + 1), \forall b \in \{0, 1\} \tag{9.9}$$

Lorsqu'un entier est mis sous la forme suivante :

$$h' = 2 \times \left\lfloor \frac{h'}{2} \right\rfloor + \text{LSB}(h') \tag{9.10}$$

il est possible de modifier son bit de poids faible sans provoquer de dépassement de capacité, à la condition que :

$$\left| 2 \times \left\lfloor \frac{h}{2} \right\rfloor + b \right| \leq \min(2(255 - l), 2l + 1), \forall b \in \{0, 1\} \tag{9.11}$$

Ces conditions permettent de répartir les paires de pixels en plusieurs catégories. Ainsi, on notera les deux définitions suivantes :

Définition 1 (Extensible)**.** Le gradient h d'une paire de pixels qui répond à la condition 9.9 est dit « extensible » sous l.

Définition 2 (Changeable)**.** Le gradient h d'une paire de pixels qui répond à la condition 9.11 est dit « changeable » sous l.

À partir de ces définitions, il est démontré quatre propriétés à retenir.

---

1. Un gradient h changeable le reste après modification de son LSB.
2. Un gradient h extensible est toujours changeable.
3. Après application de l'ED, un gradient étendu h' est changeable.
4. Si un gradient h vaut 0 ou -1, les conditions d'extensibilité et de changeabilité sont équivalentes ($2 \times h + b = 2 \times \left\lfloor \frac{h}{2} \right\rfloor + b$).

Ces propriétés méritent quelques explications. L'utilisation de gradients extensibles (qui sont en général largement majoritaires dans les images naturelles) comme support pour l'insertion va donc, d'après la propriété 3, les rendre changeables. Le décodeur ne pourra alors plus faire la distinction entre

les gradients devenus changeables et ceux nativement changeables, et va donc collecter les LSB de tous les changeables. La propriété 1 permet d'utiliser les changeables natifs comme support à travers leurs LSB. Mais, si ils étaient simplement modifiés, les LSB originaux ne pourraient être retrouvés. Pour obtenir une méthode réversible, ils seront « sauvegardés » dans un flot qui sera concaténé à la charge utile.

Tout ceci ne règle pas totalement le problème initial du décodeur : faire la distinction des extensibles et des changeables. Par conséquent, il lui faut une carte de position des deux types de gradients, carte qui sera concaténée au flot précédent. Ainsi, une fois les LSB des changeables collectés, le décodeur pourra récupérer la carte des positions. Il connaîtra alors la position des extensibles et des changeables, puis il pourra restaurer les valeurs originales de tous les gradients en fonction du bit récupéré sur un gradient extensible, et du LSB original collecté dans le flot pour un gradient changeable.

La méthode est en fait un peu plus complexe. En effet, il n'est pas nécessaire de considérer tous les extensibles comme support si la charge utile est inférieure à la capacité totale. Les effets d'un gradient changé étant beaucoup moins visibles que ceux d'un gradient étendu (car, sauf dans le cas de la propriété 4 où il y a égalité parfaite, $(2 \times \lfloor \frac{h}{2} \rfloor + b)$ sera toujours plus proche de h que $(2 \times h + b)$), il est préférable pour la qualité de l'image après insertion que le plus d'extensibles possibles soient considérés comme changeables. La carte de position distinguera alors d'un côté les extensibles choisis pour l'ED, et donc réellement étendus, et de l'autre les extensibles changés et les changeables changés. La propriété 4 fait que le résultat de l'insertion dans h sera le même quelle que soit la considération de h. Dans ce cas, il s'avère plus judicieux de considérer le gradient comme extensible : cette solution coûte moins cher (pas de nomlsb à stocker) et permettra la mise en œuvre d'une méthode originale (qui sera expliquée plus loin) pour réduire, dans un cas bien précis, la taille du flot contenant les LSB.

9.1.2.2 Codage

L'algorithme d'insertion de données par l'extension de la différence se déroule en six étapes. La figure 9.1 présente la succession de ces étapes.

1. Les pixels de l'image originale sont appariés suivant un balayage et un modèle à notre convenance, puis la transformée en S est appliquée aux paires de pixels. Les gradients h_n obtenus sont rangés dans une liste unidimensionnelle $\{h_1, h_2, \cdots, h_n\}$;
2. Chacun de ces gradients est affecté à une et une seule des quatre catégories suivantes :

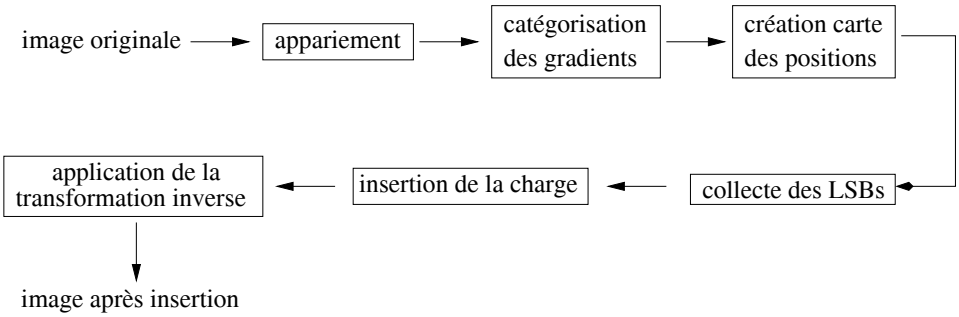


FIG. 9.1: étapes de la méthode d'extension de la différence

- EZ : contient tous les extensibles qui valent 0 ou -1 ;
- EN : contient tous les extensibles qui n'appartiennent pas à EZ ;
- CN : contient tous les changeables natifs ;
- NC : contient tous les non-changeables ;

3. La carte de position est créée. Tous les EZ sont sélectionnés pour la DE, plus tout ou partie des EN suivant la taille de la charge utile. Les EN peuvent donc être séparés en deux sous-catégories :
   - EN1 : les EN qui seront étendus ;
   - EN2 : les EN qui seront changés ;

   Le choix de la limite entre EN1 et EN2 sera expliqué plus loin. En résumé, la carte de position se voit affectée d'un '1' lorsqu'un gradient appartient à EZ $\cup$ EN1 et d'un '0' lorsqu'il appartient à EN2 $\cup$ CN $\cup$ NC. La carte binaire ainsi obtenue est compressée par un codeur type JBIG2 ou run-length. Le flot obtenu est noté $\mathcal{L}$;
4. Les LSB des gradients appartenants à EN2 $\cup$ CN sont collectés dans le flot noté $\mathcal{C}$, sauf pour ceux dont les valeurs sont 1 et -2 car elles pourront être déterminées grâce à la carte de position (explication dans la section suivante) ;
5. Soit $\mathcal{P}$ la charge utile que l'on veut insérer. Le flot $\mathcal{B} = \mathcal{L}$ + marqueur « fin de $\mathcal{L}$ » + $\mathcal{C}$ + $\mathcal{P}$ est inséré selon l'algorithme 1.
6. Enfin, la transformée inverse est appliquée pour obtenir l'image marquée.

9.1.2.3 Discussions

À partir de la définition de la méthode et de ses propriétés mathématiques, il est possible de déterminer quelques caractéristiques comme la capacité, ou comment déterminer la limite entre EN1 et EN2, par exemple.

Algorithme 1 Insertion de la charge utile

Require: i, j
  $i \leftarrow 1$
  $j \leftarrow 0$
  while ($i \leq |\mathcal{B}|$) **do**
    $j \leftarrow j + 1$
    if ($h_j \in (EZ \cup EN1)$) **then**
      $h_j \leftarrow 2 \times h_j + b_i$
      $i \leftarrow i + 1$
    else
      if ($h_j \in (EN2 \cup CN)$) **then**
        $h_j \leftarrow 2 \times \left\lfloor \frac{h_j}{2} \right\rfloor + b_i$
        $i \leftarrow i + 1$
      end if
    end if
  end while

Capacité La longueur du flot total $\mathcal{B}$ est égale à $|\mathcal{L}| + |\mathcal{C}| + |\mathcal{P}|$. D'après l'algorithme d'insertion, la capacité totale vaut $|EZ|+|EN1|+|EN2|+|CN|$. Donc, pour que $\mathcal{B}$ puisse être inséré entièrement, il faut que :

$$|\mathcal{L}| + |\mathcal{C}| + |\mathcal{P}| \leq |EZ| + |EN1| + |EN2| + |CN| \tag{9.12}$$

Soit N le nombre de gradients de valeur $h = 1$ et $h = -2$ dans $EN2 \cup CN$, alors :

$$|\mathcal{P}| \leq |EZ| + |EN1| + N - |\mathcal{L}| \tag{9.13}$$

Ainsi, la taille de la charge utile est maximisée par la somme du nombre de gradients sélectionnés pour la DE plus le nombre de gradients valant 1 ou -2 moins la taille de la carte de position.

Détermination de la limite EN1/EN2 Soit une paire de pixels (x, y). Soit (x', y') la nouvelle paire après DE de (x, y). Comme la valeur moyenne l reste inchangée,

$$x - x' \approx y - y' \tag{9.14}$$

l'approximation étant du à la fonction d'arrondi par défaut. On a alors :

$$\begin{aligned}(x - x')^2 + (y - y')^2 &\approx 2 \times (y - y')^2 \\ &= 2 \times \left(\left\lfloor \frac{h}{2} \right\rfloor - \left\lfloor \frac{h'}{2} \right\rfloor \right)^2 \\ &\approx \frac{h^2}{2}\end{aligned}$$

Ainsi, la distance euclidienne entre (x, y) et (x', y') est proportionnelle au gradient h. Pour minimiser les distorsions sur l'image marquée, il est préférable de choisir des gradients de faible amplitude absolue.
Soit T un seuil qui sépare EN1 de EN2 :

$$\begin{cases} EN1 = \{h \in EN : |h| \leq T\} \\ EN2 = \{h \in EN : |h| > T\} \end{cases} \tag{9.15}$$

Le principe repose sur un système itératif, qui commence avec une faible valeur de T et l'incrémente à chaque boucle jusqu'à ce que la limite de la condition 9.13 soit atteinte.

9.1.2.4 Décodage

L'opération de décodage peut se résumer en cinq étapes :

1. La transformation en S est appliquée aux pixels appariés suivant un balayage et un modèle définis lors de l'insertion des données ;
2. Chaque gradient de la liste obtenue $\{h_1, \cdots, h_n\}$ est affecté à une et une seule des deux catégories suivantes :
   – CH : tous les gradients changeables ;
   – NC : tous les gradients non-changeables ;
   Pour l'image *marquée*, le flot CH est identique à $EZ \cup EN \cup CN$. Cette propriété invariante est la clé pour la reconstruction exacte ;
3. Le LSB des gradients dans CH est soustrait de ses gradients respectifs et collecté pour former le flot binaire $\mathcal{B}$;
4. Le flot $\mathcal{L}$ inclus dans $\mathcal{B}$ (du début de $\mathcal{B}$ jusqu'au marqueur de fin) est décodé pour obtenir la carte de position. Soit s la taille de $\mathcal{L}$ dans $\mathcal{B}$, plus la taille du marqueur de fin. Les gradients originaux sont alors restaurés selon l'algorithme 2.
   Si la valeur de la carte est *1*, ceci signifie que le gradient a été étendu lors de l'insertion. Inversement, un gradient non-changeable doit avoir *0* dans la carte, sinon l'image a été nécessairement altérée. Pour un gradient h changeable, si sa valeur dans la carte est *0*, alors sa valeur originale sera égale ou différente de h au LSB près. Ainsi, si $h = 0$ ou $h = 1$, sa valeur originale vaut forcément 1 : si elle valait 0, ce serait un 0 extensible (un 0 changeable est extensible), et sa valeur dans la carte serait *1*. De la même façon, si $h = -1$ ou $h = -2$, sa valeur originale vaut forcément -2. Pour les autres gradients changeables, leurs LSB sont restaurés grâce au flot $\mathcal{C}$. Une fois que toutes les valeurs ont été restaurées, les bits restants dans $\mathcal{B}$ forment la charge insérée $\mathcal{P}$;

5. Enfin la transformée inverse est appliquée aux gradients reconstruits et à leurs moyennes, qui elles n'ont jamais été modifiées.

Algorithme 2 Reconstruction des gradients originaux

Require: i
  $i \leftarrow s+1$
  for ($j \leftarrow 1$ à $|\mathcal{B}|$) **do**
    if ($h_j \in CH$) **then**
      if (valeur de la carte de position à l'indice $h_j = 1$) **then**
        $h_j \leftarrow \left\lfloor \frac{h_j}{2} \right\rfloor$
      else
        if ($0 \leq h_j \leq 1$) **then**
          $h_j \leftarrow 1$
        else
          if ($-2 \leq h_j \leq -1$) **then**
            $h_j \leftarrow -2$
          else
            $h_j \leftarrow 2 \times \left\lfloor \frac{h_j}{2} \right\rfloor + b_i$
            $i \leftarrow i+1$
          end if
        end if
      end if
    end if
  end for

9.2 Intégration de ED au LAR iS+P

L'insertion de données par ED et le codeur LAR iS+P utilisent la même transformée en S. Cette similitude permet d'utiliser de manière intégré l'insertion de données dans le codeur. L'idée principale est de piloter l'insertion de données à l'aide de la partition quadtree. La figure 9.2 illustre ce principe de pilotage. À côté de cette similitude, quelques différences existent. Elles concernent d'une part la manière d'apparier les pixels et d'autre part l'adaptation à la charge utile demandée.

9.2.1 Appariement des pixels

Dans la technique d'insertion de données par DE, l'information est portée par le gradient de paires de pixels. Le choix du motif d'appariement est laissé

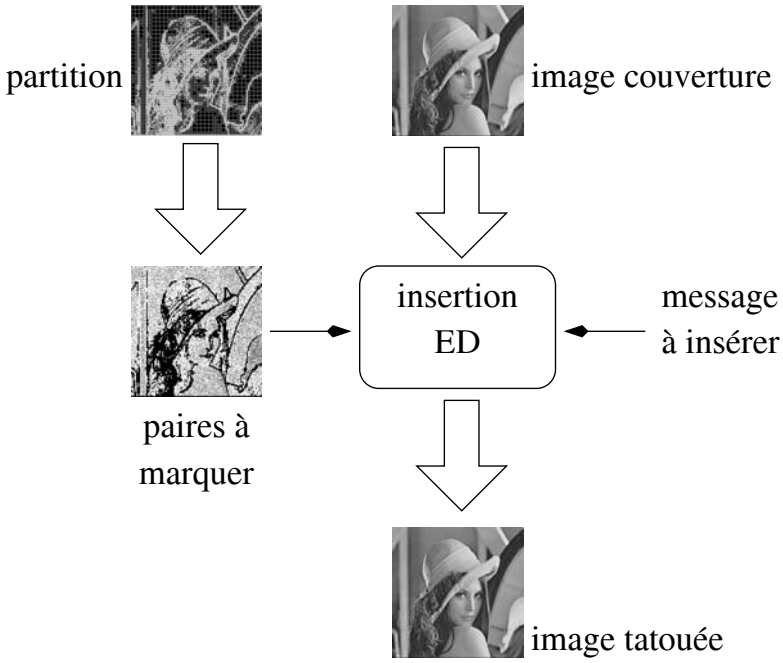


FIG. 9.2: Pilotage de l'insertion ED par la partition quadtree du LAR IS+P.

libre dans la mise en œuvre. En fait, tout dépend des critères que l'on se donne, et notamment du compromis capacité/distorsion. En effet, plus il y aura de paires marquées, et plus les distorsions apparaîtront dans l'image. Les zones marquées de l'image ont aussi un rôle important : en effet, le système visuel humain étant moins sensible dans les zones de contours que dans les zones uniformes, ces dernières devraient donc faire plus apparaître les distorsions. C'est pourquoi, dans un premier temps, la zone de tatouage est limitée aux blocs 2x2 fournis par la partition du LAR.

9.2.2 Adaptation à la charge utile

Une des caractéristiques du LAR est aussi sa rapidité d'exécution. Il faut donc veiller à ce que la méthode de tatouage reste raisonnable en termes de complexité algorithmique. Or, le système itératif de sélection des gradients extensibles qui serviront de support, effectuant l'ajustement de la taille du support à la charge utile, peut prendre beaucoup de temps. Pour éviter cela, le système itératif d'adaptation à la charge ne sera pas utilisé. Le système de substitution mis en place est alors le suivant : le système suppose, pour commencer, que toute la capacité est utilisée, c'est-à-dire que, d'après la section 9.1.2, $EN1 = EN \Rightarrow EN2 = 0$. À partir de ce postulat, la carte de

position est établie et compressée, le flot de LSB est créé. Le codeur connaît alors la taille de la carte ainsi que celle des LSB. En soustrayant ceci à la capacité totale, est obtenue la capacité de la charge utile correspondante, qui constitue par ailleurs la charge maximale admissible. Mais la charge utile réelle peut être moindre. Dans ce cas, il existe deux possibilités :

- mettre un marqueur de fin à la charge réelle et remplir l'espace restant par, par exemple, du bruit. Dans ce cas la capacité maximale est toujours utilisée, ce qui va inévitablement dégrader l'image inutilement ;
- mettre un marqueur de fin à la charge réelle, et donc à la fin du flot à insérer. Toutefois, la fin de la carte de position ne correspond plus au flot inséré. Le décodeur est alors modifié en conséquence et il ignorera la fin la carte. Cependant, si le décodeur s'arrête une fois la reconstruction de la seule partie du flot réellement modifiée, il n'aura pas moyen de déterminer la limite entre le flot de LSB éventuellement restant et le flot de la charge. Pour palier cela, deux solutions sont envisageables :
  - Mettre en entête du flot des LSB sa taille ;
  - Laisser se dérouler l'algorithme de reconstruction en comptant le nombre de passages par la ligne $h_j = 2 \times \left\lfloor \frac{h_j}{2} \right\rfloor + b_i$, sans exécution de celle-ci. En effet, dans ce cas, les gradients concernés n'ont pas été modifiés. Le décodeur calcule donc la taille du flot. Ceci augmente de fait la complexité du décodeur, mais n'augmente pas la taille du flot à insérer.

  Cette dernière solution est celle que nous utiliserons par la suite.

9.3 Validation de la méthode

Le codeur LAR découpe l'image originale en blocs de taille $\mathcal{T}$ = 2x2, 4x4, 8x8 et 16x16, en utilisant un partitionnement quadtree dépendant de l'activité locale, comme l'a montré le chapitre 1. La partition, déjà utilisée par la suite, est la pierre d'angle du codage, en permettant l'ajout d'une sémantique liée au contenu de l'image.

Les expérimentations se feront suivant trois directions. D'abord, en utilisant les paires de pixels dans les diagonales des différents blocs, quelle est la quantité d'information utile, la charge utile, que l'on peut insérer dans l'image ? Ensuite, quels seront les effets de cette insertion sur les images, en termes de distorsion et de dégradation de la qualité visuelle due à cette insertion ? Enfin, l'insertion de données venant perturber le bon fonctionnement du codeur, quel sera le surcoût occasionné dans le processus de codage ?

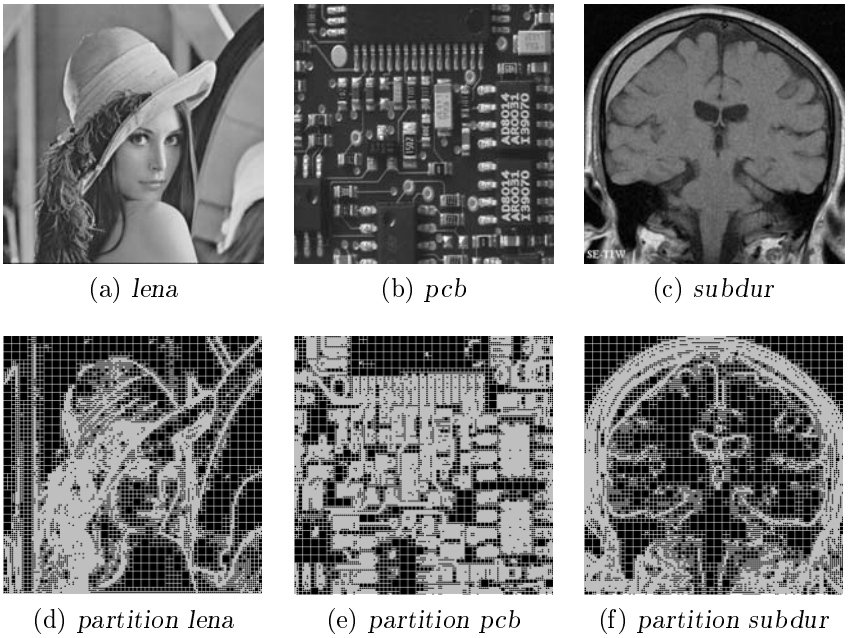
Nous ne présentons que quelques résultats représentatifs obtenus. En particulier, nous montrerons les résultats correspondants aux images *lena*, *pcb* et

subdur. Sans prétendre à l'exhaustivité, ces images ont des caractéristiques très différentes et permettent d'illustrer les performances de la méthode d'insertion par extension de la différence couplée au codeur LAR.

L'insertion se fera dans l'image pleine résolution, sur les blocs 2x2, 4x4, 8x8 et 16x16, indépendemment. Ensuite, une insertion combinée sur les blocs 4x4, 8x8 et 16x16 sera effectuée, afin d'augmenter la quantité d'information insérée. La charge utile insérée, le *payload* $\mathcal{P}$ est une séquence binaire pseudo-aléatoire. En effet, pour des raisons évidentes de sécurité, les données insérées sont chiffrées au préalable, à l'aide d'une primitive de chiffrement par bloc ou par flot comme celles présentées dans le chapitre 4. Or, les données chiffrées présentent une entropie informationnelle importante, comme pour un flot binaire pseudo-aléatoire. Même imparfaite, la fonction C `rand` est suffisante pour une telle application.

En termes de charge utile, nous considérerons le scenario suivant : insérer la quantité maximale de données comme charge utile $\mathcal{P}$. Cette quantité dépend de l'image et des blocs choisis comme support d'insertion.

La figure 9.3 montre les images originales et les partitions quadtree associées. Les images sont toutes de dimensions 512×512.



(a) *lena* (b) *pcb* (c) *subdur*

(d) *partition lena* (e) *partition pcb* (f) *partition subdur*

FIG. 9.3: images originales et leurs partitions quadtree associées

Blocs	EZ	EN	CN	NC	$\|\Omega\|$ bits	$\|\mathcal{L}\|$ bits	$\mathcal{P}_{\max}$ bits	$\mathcal{P}_{\max}$ cbpp
2×2	1759	32394	663	0	34816	7011	27142	10.3
4×4	4042	26806	0	0	30848	994	29854	11.4
8×8	6502	26138	0	0	32640	1042	31598	12.0
16×16	7929	24839	0	0	32768	1058	31710	12.1
4×4 à 16×16	18473	77783	0	0	96256	3042	93214	35.6

TAB. 9.1: Évaluation des catégories et capacités pour l'image *lena*

9.3.1 Évaluation de la capacité d'insertion

Pour les différentes images, nous avons évalué les paramètres suivants :
- le nombre de gradients de chaque catégorie :
  - EZ : extensibles valant 0 ou 1 ;
  - EN : les autres extensibles ;
  - CN : les changeables ;
  - NC : les autres ;
- la capacité totale $|\Omega|$;
- la taille de la carte de position, compressée par un codeur à longueur de plage adapté, $|\mathcal{L}|$;
- la charge utile de taille $|\mathcal{P}_{\max}|$.

Les résultats sont donnés soit en bits, soit en bits par pixels (bpp) ou en centièmes de bits par pixels (cbpp). Rapellons qu'étant donné qu'un bit peut être inséré par paire (ou diagonale), la capacité ne peut excéder 0.5 bpp, soit 50 cbpp. Les tables 9.1, 9.2 et 9.3 donnent les valeurs de capacités pour les images *lena*, *pcv* et *subdur*, respectivement.

On peut remarquer que ce sont les gradients des diagonales des blocs 2×2 qui peuvent participer le plus à l'insertion de données. Cela peut s'expliquer par le fait que les blocs de cette taille sont les plus nombreux. Cependant, leur capacité d'insertion n'est pas très grande, car le coût de codage de la carte des positions correspondante est plus élevé que pour les autres tailles de blocs. Cela peut s'expliquer de la manière suivante. La carte des positions note la position des gradients changés dans l'image. C'est une carte binaire, codée par une technique de longueur de plage. Pour des blocs 2×2, la distribution statistique des gradients modifiés est telle que son entropie est plus importante que pour les autres tailles de blocs. Par conséquent, son codage se fait de manière moins efficace, grevant d'autant la charge utile.

Les résultats de cette section amène donc à effectuer le choix suivant : l'insertion d'une grande quantité de données, comme 30 cbpp, passe par l'utilisation conjointe des gradients des diagonales des blocs de taille supérieure

Blocs	EZ	EN	CN	NC	$\|\Omega\|$ bits	$\|\mathcal{L}\|$ bits	$\mathcal{P}_{\max}$ bits	$\mathcal{P}_{\max}$ cbpp
2×2	2747	53737	1842	66	58326	22218	36108	13.8
4×4	6766	21292	13	1	28071	1082	26977	10.3
8×8	8011	16373	0	0	24383	786	23598	9.0
16×16	6102	14122	0	0	20224	658	19566	7.5
4×4 à 16×16	20879	51787	13	1	72679	2474	70193	26.8

TAB. 9.2: Évaluation des catégories et capacités pour l'image *psb*

Blocs	EZ	EN	CN	NC	$\|\Omega\|$ bits	$\|\mathcal{L}\|$ bits	$\mathcal{P}_{\max}$ bits	$\mathcal{P}_{\max}$ cbpp
2×2	1795	30673	1112	36	33580	12942	19528	7.4
4×4	4286	27259	119	0	31664	2458	29087	11.1
8×8	6149	23608	67	0	29824	1786	27971	10.7
16×16	8524	27310	134	0	35968	1930	32904	12.6
4×4 à 16×16	18959	78177	320	1	97456	7010	90126	34.4

TAB. 9.3: Évaluation des catégories et capacités pour l'image *subdur*

à 2, soient de la taille 4×4 à la taille 16×16. Ceci nous fournira également une plage de débits plus importante, lorsque la charge utile est moindre.

9.3.2 Performances capacité-distorsion

La figure 9.4 présente les courbes capacité-distorsion pour le LAR IS+P+ED, l'ED originale (équivalente au LAR+ED effectué pour tous les blocs), l'ED hiérarchique et la méthode G-LSB introduite par Celik et al. (2002) pour l'image *lena*. LAR IS+P+ED est considéré pour plusieurs tailles de blocs. On peut remarquer qu'il y a davantage de distorsions lorsque seuls les blocs 2x2 sont marqués. Ceci est confirmé par des expérimentations sur d'autres images. Pour les autres tailles de blocs, LAR IS+P+ED est meilleur que G-LSB et parfois meilleur que l'ED hiérarchique (itérative), principalement pour les petites charges utiles. Dans cette configuration, l'ED originale ne fournit pas de bons résultats.

L'utilisation de la méthode d'extension de la différence en l'adaptant au codeur LAR IS+P nous permet d'obtenir de bons résultats avec des distorsions raisonnables pour une capacité d'insertion élevée.

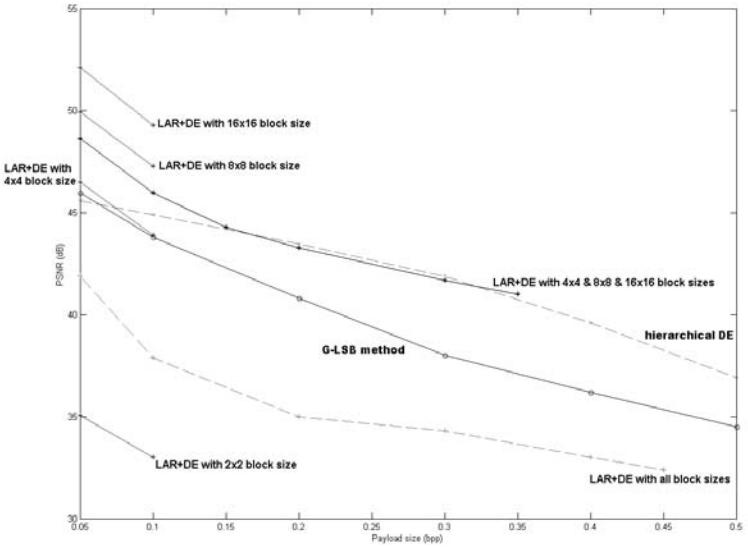


FIG. 9.4: Distorsion en fonction de la capacité pour *lena*. Comparaison des méthodes LAR IS+P+ED, G-SLB et ED original et hiérarchique.

9.3.3 Évaluation de la qualité visuelle

Les figures 9.5, 9.6 et 9.7 permettent de voir les différents effets sur la qualité visuelle de l'insertion de données, avec les charges utiles les plus grandes possibles.

D'une manière générale, les dégradations visibles sont relativement faibles et peu perceptibles. Cependant, il apparaît que, lorsque les blocs 2×2 servent de support à l'insertion, celle-ci introduit davantage de distorsions que pour les autres tailles de blocs. Ceci se vérifie pour toutes les images traitées. Ainsi, l'idée de départ, à savoir considérer comme principal support les blocs 2×2 pour s'adapter aux caractéristiques du système visuel humain, n'est pas tant pertinente que cela. Au contraire, plus les blocs servant de support sont grands, et plus les distorsions sont contenues à des dégradations visuelles peu importantes. De plus, même en prenant comme support les blocs 4x4, 8x8 et 16x16 simultanément et en insérant la charge maximale possible, les distorsions sont moindres que lors de l'insertion sur les blocs 2x2 uniquement.

La dégradation constatée lors de l'insertion sur les blocs 2×2 peut s'expliquer de la manière suivante. Il s'agit d'un effet inter-blocs : les petits blocs délimitant un contour, et les gradients de ces blocs étant presque toujours extensibles, la modification des valeurs de ces pixels sans tenir compte des

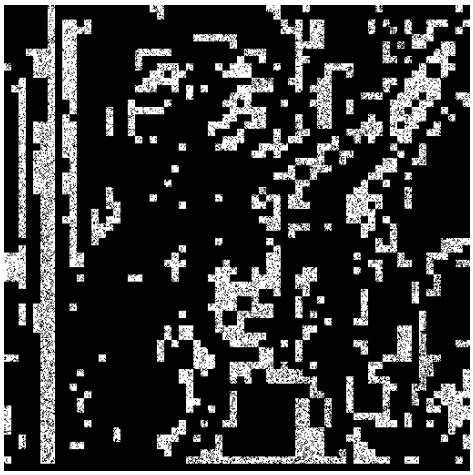
(a) blocs 2×2, 27142 bits insérés

(b) blocs 2×2

(c) blocs 4×4, 29854 bits insérés

(d) blocs 4×4

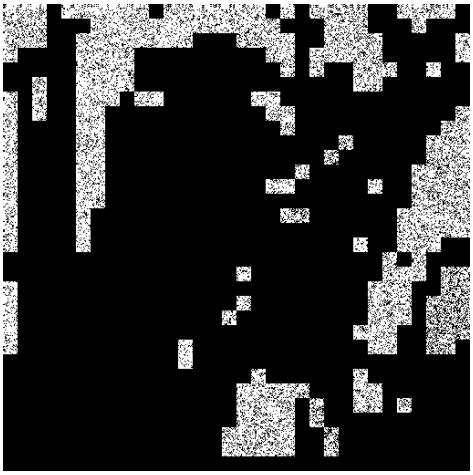
(e) blocs 8×8, 31598 bits insérés



(f) blocs 8×8

FIG. 9.5: Évolution de la qualité visuelle selon la taille de blocs tatoués, image *lena*. Pour chaque image, on trouve en regard les positions de pixels qui ont été modifiés (en blanc sur fond noir)

(g) blocs 16×16, 31710 bits insérés

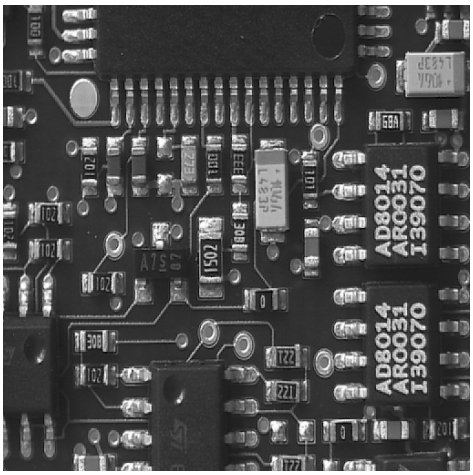


(h) blocs 16×16

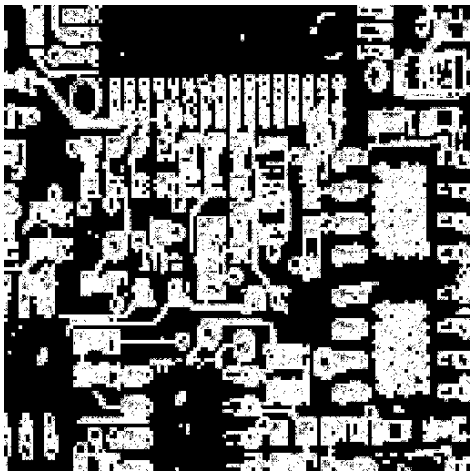
(i) blocs 4×4 à 16×16, 93214 bits insérés

(j) blocs 4×4 à 16×16

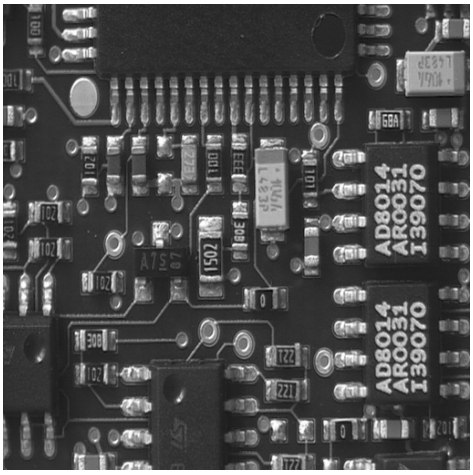
FIG. 9.5: Évolution de la qualité visuelle selon la taille de blocs tatoués, image *lena*. Pour chaque image, on trouve en regard les positions de pixels qui ont été modifiés (en blanc sur fond noir) (suite et fin)



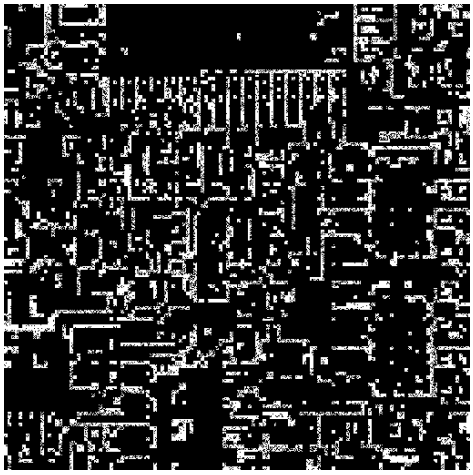
(a) blocs 2×2, 36108 bits insérés



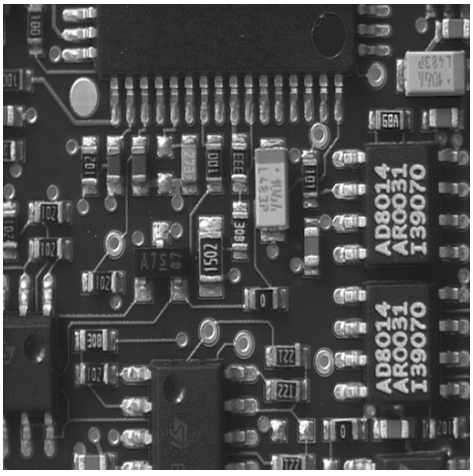
(b) blocs 2×2



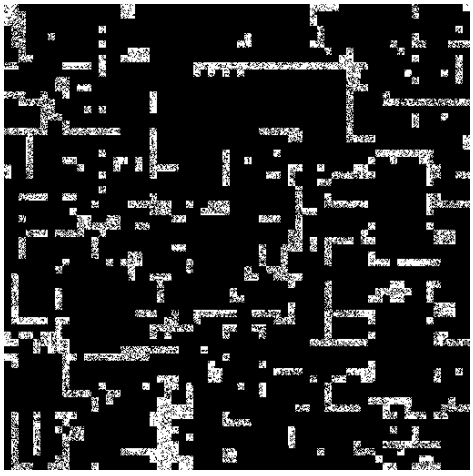
(c) blocs 4×4, 26977 bits insérés



(d) blocs 4×4

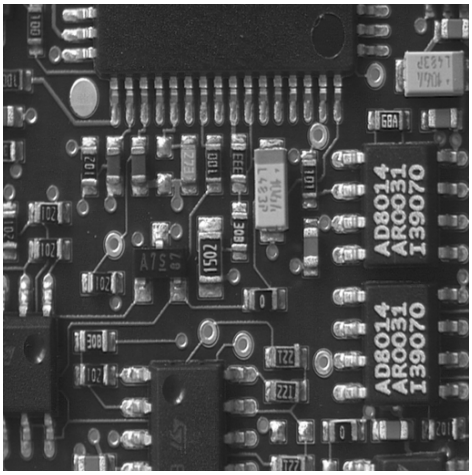


(e) blocs 8×8, 23598 bits insérés

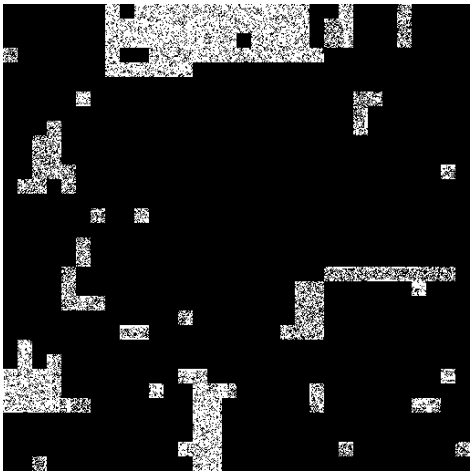


(f) blocs 8×8

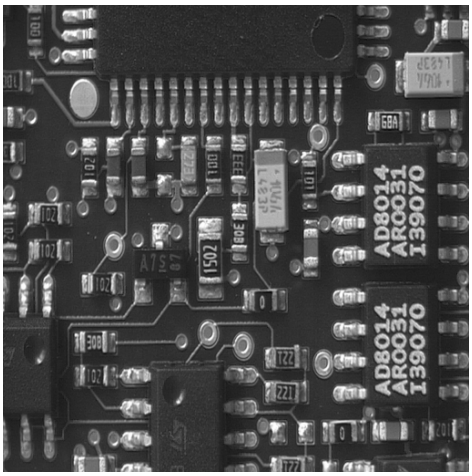
FIG. 9.6: Évolution de la qualité visuelle selon la taille de blocs tatoués, image *pcb*. Pour chaque image, on trouve en regard les positions de pixels qui ont été modifiés (en blanc sur fond noir)



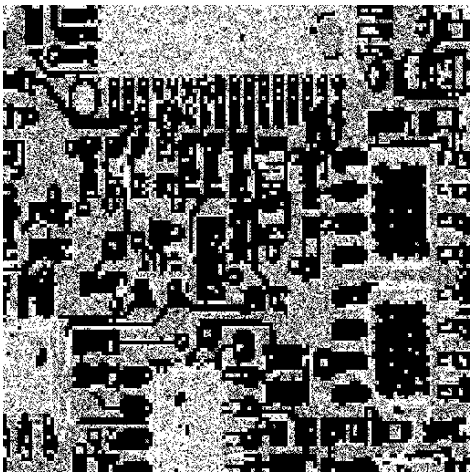
(g) blocs 16×16, 19566 bits insérés



(h) blocs 16×16

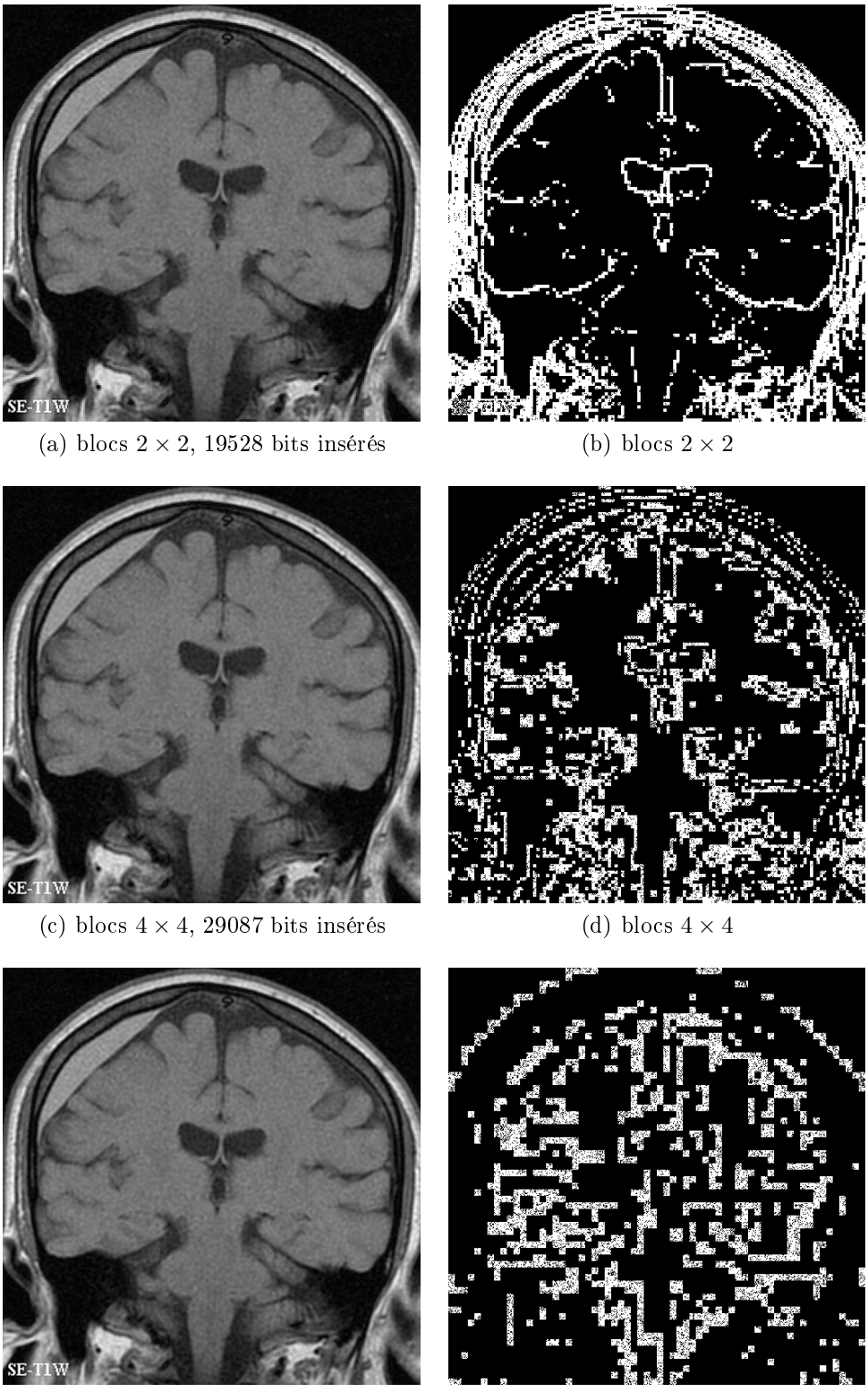


(i) blocs 4×4 à 16×16, 70193 bits insérés



(j) blocs 4×4 à 16×16

FIG. 9.6: Évolution de la qualité visuelle selon la taille de blocs tatoués, image *pcb*. Pour chaque image, on trouve en regard les positions de pixels qui ont été modifiés (en blanc sur fond noir) (suite et fin)

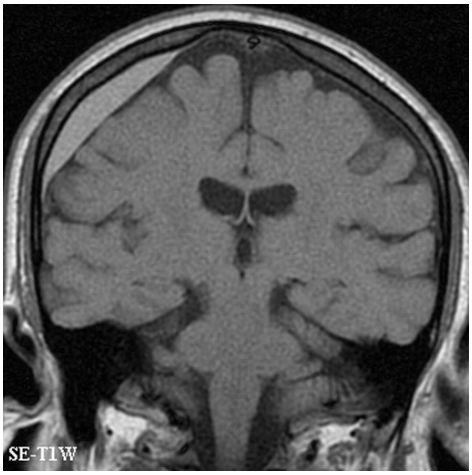


(a) blocs 2×2, 19528 bits insérés (b) blocs 2×2

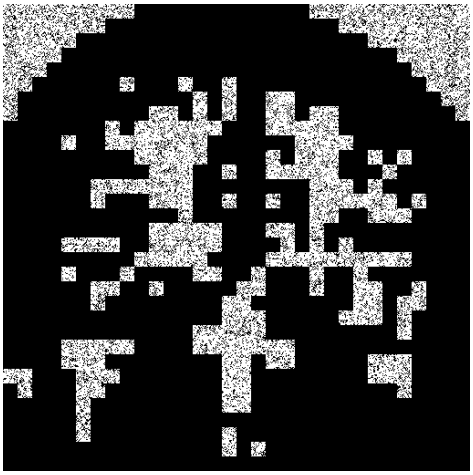
(c) blocs 4×4, 29087 bits insérés (d) blocs 4×4

(e) blocs 8×8, 27971 bits insérés (f) blocs 8×8

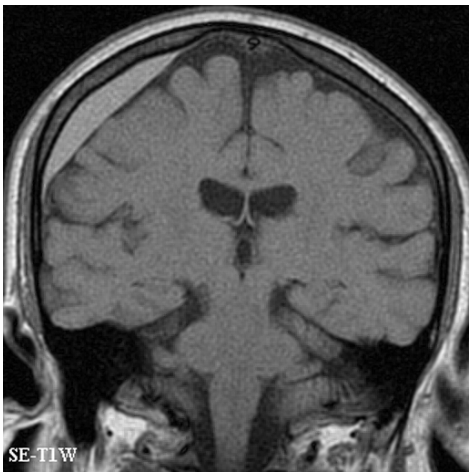
FIG. 9.7: Évolution de la qualité visuelle selon la taille de blocs tatoués, image *subdur*. Pour chaque image, on trouve en regard les positions de pixels qui ont été modifiés (en blanc sur fond noir)



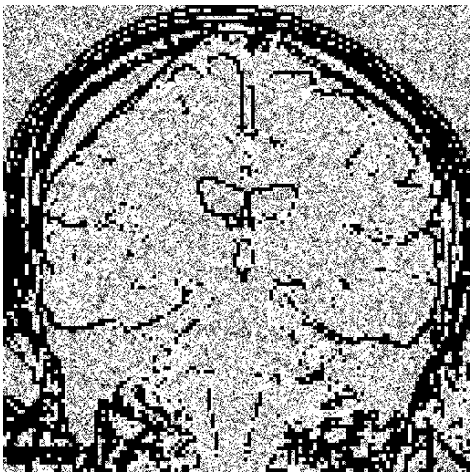
(g) blocs 16×16, 32904 bits insérés



(h) blocs 16×16



(i) blocs 4×4 à 16×16, 90126 bits insérés



(j) blocs 4×4 à 16×16

FIG. 9.7: Évolution de la qualité visuelle selon la taille de blocs tatoués, image *subdur*. Pour chaque image, on trouve en regard les positions de pixels qui ont été modifiés (en blanc sur fond noir) (suite et fin)

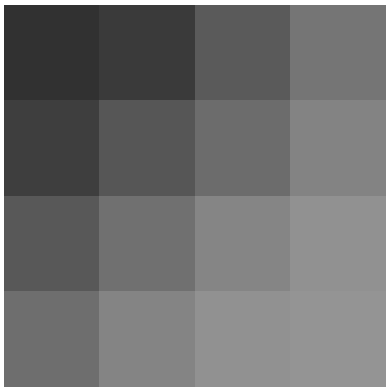
valeurs des pixels adjacents entraîne un déplacement local du contour. Pour illustrer ce phénomène, considérons l'exemple suivant : une zone formée de quatre blocs 2x2, dont les deux ayant les paires de pixels originales sur leur premières diagonales de valeurs respectives $(49, 86)$ et $(133, 148)$ seront modifiés, car extensibles (en italique sur la figure 9.8a). Soient les deux bits à insérer $b = 1$ et $b = 0$ respectivement.

49	**58**	**90**	117
62	***86***	108	131
88	112	*133*	145
110	132	145	*148*

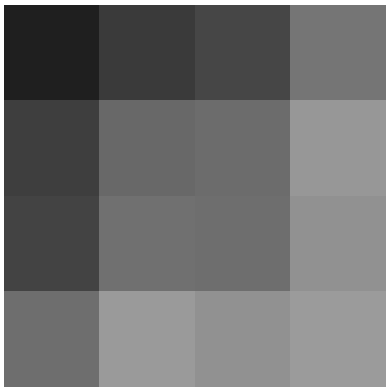
(a) Zone originale

31	**58**	**70**	117
62	*104*	108	151
67	112	***125***	145
110	154	145	155

(b) Zone modifiée après DE



(c) Zone originale



(d) Zone modifiée

FIG. 9.8: Effet de la DE sur les zones de contour

Les pixels les plus foncés ont été mis en gras, entre autre afin de mieux discerner le contour. Le résultat après ED montre un décalage du contour : un pixel clair a naturellement tendance à foncer alors qu'il se trouve toujours parmi la zone claire (figure 9.8b).

La modification d'un seul bloc 2x2 parmi plusieurs blocs 2x2 adjacents n'a donc pas de conséquence visuelle notable. En revanche la modification de plusieurs blocs 2x2 adjacents a un effet cumulatif qui fait apparaître de fortes distorsions, dont un aspect *pixel*.

9.3.4 Influence sur les performances de codage

L'évaluation de l'impact de l'insertion de données sur les performances de compression passe par le calcul de la taille du flot binaire nécessaire à coder sans perte les différentes images. Le tableau 9.4 résume les résultats obtenus pour l'image *lena*. La charge utile étant issue d'une séquence binaire pseudo-aléatoire, son coût de codage minimal correspond à sa longueur. De même, il faut également prendre en compte le surcoût de l'ED lié à la carte des positions.

taille bloc	charge utile [cbpp]	surcoût ED [cbpp]	coût codage [bpp]	surcoût codage [cbpp]	surcoût codage p/r charge utile [%]
2	10	3	4.50	6	60
4	11	0.4	4.51	8	73
8	12	0.4	4.51	8	67
16	12	0.4	4.49	7	58
4–16	36	1	4.73	5	14

TAB. 9.4: Coût de codage pour le LAR+ED calculé pour l'image *lena*. L'entropie brute de l'image originale compressée par le LAR-iSP s'élève à 4.31 bpp.

La caractéristique majeure du tableau 9.4 concerne les surcoûts liés au codage, c'est-à-dire le coût engendré par l'insertion de données. Ajouté au surcoût lié à l'extension de la différence, il donne le surcoût total du LAR+ED. Le surcoût de l'extension de la différence ayant été vu à la section 9.3.1, seul le surcoût lié au codage est ici discuté.

Le surcoût de codage reflète principalement l'adapation (ou non) du prédicteur aux nouvelles valeurs des gradients modifiés par l'insertion de données. Le tableau 9.4 montre que le surcoût de codage est directement lié à la charge utile. Plus nombreux sont les pixels marqués, ou, de manière équivalente, plus nombreux sont les gradients modifiés, meilleur sera le prédicteur. De fait, la distribution des gradients s'uniformise et permet au prédicteur de mieux fonctionner. Ainsi, l'insertion de 0.36 bpp de données n'induit-elle un surcoût de codage de 0.05 bpp, soit seulement 14 % de la charge utile.

Cette étude sur le surcoût de codage lié à l'insertion de données est un aspect original de notre travail. En effet, l'impact de l'insertion de données sur l'efficacité de codage n'est, semble-t'il, jamais pris en compte dans la littérature.

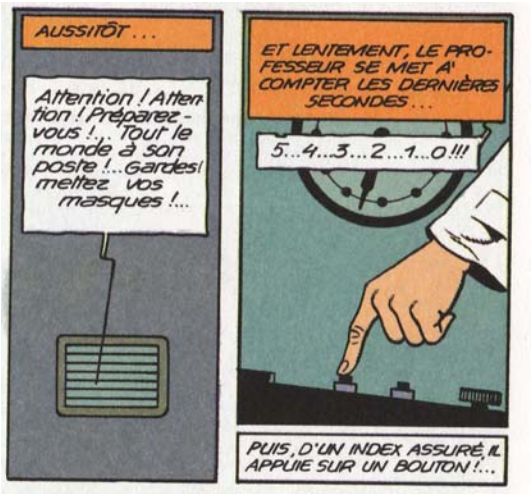
9.4 À retenir

L'insertion de données par extension de la différence est une méthode d'insertion permettant d'obtenir une grande capacité pour une charge utile importante. Reposant sur la même transformation que le codeur LAR IS+P, l'intégration de cette insertion se fait de manière naturelle, tant du côté de la mise en œuvre que de celui des performances obtenues.

Parmi les enseignements tirés de l'expérimentation, le fait qu'il faille insérer les données dans les blocs de taille supérieure à 2 est sans conteste le plus important. Alors qu'une approche simple tendait à utiliser les blocs 2×2 présents sur les contours pour insérer les données, la dégradation visuelle observée, aisément expliquable *a posteriori*, a contredit cette hypothèse de travail.

L'autre enseignement, important du point de vue de l'efficacité de codage, est le faible surcoût engendré par l'insertion de données. Ce surcoût provient de deux origines. Le premier facteur en est la carte des positions, qu'il faut coder et qui vient réduire d'autant la place disponible pour la charge utile. Le second est la baisse, toute relative, des performances du codeur en présence des données insérées, analogues à du bruit.

Nous avons donc montré tout l'intérêt qu'il y avait à coupler une méthode d'insertion à un codeur d'image fixe performant, utilisant les meilleures propriétés de chacun.



E. P. Jacobs, *SOS météores*, page 55, cases 7 et 8.

Conclusions et perspectives

L'introduction avait précisé les différentes nécessités afférentes à l'archivage sécurisé d'images :

- un codeur d'images fixes performant ;
- un service assurant la confidentialité des données ;
- un service autorisant l'ajout de metadonnées.

La famille des codeurs LAR, en particulier le codeur LAR iS+P, a montré toute sa pertinence pour faire de la compression avec et sans pertes. L'objectif des travaux présentés dans ce manuscrit a donc été de proposer une intégration de nouveaux services dans le LAR iS+P, à travers l'ajout de primitives nouvelles. Ces primitives concernent d'une part le chiffrement de l'image et d'autre part l'insertion de données. Dans tous les cas, la structure même du codeur LAR iS+P est mise à profit pour obtenir une bonne adéquation avec la primitive concernée. En particulier, nous avons rappelé l'importance de la partition quadtree ainsi que son implication nécessaire dans le décodage de l'image. Cette partition sert non seulement à une modélisation simple mais efficace du contenu sémantique de l'image, mais également de pierre d'angle pour la reconstruction de l'image.

Dans un premier temps, nous décrirons brièvement les services ajoutés dans une optique d'archivage au codeur LAR iS+P, tandis que dans un second temps, nous présenterons des améliorations envisagées ainsi que des perspectives de recherche dans la continuation des présents travaux.

Synthèse des travaux menés

Chiffrement partiel à coût nul

Le chiffrement partiel procède en séparant l'image en deux sous-flots binaires. Cela peut consister en une scission entre le *flat*-LAR et la texture locale, ou en une séparation entre la partition et les résidus de la prédiction. Le premier sous-flot apparaît comme indispensable pour le décodage de l'image et c'est donc lui qui sera chiffré. Nous nous sommes servis de ce paradigme en considérant que la partition quadtree du codeur LAR iS+P représentait l'information capitale pour la reconstruction de l'image originale.

En enlevant cette information du flot binaire, nous réalisons, à coût nul, un chiffrement de l'image. Nous avons montré l'extrême difficulté à retrouver l'image en l'absence de la partition. De même, la sécurité inconditionnelle semble être vérifiée. Évidemment, dans ce domaine, toutes les attaques ne sont pas connues et il y a là matière à poursuivre nos activités. Ainsi, nous avons proposé une méthode de chiffrement dans laquelle la partition de l'image elle-même sert de clef de chiffrement.

Chiffrement sélectif des flots LAR iS+P

La deuxième primitive de chiffrement concerne le chiffrement sélectif des sous-flots résultants du codeur LAR iS+P. Nous avons intégré au codeur des primitives de chiffrement de données, par bloc ou par flot. L'aspect pyramidal du codeur a été utilisé pour faire un chiffrement différencié suivant le niveau de la pyramide. L'idée sous-jacente est de permettre un accès hiérarchique à l'image.

La méthode de chiffrement que nous proposons respecte la compatibilité du flot chiffré avec le codeur LAR iS+P. Le flot chiffré reste donc décodable. Dans ce contexte, nous avons évalué les performances du chiffrement en termes de PSNR, mais également en termes de métriques modélisant le système visuel humain. Les résultats obtenus montrent que cette méthode de chiffrement sélectif rend particulièrement inintelligible le contenu de l'image et cela pour une faible complexité calculatoire.

Insertion de données

La dernière primitive intégrée au codeur LAR iS+P concerne l'insertion de données, en utilisant l'extension de la différence. C'est un exemple d'utilisation conjointe de deux méthodes reposant sur une transformation commune, la transformée en S. Ce point commun permet d'expliquer la qualité des résultats obtenus.

Nous avons d'abord évalué les performances de l'insertion de données, en termes de capacité et de courbes capacité-distorsion. L'enseignement majeur que nous en avons tiré est que l'insertion doit se faire sur les blocs de taille supérieure ou égale à 4×4. De même, il est possible d'utiliser plusieurs tailles de blocs pour augmenter la capacité.

Nous nous sommes ensuite attachés à évaluer l'influence de l'insertion de données sur les performances de compression. Nous avons montré que le surcoût de codage lié à l'insertion était relativement restreint. On peut penser que cela provient de l'usage de la même transformée et de l'efficacité des prédicteurs.

Perspectives

Sécurité du chiffrement partiel à coût nul

La sécurité du chiffrement partiel à coût nul repose sur l'absence d'information mutuelle entre le flot binaire correspondant à la partition quadtree et les sous-flots binaires correspondants aux descentes de la pyramide LAR iS+P. Quelques pistes, théoriques et expérimentales, peuvent être proposées pour vérifier cette condition.

D'abord, il serait pertinent de mesurer le niveau d'aléa des sous-flots binaires codant les deux descentes de la pyramide. Cette mesure peut se faire en évaluant son entropie ou bien en utilisant les tests usuels de statistique, comme le test du χ_2 et la distance de KOLMOGOROV-SMIRNOV. Ces tests présentent l'intérêt d'être à la fois des outils expérimentaux, capables de fournir des indications chiffrées pour un ensemble d'images, et des outils théoriques, permettant d'obtenir des résultats généraux, propres à valider une sécurité calculatoire.

Ensuite, la mesure de l'information mutuelle entre les deux flots peut s'avérer riche d'enseignements. Cette mesure est difficile, car mettant en œuvre un grand nombre de symboles. Néanmoins, elle permettrait de donner des éléments théoriques concernant la sécurité du chiffrement partiel proposé.

Fonction de hachage

Peut-on utiliser la partition quadtree comme fonction de hachage ? La réponse à cette question passe par l'évaluation du taux de collisions pour une grande base d'images. En cas de réponse favorable, nous pourrions utiliser l'aspect hiérarchique du codage de la partition quadtree dans le LAR IS+P pour pouvoir faire une indexation hiérarchique d'images. Cela permettrait d'ajouter un service supplémentaire lié à l'archivage d'images.

Gestion des clefs du chiffrement sélectif

Notre méthode de chiffrement sélectif de flots LAR IS+P ne propose pas de méthode de distribution des clefs. En particulier, faut-il un tiers de confiance pour gérer l'accès aux clefs ? Tout récemment, l'idée d'échanger des masques jetables a été expérimentée et s'est avérée pertinente. Ces problématiques s'éloignent du codeur LAR IS+P et des services associés pour faire place à l'étude d'une architecture globale d'archivage d'images.

Méthode jointe d'insertion et prédiction de données

Nous avons constaté que l'insertion de données par la méthode LAR IS+P+ED se fait sans un surcoût de codage excessif. Pour l'instant, cette insertion se fait dans le codeur, mais en amont de la prédiction. Une piste d'investigation est de faire l'insertion de données conjointement à la prédiction des coefficients transformés. Cette prédiction pilotée par l'insertion présenterait plusieurs avantages. D'abord, cela simplifierait la structure du codeur et du décodeur, en intégrant les deux fonctions. Ensuite, l'insertion pourrait être pilotée par le contexte de la prédiction. En effet, pour l'instant, l'insertion de données se fait sur les pixels situés sur les diagonales des blocs de taille prédéfinie. L'insertion conjointe permettrait d'avoir un contrôle plus fin de l'insertion, particulièrement en termes de distorsion. Enfin, un schéma unifié simplifierait les structures du codeur et du décodeur.

On le voit, les pistes à explorer sont presque plus longues que le travail réalisé jusqu'à présent. L'idée première demeure : fournir les services propres à permettre la mise en place d'un outil d'archivage performant, reposant sur un codeur d'images fixes efficace, des primitives de chiffrement robustes et l'ajout de metadonnées pertinentes.

Table des figures

1 une image vue en basse et haute résolution 4

1.1 Schéma global LAR à deux couches : codeurs spatial et spectral 11
1.2 Schéma de principe du codeur spatial 12
1.3 Résultats pour une partition $\mathrm{QP}^{[16..2]}$. 15
1.4 Schéma de principe du codeur spectral pour une partition $QP^{[16..2]}$ 16
1.5 Principe de la progressivité sémantique 17
1.6 Principe de l'approche pyramidale prédictive 20
1.7 Construction *bottom-up* 21
1.8 Construction incomplète de la pyramide 21
1.9 décomposition *top-down* de la pyramide 22

2.1 Application particulière de la transformée en S sur une image pleine résolution. 27
2.2 Construction des deux pyramides en S entrelacées. 28

3.1 arbre binaire et ses parcours 38
3.2 Étapes de codage d'une partition $\mathrm{QP}^{[16..2]}$. 39
3.3 Structure du flot binaire LAR 40
3.4 image *lena* et la partition quadtree $\mathrm{QP}^{[16..2]}$ associée 41
3.5 Première étape du codage de la partition : remplacement d'un bloc 16×16 par un bloc 2×2 dans le coin en haut à gauche. . 41
3.6 Première étape du codage de la partition : remplacement d'un bloc 2×2, 4×4 ou 8×8 par un bloc 16×16 aux deux tiers de la première ligne de blocs. 42
3.7 Deuxième étape du codage de la partition : remplacement d'un bloc 8×8 par un bloc 2×2 situé aux deux tiers de la première ligne de blocs. 43
3.8 Troisième étape du codage de la partition : remplacement d'un bloc 2×2 par un bloc 4×4 situé en haut à droite. 43

3.9 Partition inconnue au décodeur : hypothèse d'une partition uniforme constituée de blocs 16×16. 44
3.10 Propagation intra-niveau des erreurs 45
3.11 Valeur d'erreur de prédiction erronée observée sur le pixel $Y_{l_T}(0,0) = Y_4(0,0)$. 45
3.12 Valeur d'erreur de prédiction erronée observée sur un pixel du niveau $l_T = 4$. 46
3.13 Observation de la propagation intra-niveau de l'erreur. 47
3.14 Observation de la propagation complète de l'erreur (intra- et inter-niveau) appliquée sur l'image LAR basse résolution : décomposition totale de la pyramide. 48
3.15 Observation de la propagation complète de l'erreur (intra- et inter-niveau) appliquée sur l'information de texture : décomposition totale de la pyramide. 49
3.16 Exemple d'images reconstruites après réception d'une valeur erronée. 49

4.1 Principe du chiffrement 59
4.2 Principe du chiffrement symétrique 60
4.3 Principe du chiffrement asymétrique 61
4.4 Chiffrement et déchiffrement par bloc : mode ECB. 65
4.5 Chiffrement et déchiffrement par bloc, mode CBC. 66
4.6 Chiffrement et déchiffrement par bloc, mode CFB. 67
4.7 Chiffrement et déchiffrement par bloc, mode OFB. 67
4.8 Chiffrement et déchiffrement par bloc, mode CTR. 68

5.1 Chiffrement d'une image 74
5.2 Principe du chiffrement partiel d'image 76
5.3 Principe du chiffrement sélectif 78

6.1 Proposition de schéma de masquage d'image par niveaux . . . 96
6.2 Entropie de la partition LAR en fonction du seuil d'activité . . 100
6.3 Entropie de la partition LAR en fonction du nombre de blocs . 101
6.4 Fonction entropie binaire $H_2(p)$. 103

7.1 Flots binaires LAR chiffrés 109
7.2 Partitions issues des flots des niveaux en sortie du codeur pour l'image *lena*. 110
7.3 Images *lena* reconstruites en sortie du décodeur après chiffrement (RC4, clef de 256 bits) d'un niveau donné de la partition. 112
7.4 Images *lena* reconstruites en sortie du décodeur après chiffrement (AES, clef de 256 bits) d'un niveau donné de la partition. 113

7.5 Image *lena* chiffrée au niveau 3 par RC4 et AES avec des clefs de tailles différentes. 114
7.6 Image *baboon* avec le niveau 4 chiffré, RC4 et clef de 512 bits, et déchiffrée avec une clef erronée. 118
7.7 Vue de SynDEX, fenêtre principale, présentant l'ajout du chiffrement des flots du LAR flat. 120
7.8 Chiffrement RC4 de la partition et du flat de l'image *baboon* pour différents niveaux. 121
7.9 Comparaison du chiffrement de la partition et du chiffrement combiné de la partition et *flat* pour le niveau 4 122
7.9 Comparaison du chiffrement de la partition et du chiffrement combiné de la partition et *flat* pour le niveau 4 (suite et fin). . 123
7.10 Image *lena* déchiffrée (RC4) après modification au décodeur d'une valeur initiale. 127
7.11 Visualisation de régions sombres ou claires sur l'image *lena* déchiffrée (RC4) après modification d'un des paramètres au décodeur. 127
7.12 Chiffrement RC4 combiné partition, flat et vecteur d'initialisation sur image lena pour différents niveaux 128
7.13 Chiffrement AES combiné partition, flat et vecteur d'initialisation sur l'image *lena* pour différents niveaux 129

8.1 Attaque par impression & numérisation 142
8.2 Principe de modification du code fractal 145
8.3 La méthode de Zhao & Koch (1995) par inversion de coefficients 146
8.4 Principe d'insertion et de détection d'après Xia et al. (1997) . 150
8.5 Transformation de Fresnel sur la phase de la marque 150

9.1 étapes de la méthode d'extension de la différence 158
9.2 Pilotage de l'insertion ED par la partition quadtree du LAR iS+P. 162
9.3 images originales et leurs partitions quadtree associées 164
9.4 Distorsion en fonction de la capacité pour *lena*. Comparaison des méthodes LAR iS+P+ED, G-SLB et ED original et hiérarchique. 167
9.5 Évolution de la qualité visuelle selon la taille de blocs tatoués, image *lena*. Pour chaque image, on trouve en regard les positions de pixels qui ont été modifiés (en blanc sur fond noir) . . 168

9.5 Évolution de la qualité visuelle selon la taille de blocs tatoués, image *lena*. Pour chaque image, on trouve en regard les positions de pixels qui ont été modifiés (en blanc sur fond noir) (suite et fin) 169
9.6 Évolution de la qualité visuelle selon la taille de blocs tatoués, image *pcb*. Pour chaque image, on trouve en regard les positions de pixels qui ont été modifiés (en blanc sur fond noir) . . 170
9.6 Évolution de la qualité visuelle selon la taille de blocs tatoués, image *pcb*. Pour chaque image, on trouve en regard les positions de pixels qui ont été modifiés (en blanc sur fond noir) (suite et fin) 171
9.7 Évolution de la qualité visuelle selon la taille de blocs tatoués, image *subdur*. Pour chaque image, on trouve en regard les positions de pixels qui ont été modifiés (en blanc sur fond noir) 172
9.7 Évolution de la qualité visuelle selon la taille de blocs tatoués, image *subdur*. Pour chaque image, on trouve en regard les positions de pixels qui ont été modifiés (en blanc sur fond noir) (suite et fin) 173
9.8 Effet de la DE sur les zones de contour 174

Liste des tableaux

1.1 Entropie [bpp] pour le LAR-APP, CALIC et S+P. 22

2.1 Entropies comparées des flots binaires de compression sans pertes obtenues par LAR iS+P, CALIC et S+P 33
2.2 Entropie de l'image *lena* codée sans perte (le seuil valant 20) par niveau de la pyramide, par numéro de passe, par coefficient et par taille. Entropie totale 4.31 bpp. 34
2.3 Nombre de blocs concernés pour chaque niveau, selon la passe et la nature du coefficient pour l'image *lena* 34

5.1 Tableau comparatif des propriétés des différentes techniques de chiffrement d'images présentées à la section 5.6 93

6.1 $\Phi(n)$ et $\Phi_a(n)$ pour les premières valeurs de n 98
6.2 Entropie des niveaux de la partition LAR en [bits] pour des images de 512×512 en fonctions des tailles de blocs. 101
6.3 Nombre de partitions P_{32} du passage du niveau du haut au suivant. 103

7.1 Chiffrement RC4 de la partition : résultats sur image *lena* pour différentes tailles de clef. CG est le gain composé, KOMPARATOR fournit une mesure objective et une note subjective. . . . 116
7.2 Chiffrement AES de la partition : résultats sur image *lena* pour différentes tailles de clef. 117
7.3 Chiffrement RC4 de la partition et du flat : mesure de la distorsion (PSNR), du gain composé (CG) et des notes de KOMPARATOR (Mobj et Nobj) pour l'image lena pour différentes tailles de clef 125
7.4 Comparaison des PSNR entre le chiffrement simple de la partition et celui combiné partition et *flat*. 126

7.5 Comparaison des PSNR entre le chiffrement simple de la partition, combiné partition et flat et combiné partition, flat et vecteur d'initialisation pour l'image *lena* 130

9.1 Évaluation des catégories et capacités pour l'image *lena* 165
9.2 Évaluation des catégories et capacités pour l'image *psb* 166
9.3 Évaluation des catégories et capacités pour l'image *subdur* . . 166
9.4 Coût de codage pour le LAR+ED calculé pour l'image *lena*. L'entropie brute de l'image originale compressée par le LAR-iSP s'élève à 4.31 bpp. 175

Table des matières

I Codage d'images fixes **7**

1 Codeur d'images LAR **9**
1.1 Principe de codage à deux couches 10
1.2 Codeur LAR simple 11
1.2.1 Codeur spatial *flat*-LAR 11
1.2.2 Le codeur spectral 16
1.2.3 Pour la suite 17
1.3 Approche pyramidale prédictive 18
1.3.1 Prédicteur de Wu 18
1.3.2 Principes généraux de l'approche pyramidale prédictive 19
1.3.3 Construction de la pyramide 19
1.3.4 Décomposition pyramidale à contexte enrichi 21
1.4 À retenir 23

2 Codeur d'images LAR Interleaved S+P **25**
2.1 Principes généraux du LAR iS+P 26
2.2 Transformée en S et prédicteur de Wu 28
2.2.1 Éléments de notations 29
2.2.2 Moyenne de la première diagonale 29
2.3 Jeu de prédicteurs adaptés 30
2.3.1 Prédicteurs pour la première pyramide en S – image des blocs 30
2.3.2 Prédicteurs pour la seconde pyramide en S – image des blocs 30
2.4 Prédiction de la texture locale 31
2.4.1 Première pyramide en S – texture 31
2.4.2 Deuxième pyramide en S – texture 32
2.5 Performances en compression sans perte 32
2.6 À retenir 35

3 Influence de la partition quadtree **37**
3.1 Codage de la partition quadtree 38
3.2 Structure du flot binaire LAR 39
3.3 Influence des erreurs du flot binaire 40
3.3.1 Erreur dans la partition 40
3.3.2 Erreur sur la valeur d'un résidu de prédiction 44
3.4 À retenir 50

II Chiffrement d'images **51**

4 Chiffrement de données **53**
4.1 Crypto-vocabulaire 54
4.2 Sécurité des cryptosystèmes 55
4.2.1 Principes 55
4.2.2 Cryptanalyse 57
4.3 Structures de chiffrement 59
4.3.1 Chiffrement symétrique 59
4.3.2 Chiffrement asymétrique 61
4.3.3 Chiffrement hybride 62
4.3.4 Quelle taille de clef utiliser ? 62
4.4 Chiffrement par flot 63
4.5 Chiffrement par bloc 63
4.5.1 Réseaux de Feistel 64
4.5.2 Modes de fonctionnement 64
4.6 Primitives de chiffrement 68
4.6.1 Méthode DES/3DES 69
4.6.2 Méthode AES 69
4.6.3 Méthode RC4 70
4.6.4 Méthode RSA 70
4.7 À retenir 71

5 Chiffrement d'images **73**
5.1 Introduction 74
5.2 Chiffrement partiel 76
5.3 Chiffrement sélectif 77
5.4 Analyse 78
5.5 Évaluations des performances 79
5.5.1 Coût temporel 79
5.5.2 Sécurité 79
5.5.3 Conformité du flot binaire 80

5.5.4 Traitement dans le domaine compressé 80
5.5.5 Effet sur les performances de compression 80
5.6 Techniques de chiffrement d'images 80
5.6.1 Techniques dans le domaine image 81
5.6.2 Techniques dans le domaine transformé 81
5.6.3 Techniques orientée flot binaire 87
5.6.4 Comparaison des différentes techniques de chiffrement d'images 92
5.7 Normes de sécurité pour les images 92
5.8 À retenir 94

6 Chiffrement partiel à coût nul **95**
6.1 Principe du chiffrement à coût nul 96
6.2 Du nombre de partitions LAR 97
6.2.1 Approche théorique 97
6.2.2 Validation expérimentale 99
6.2.3 Analyse 102
6.3 Du passage d'un niveau au suivant dans la partition LAR . . . 102
6.4 Discussion des résultats 104
6.5 Sécurité 104
6.6 À retenir 105

7 Chiffrement sélectif du flot LAR iS+P **107**
7.1 Position du problème et étude préliminaire 108
7.2 Chiffrement de la partition 109
7.2.1 Principe 109
7.2.2 Modifications visuelles 111
7.2.3 Résultats 111
7.3 Chiffrement combiné : partition et *flat*-LAR 119
7.3.1 Principe 119
7.3.2 Modifications visuelles 119
7.3.3 Résultats 120
7.4 Chiffrement combiné : partition, flat et vecteur d'initialisation 124
7.4.1 Principe 124
7.4.2 Modifications visuelles 124
7.4.3 Résultats 130
7.5 À retenir 131

III Insertion de données 133

8 Dissimulation de données dans les images 135
8.1 Généralités . . . 136
8.2 Tour d'horizon . . . 137
8.2.1 Stéganographie . . . 137
8.2.2 Tatouage numérique . . . 137
8.2.3 Prise d'empreinte numérique . . . 138
8.2.4 Insertion de (meta)-données . . . 138
8.2.5 Pour la suite . . . 138
8.3 Propriétés de l'insertion de données . . . 139
8.3.1 Capacité d'insertion . . . 139
8.3.2 Complexité de la méthode . . . 139
8.3.3 Inversibilité . . . 139
8.3.4 Robustesse . . . 140
8.3.5 Sécurité . . . 140
8.3.6 Transparence . . . 140
8.3.7 Type de vérification . . . 140
8.4 Les attaques . . . 141
8.4.1 Attaque par filtrage . . . 141
8.4.2 Attaques géométriques . . . 141
8.4.3 Attaques additives . . . 142
8.4.4 Attaque par compression . . . 142
8.4.5 Attaque par impression et numérisation . . . 142
8.4.6 Attaque par ré-échantillonnage . . . 142
8.4.7 À retenir . . . 143
8.5 Techniques d'insertion de données . . . 143
8.5.1 Techniques du domaine spatial . . . 143
8.5.2 Les techniques dans le domaine transformé . . . 145
8.6 Conclusion . . . 152

9 Extension de la différence au LAR iS+P 153
9.1 Extension de la différence et LAR iS+P . . . 154
9.1.1 Similarités entre ED et LAR iS+P . . . 154
9.1.2 L'extension de la différence . . . 155
9.2 Intégration de ED au LAR iS+P . . . 161
9.2.1 Appariement des pixels . . . 161
9.2.2 Adaptation à la charge utile . . . 162
9.3 Validation de la méthode . . . 163
9.3.1 Évaluation de la capacité d'insertion . . . 165
9.3.2 Performances capacité-distorsion . . . 166

9.3.3 Évaluation de la qualité visuelle 167
9.3.4 Influence sur les performances de codage 175
9.4 À retenir 176

Bibliographie

Almeida, L. B. 1994, «The fractional fourier transform and time-frequency representations», *IEEE Trans. on Signal Processing*, vol. 42, n° 11, p. 3084–3091.

Babel, M. 2005, *Compression d'images avec et sans perte par la méthode LAR Locally Adaptive Resolution*, thè de doctorat, INSA Rennes.

Babel, M., O. Déforges & J. Ronsin. 2005a, «Interleaved s+p : décomposition pyramidale entrelacée à contexte de prédiction enrichi», *in Proc. of GRETSI*.

Babel, M., O. Déforges & J. Ronsin. 2005b, «Interleaved s+p pyramidal decomposition with refined prediction model», *in ICIP*, vol. 2, p. 750–753.

Babel, M., B. Parrein, O. Déforges, N. Normand, J. P. Guédon & J. Ronsin. 2005c, «Secure and progressive transmission of compressed images on the internet : application to telemedicine», *in SPIE*, p. 126–136.

Barba, D. & P. L. Callet. 2003, «A robust quality metric for color image quality assessment», *in Proc. of the International Conference on Image Processing*, p. 437–440.

Barni, M., F. Barolini, V. Cappellini & A. Piva. 1998, «A DCT-domain system for robust image watermarking», *Signal Processing*, vol. 66, p. 357–372.

Bas, P. 2000, *Méthodes de tatouages d'images fondées sur le contenu*, thè de doctorat, Institut National Polytechnique de Grenoble, France.

Bas, P., J.-M. Chassery & F. Davoine. 1998, «Tatouage d'images par modification du code fractal», *in CORESA*, Institut Albert Bonniot, Lab. TIMC-IMAG.

Bender, W., W. Butera, D. Gruhl, R. Hwang, F. J. Paiz & S. Pogreb. 2000, «Applications for data hiding», *IBM Syst. J.*, vol. 39, n° 3-4, p. 547–568, ISSN 0018-8670.

Bender, W., D. Gruhl, N. Morimoto & A. Lu. 1996, «Techniques for data hiding», *IBM Systems*, vol. 35.

Bhargava, B., C. Shi & Y. Wang. 2004, «Mpeg video encryption algorithm», *Multimedia Tools and Applications*, vol. 24, n° 1, p. 57–79.

Biham, E. & A. Shamir. 1993, *Differential Cryptanalysis of the Data Encryption Standard*, Springer-Verlag.

Borisov, N., I. Goldberg & D. Wagner. 2001, «Intercepting mobile communications : The insecurity of 802.11», *in Proc. of the 7th annual International Conference on Mobile Computing and Networking*.

Celik, M. U., G. Sharma, E. Saber & A. M. Tekalp. 2005, «Lossless generalized-lsb data embedding», *IEEE Trans. on Image Processing*, vol. 14, n° 2, p. 253–266.

Celik, M. U., G. Sharma, A. M. Tekalp & E. Saber. 2002, «Reversible data hiding», *in Proc. of the Int. Conf. on Image Processing*, p. 157–160.

Cheng, H. & X. Li. 1996, «On the application of image decomposition to image compression and encryption», *in Communications and Multimedia Security II IFIP TC6/TC11 Second Joint Working Conference on Communications and Multimedia Security*, P. Horster, Chapman and Hall, p. 116–127.

Cheng, H. & X. Li. 2000, «Partial encryption of compressed images and videos», *IEEE Transactions on Signal Processing*, vol. 48, n° 8, p. 2439–2451.

Cormen, T., C. Leiserson, R. Rivest & C. Stein. 2002, *Introduction à l'algorithmique*, 2e édition, Dunod.

Cox, I., J. Killian, F. Leighton & T. Shamoon. 1997, «Secure spread spectrum watermarking for multimedia», *IEEE Trans. on Image Processing*, vol. 6, n° 12, p. 1673–1687.

Cox, I. J., J. Kilian, T. Leighton & T. Shamoon. 1995, «Secure spread-spectrum watermarking for multimedia», rapport technique, NEC Research Institute.

Cox, I. J., M. Miller, J. Bloom, J. Fridrich & T. Kalker. 2007, *Digital watermarking and stenography*, Morgan Kaufmann.

Daemen, J. & V. Rijmen. 2002, *The Design of Rijndael : AES – The Advanced Encryption Standard*, Springer-Verlag.

Davoine & Pateux, . 2004, *Tatouage de documents audiovisuels numériques*, Traité IC2 série Traitement du Signal et de l'Image, Hermès.

Deng, R. H., D. Ma, W. Shao & Y. Wu. 2005, «Scalable trusted online dissemination of jpeg2000 images», *Multimedia Systems*, vol. 11, n° 1, p. 60–67.

Diffie, W. & M. E. Hellman. 1976, «New directions in cryptography», *IEEE Trans. on Information Theory*, vol. 22, p. 644–654.

Diffie, W. & M. E. Hellman. 1977, «Exhaustive cryptanalysis of the nbs data encryption standard», *Compuer*, , n° 10, p. 74–84.

Djurovic, I., S. Stankovic & I. Pitas. 2001, «Digital watermarking in the fractional fourier transformation domain», *Journal of Network and Computer Applications*, , n° 24, p. 167–173.

Déforges, O. 2004, «Codage d'images par la méthode LAR et méthodologie Adéquation Algorithme Architecture. De la définition des algorithmes de compression au prototypage rapide sur architectures parallèles hétérogènes», HDR, Université de Rennes 1.

Déforges, O. & J. Ronsin. 2002, «Supervised segmentation at low bit rates for region representation and color image compression», *in ICME*, vol. 1, p. 665–668.

Fang, J., J. Sun & H. Qian. 2006, «Compliant asymmetric authenticated encryption scheme for jpeg2000 code-streams», *International Journal of Computer Science and Network Security*, vol. 6, n° 11, p. 272–276.

Finkel, R. & J. L. Bentley. 1974, «Quad trees : A data structure for retrieval on composite keys», *Acta Informatica*, vol. 4, n° 1, p. 1–9.

FIPS PUB 46–3, . 1999, «Data encryption standard», rapport technique, National Institute of Standard and Technology.

Furon, T. 2002, *Application du tatouage numérique à la protection de copie*, thè de doctorat, ENST Paris.

Garcia, J. A., J. Fernandez-Valdivia, X. R. Fernandez-Vidal & R. Rodriguez-Sanchez. 2001, «Information theoretic measure for visual target distinctness», *IEEE. Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, n° 4, p. 362–383.

Gilani, S. A. M. & A. N. Skodras. 2000, «DLT-based digital image watermarking», .

Graham, R. E. 1958, «Predictive quantizing of television signals», *in IRE-WESCON Convention Record*, vol. 22, p. 147–157.

Griwotz, C. 1998, «Video protection by partial content corruption», *in Multimedia and Security Workshop at the 6th ACM International Multimedia Conference*, p. 37–39.

Griwotz, C., O. Merkel, J. Dittman & R. Steinmetz. 1998, «Protecting vod the easier way», *in Proc. of the 6th ACM Multimedia Conference*, p. 21–28.

Guédon, J.-P. «Histoire de la transformée mojette (la forme et la projection», .

Hayes, M. H. 1992, «The reconstruction of a multidimensional sequence from the phase or magnitude of the FFT», *in IEEE Transactions on Acoustics, Speech and Signal Processing*, p. 140–154.

Inoue, H., A. Miyazaki & T. Katsura. 1999, «An image watermarking method based on the wavelet transform», .

JEITA, . 2002, «Exchangeable image file format for digital still cameras : Exif version 2.2», rapport technique, Japan Electronics and Information Technology Industries Association.

Kailasanathan, C. 2002, «Compression performance of jpeg encryption scheme», *in Proc. Of the 14th IEEE International Conference on Digital Signal Processing*, vol. 2, p. 1329–1332.

Kang, S. & Y. Aoki. 1999, «Digital image watermarking by Fresnel transform and its robustness», .

Kankanhalli, M. S. & K. F. Hau. 2002, «Watermarking of electronic text documents», *Electronic Commerce Research*, vol. 2, n° 1, p. 169–187.

Kaukonen, K. & T. Thayer. 1999, *A Stream Cipher Encryption Algorithm "Arcfour"*, IETF Internet-draft.

Kerckhoffs, A. 1883, «La cryptographie militaire», *Journal des sciences militaires*, vol. IX, p. 5–38, 161–191.

Knuth, D. E. 1998, *Seminumerical Algorithms*, *The Art Of Computer Programming*, vol. 2, 3e édition, Addison-Wesley.

Kunkelmann, T. & U. Horn. 215–218, «Partial video encryption based on scalable coding», *in Proc. of the 5th International Workshop on Systems, Signals and Image Processing*, p. 1998.

Kunkelmann, T. & R. Reinema. 1997, «A scalable security architecture for multimedia communication standard», *in Proc. of the IEEE International Conference on Multimedia Computing and Systems*, p. 660–661.

Kutter, M. & F. Petitcolas. 1999, «A fair benchmark for image watermarking systems», *in Proc. of Electronic Imaging, Security and Watermarking of Multimedia Content*, vol. 3657, SPIE, p. 226–239.

Lai, X. & J. L. Massey. 1990, «A proposal for a new block encryption standard», *in Proc. of Eurocrypt '90*, p. 389–404.

Lookabaugh, T. D., D. C. Sicker, D. M. Keaton, W. Y. Guo & I. Vedula. 2003, «Security analysis of selectively encrypted mpeg-2 stream», *Multimedia Systems and Applications*, vol. 5241, n° 6, p. 10–21.

Matsui, K. & K. Tanaka. 1994, «Video-steganography», *in Proc. of IMA Intellectual Property Project*, vol. 1, p. 187–206.

Matsui, M. 1993, «Linear cryptanalysis method for des cipher», *in Proc. of Eucrypt '93*, *LNCS*, vol. 765, Springer-Verlag, p. 386–397.

Matsumoto, M. & T. Nishimura. 1998, «Mersenne twister : a 623-dimensionally equidistributed uniform pseudorandom number generator», *ACM Trans. on Modeling and Computer Simulation*, vol. 8, n° 1, p. 3–30.

Menezes, A., P. van Oorschot & S. Vanstone. 1996, *Handbook of Applied Cryptography*, CRC Press. Http ://www.cacr.math.uwaterloo.ca/hac.

Motsch, J., O. Dutheil, O. Déforges & M. Babel. 2007, «Codage multirésolution sans perte et insertion de données : une approche intégrée», *in Proc. of CORESA*.

Motsch, J., O. Déforges & M. Babel. 2006a, «Embedding multilevel image encryption in the lar codec», *in Proc. of the IEEE Communications International Conference*.

Motsch, J., O. Déforges & M. Babel. 2006b, «Protection de données à coût nul dans un codeur d'images multirésolution», *in Proc. of CORESA*.

Muresan, D. D. & T. W. Parks. 2001, «Optimal recovery approach to image interpolation», *in Proc. of the International Conference on Image Processing*, vol. 3, p. 848–851.

Nahrstedt, K. & L. Qiao. 1997, «Is mpeg encryption by using random list instead of zigzag scan order secure ?», *in Proc. of the Internationale Symposium on Consumer Electronics*, p. 226–229.

Nicolas, H. 1992, *Hiérarchie de modèles de mouvement et méthodes d'estimation associées : Application au codage de séquences*, thè de doctorat, université de Rennes 1.

Norcen, R., M. Podesser, A. Pommer, H.-P. Schmidt & A. Uhl. 2003, «Confidential storage and transmission of medical image data», *Computers in Biology and Medicine*, vol. 3, n° 33, p. 277–292.

Norcen, R. & A. Uhl. 2003, «Selective encryption of the jpeg2000 bitstream», *in Proc. of the sixth joint working conference on Communications and Multimedia Security, Lectures notes on Computer Science*, vol. 2828, A. Lioy & D. Mazzochi, Springer Verlag, p. 194–204.

Pennebaker, W. B. & J. L. Mitchell. 1992, *JPEG Still image Data Compression Standard*, 1re édition, Kluwer Academic Publishers.

Petitcolas, F. 2000, «Watermarking schemes evaluation», *IEEE Trans. on Signal Processing*, vol. 17, n° 5, p. 58–64.

Petitcolas, F., R. Anderson & M. Kuhn. 1998, «Attacks on copyright marking systems», *in Proc ; of Information Hiding Workshop, LNCS*, vol. 1525, p. 219–239.

Pi, M. H., C. H. Li & H. Li. 2006, «A novel fractal image watermarking», *IEEE Trans. on Multimedia*, vol. 8, n° 3, p. 488–499.

Piva, A., M. Barni, F. Bartolini & V. Cappellini. 1997, «DCT-based watermark recovering without resorting to the uncorrupted original image», .

Pommer, A. & A. Uhl. 2002, «Selective encryption of wavelet packet subband structures for secure transmission of visual data», *in Multimedia and Security Workshop*, J. Dittmann, J. Fridrich & P. Wohlmacher, ACM Multimedia, Juan-lès-Pins, p. 67–70.

Qiao, L. & K. Nahrsted. 1997, «A new algorithm for mpeg video encryption», *in Proc. of the International Conference on Imaging Science, System and Technology*, p. 21–29.

Qiao, L. & K. Nahrstedt. 1998, «Comparison of mpeg encryption algorithm», *International Journal on Computers and Graphics*, vol. 22, n° 3, p. 437–444.

Redl, M., M. K. Weber & M. W. Oliphant. 1995, *An Introduction to GSM*, Artech House.

Rivest, R., L. Shamir & L. Adleman. 1978, «A method for obtaining digital signatures and public-key cryptosystems», *Communications of the ACM*, vol. 21, n° 2, p. 120–126.

Ruanaidh, J. J. O., W. J. Dowling & F. M. Boland. 1997, «Phase watermarking for digital images», rapport technique, Dpt. of Electronic and Electrical Engineering, Univ. of Dublin, Ireland.

Ruanaidh, J. J. O. & T. Pun. 1997, «Rotation, scale and translation invariant digital image watermarking», *submitted to Signal Processing*.

Said, A. & W. Pearlman. 1993, «Reversible image compression via multiresolution representation and predictive coding», *in Visual Communication and Image Processing*, vol. 209, SPIE, p. 664–674.

Said, A. & W. A. Pearlman. 1996, «A new fast and efficient image codec based on set partitioning in hierarchical trees», *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, p. 242–250.

Schneier, B. 1996, *Applied Cryptography*, 2e édition, John Wiley and Sons.

van Schyndel, R. 2001, *Digital Watermarking and Signal Delay Estimations using Pseudonoise Sequences*, thè de doctorat, Monash University.

van Schyndel, R. G., A. Z. Tirkel & C. F. Osborne. 1994, «A digital watermarking», *in Proc. of the International Conference on Image Processing*, vol. 2, p. 86–90.

Shannon, C. 1949, «Communication theory of secrecy systems», *Bell System Technical Journal*, vol. 28, n° 4, p. 656–715.

Shi, C. & B. Bhargava. 1998a, «A fast mpeg video encryption algorithm», *in Proc. of the 6th ACM International Multimedia Conference*, p. 81–88.

Shi, C. & B. Bhargava. 1998b, «Light-weight mpeg video encryption algorithm», *in Proc. of the International Conference on Multimedia*, p. 55–61.

Shin, S. U., K. S. Sim & K. H. Rhee. 1999, «A secrecy scheme for mpeg video data using the joint of compression and encryption», *in Proc. of the 1999 Information Security Workshop*, *Lecture Notes on Computer Science*, vol. 1729, Springer Verlag, Kuala Lumpur, p. 191–201.

Shor, P. W. 1994, «Polynomial-time algorithms fo prime factorization and discrete logarithms on a quantum computer», *in Proc. of the Annual Symposium on Foundations of Computer Science*, p. 124–134.

Sinkov, A. 1966, *Elementary Cryptanalysis : A Mathematical Approach*, The Mathematical Association of America.

Stanković, S., I. Djurović & I. Pitas. 2001, «Watermarking in the space/spatial-frequency domain using two-dimensional Radon-Wigner distribution», *in IEEE Transactions on Image Processing*, vol. 10, p. 650–658.

Talbot, J. & D. Welsh. 2006, *Complexity and Cryptography, an Introduction*, Cambridge University Press.

Tang, L. 1996, «Methods for encrypting and decrypting mpeg video data efficiently», *ACM Multimedia*, p. 219–229.

Taubman, D. & M. Marcellin. 2001, *JPEG2000 : Image Compression Fundamentals, Standards and Practice*, The International Series in Engineering and Computer Science, Springer Verlag.

Thomas, S. A. 2000, *SSL and TLS essentials securint the Web*, Wiley.

Tian, J. & R. O. Wells. 2004, «Reversible data-embedding with a hierarchical structure», *in ICIP*, vol. 5, School of Engineering and Science, Bremen, Germany, p. 3419–3422.

Tirkel, A. Z., R. G. Schyndel & C. F. Osborne. 1995, «A two dimensional digital watermark», *in Proc. of DICTA conference*, p. 378–383.

Tosun, A. S. & W. C. Feng. 2001a, «Lightweight security mechanisms for wireless video transmission», *in Proc. of the IEEE International Conference on Information Technology : Coding and Computing*, p. 157–161.

Tosun, A. S. & W. C. Feng. 2001b, «On error preserving encryption algorithms for wireless video transmission», *in Proc. of the 9th ACM Multimedia Conference*, p. 302–307.

Ueraha, T. & R. Safavi-Naini. 2000, «Chosen dct coefficients attack on mpeg encryption schemes», *in Proc. of the IEEE Pacific Rim Conference on Multimedia*, Sydney, p. 316–319.

Unnikrishnan, G. & K. Singh. 2000, «Double random fractional fourier-domain encoding for optial security», *Optical Engineering*, vol. 39, n° 11, p. 2853–2859.

Van Droogenbroeck, M. & R. Benedett. 2002, «Techniques for a selective encryption of uncompressed and compressed images», *in ACIVS Advanced Concepts for Intelligent Vision Systems*, Ghent, Belgium, p. 90–97. URL `http ://www.ulg.ac.be/telecom/publi/publications/mvd/acivs2002mvd/index.html`.

Wen, J., M. Severa, W. Zeng, M. Luttrell & W. Jin. 2002, «A format compliant configurable encryption framework for access control of video», *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, n° 6, p. 545–557.

Wendl, M. C. 2003, «Collision probability between sets of random variables», *Statistics and Probability Letters*, vol. 64, n° 3, p. 249–254.

Wiener, M. J. 1996, «Efficient des key search», *in Practical Cryptography for Data Internetworks*, W. Stallings, IEEE Computer Society Press, p. 31–79.

Wu, C.-P. & C.-C. J. Kuo. 2000, «Fast encrytion methods for audiovisual data confidentiality», *in Proc. of SPIE Photonics East–Symposium on Voice, Video, and Data Communications*, vol. 4209, p. 284–295.

Wu, C.-P. & C.-C. J. Kuo. 2001, «Efficient multimedia encryption via entropy codec design», *in Proc. of SPIE, Security and Watermarking of Multimedia Contents III*, vol. 4314, p. 128–138.

Wu, X. 1996, «Lossless compression of continuous-tone images, via context selection, quantization and modelling», *IEEE Trans. on Image Processing*, vol. 6, n° 5, p. 656–664.

Wu, X. & N. M. and. 1995, «A Context-based, Adaptive, Lossless/Nearly-Lossless Coding Scheme for Continuous-Tone Images (CALIC)», *International Standards Organization working document, ISO/IEC SC29/WG 1/N256.*

Wu, Y., D. Ma & R. H. Deng. 2004, «Progressive protection of jpeg2000 codestrams», *in Proc. of the International Conference on Image Processing*, vol. 5, p. 3447–3450.

Xia, X.-G., C. G. Boncelet & G. R. Arce. 1997, «A multiresolution watermark for digital images», .

XMP specification, . 2005, «Xmp specification», rapport technique, Adobe Corp.

Zeng, W. & S. Lei. 1999, «Efficient frequency domain video scrambling for content access control», *in Proc. of the 7th ACM International Multimedia Conference*, p. 285–293.

Zeng, W. & S. Lei. 2003, «Efficient frequency domain selective scrambling of digital video», *IEEE Trans. on Multimedia*, vol. 5, n° 1, p. 118–129.

Zhang, Z., G. Qiu, Q. Sun, X. Lin & Y.-Q. Shi. 2004, «A unified authentication framework of jpeg-2000», *in Proc. Of the IEEE International Conference on Multimedia and Expositions (ICME)*.

Zhao, J. & E. Koch. 1995, «Embedding robust labels into images for copyright protection», *in Proc. of the Int. Congress on Intellectual Property Rights for Specialized Information, Knowledge and new Technologies*, Fraunhofer Institute for Computer Graphics, Vienne.

Zhu, W., Z. Xiong & Y. Zhang. 1999, «Multiresolution watermarking for image and video», *IEEE Trans. on Circuits and Systems Video Technology*, vol. 9, n° 4, p. 545–550.

Zimmermann, P. 1995, *PGP Source Code and Internals*, MIT Press.

page laissée intentionnellement vide

RÉSUMÉ

La multiplication des dispositifs de prise de vues, classiques comme les appareils photographiques, nouveaux comme les imageurs IRM et les analyseurs multispectraux, amène à prendre en charge des volumes de données croissants exponentiellement. En corollaire, la nécessité d'archiver de telles quantités de données introduit des questions nouvelles. Comment gérer des images de très grande taille ? Comment garantir l'intégrité des données enregistrées ? Quel mécanisme utiliser pour lier une image à des données ?

Pendant longtemps, l'accent a été mis sur les performances de compression des schémas de codage, avec pour instruments de mesure les courbes débit-distorsion et quelques évaluations subjectives. Des propriétés de scalabilité, de gestion du débit, de résistance aux erreurs sont apparus pour prendre en compte les évolutions des techniques relatives aux media de communication. Cependant, la prise en compte des impératifs liés à l'archivage amène à considérer de nouveaux services, comme la gestion de la copie, l'ajout de métadonnées et la sécurité des images. Dans la plupart des cas, ces services sont fournis en dehors des schémas de compression, à l'exception notable de JPSEC, effort lié à JPEG-2000.

L'approche retenue dans cette étude est l'intégration dans un codeur d'images fixes performant, le codeur LAR iS+P, des primitives de chiffrement et d'insertion de données permettant d'assurer l'essentiel des services liés à l'archivage des images. Le codeur LAR iS+P est hiérarchique, scalable en résolution et qualité, et permet d'aller du codage avec perte vers du codage sans perte. Ses performances sont au-delà de l'état de l'art, particulièrement en très bas débit et en compression sans perte. Le point clef des performances de ce codeur est un partitionnement quadtree adapté au contenu sémantique de l'image.

Dans le contexte de l'archivage sécurisé d'images, ce document présente donc le triptyque suivant : compression efficace, chiffrement partiel et insertion de données. Pour la compression, le codeur scalable LAR iS+P montre des performances au-dela de l'état de l'art. Pour le chiffrement, l'utilisation de la structure hiérarchique du codeur LAR iS+P permet de mettre en place un schéma de chiffrement partiel. Il autorise une gestion fine des droits et des clefs, ainsi que le choix dans les niveaux de résolution et qualité possibles. Des éléments permettant une protection à coût nul sont également présentés. Pour l'insertion de données, le parallèle existant entre le LAR iS+P et l'extension de la différence permet de mettre en œuvre un schéma efficace.