ACADÉMIE D'AIX-MARSEILLE

UNIVERSITÉ D'AVIGNON ET DES PAYS DE VAUCLUSE

Tatouage d'image : Gain en robustesse et intégrité des images

Thèse

présentée et soutenue publiquement le 14 février 2003

pour l'obtention du

Doctorat de l'Université d'Avignon et des Pays de Vaucluse

par

Christian REY

Composition du jury :

Président :	Prof. Ramesh Pyndiah (ENST Bretagne)
Rapporteurs :	Prof. Christine Fernandez-Maloigne (Univ. Poitiers)
	Prof. Touradj Ebrahimi (EPFL)
Examinateurs :	Prof. Renato De Mori (Univ. Avignon - LIA)
	Dr. Fabien Petitcolas (Microsoft Research – UK)
	Prof. Jean-Luc Dugelay (Institut Eurécom)

Remerciements

Je tiens en premier lieu à adresser mes plus vifs remerciements à Monsieur Ramesh Pyndiah pour l'honneur qu'il me fait de présider ce jury de thèse, à Madame Christine Fernandez-Maloigne et à Monsieur Touradj Ebrahimi pour l'intérêt qu'ils ont porté à mon travail en acceptant d'être rapporteurs, et à Messieurs Renato De Mori et Fabien Petitcolas pour l'honneur qu'ils me font de participer à ce jury en tant qu'examinateurs.

Je remercie Monsieur Jean-Luc Dugelay pour m'avoir accueilli au sein du groupe Image du département Multimédia de l'Institut Eurécom, pour l'intérêt constant qu'il a porté à mon travail, ainsi que pour la confiance et le soutient dont il m'a gratifié tout au long de ma thèse.

Je remercie également Stéphane Roche qui a guidé mes premiers pas dans le domaine passionnant du tatouage d'image. Ses conseils m'ont permis de gagner beaucoup de temps et les nombreuses discussions que nous avons eues m'ont permis de franchir des étapes importantes de mon travail. Ses travaux de recherche sont à la base du travail présenté dans ce mémoire.

Je remercie également Karine Amis pour ses conseils et sa disponibilité, ainsi que toute l'équipe du département Signal et Communication de L'ENST Bretagne pour leur contribution au projet GET « Turbo Watermark ».

Je tiens également à exprimer mes remerciements les plus sincères :

à mes parents pour avoir eu la patience de relire mon mémoire et faire la chasse aux fautes d'orthographe. Si des fautes subsistent elles sont certainement dues à des ajouts de dernière minute,

à Pascal Gros pour sa disponibilité et son aide technique,

à mes compagnons de bureau : Gwen, Florent et Ejaz dit 'Pathan' (pour ces leçons de squash),

à Sergio et Stéphane pour les nombreuses journées de ski et les alertes à la poudreuse,

ainsi qu'à Ana, Caroline, Marie, Souad, Katia, Itheri, David, Fred, Alain, Benoit, Philippe, Emmanuel, Navid, Tak, Maxime, Fabrice , Younes, Nicolas,

et merci enfin à tous ceux que je n'aurais pas dû oublier…

Table des matières

Liste des abréviations

CA Contrôle d'Accès.

BBH (Block-Based Hash function) Fonction de hachage.

BCH (Bose-Chaudhuri-Hocquenghen) Codes correcteurs.

BD Base de Données.

BER (Bit Error Rate) Taux d'erreur binaire.

BM Abréviation de Block Matching.

CRC (Cyclic Redundancy Check codes) Code permettant la détection d'erreurs.

DCT (Discrete Consinus Transform) Transformée en Cosinus Discrète.

DVD (Digital Video Disc ou Digital Versatile Disc) Nouvelle génération de disque optique, offrant des capacités de stockage supérieures au CD-ROM.

DWT (Discrete Wavelet Transform) Transformée en ondelettes discrète.

EZW Embedded Zero-tree Wavelet

FFT (Fast Fourier Transform) Transformée de Fourier rapide.

GIF (Graphics Interchange Format) Format de fichier graphique, permettant de gérer des images fixes ou animées avec une palette de 256 couleurs au plus.

HSL (Hue, Saturation, Luminescence) Codage d'une image couleur suivant les plan de Teinte, de Saturation et de Luminosité.

HTML (HyperText Markup Language) Langage de description de documents multimédia diffusés sur le WEB.

HVS (Human Visual System) Le modèle visuel humain.

ID Identifiant unique.

IFS (Iterated Function Systems) Systèmes de fonctions itérées utilisés en codage fractal.

JBIG (Joint Bi-level Image Group) Format de compression d'image binaire, utilisé entre autres par les FAX.

JND (Just Noticeable Difference) Unité de mesure de qualité d'image associée à la métrique de Watson.

JPEG (Joint Photographic Experts Group) Format d'image permettant la compression avec pertes.

Lab Espace colorimétrique perceptif homogène.

LPM (Log Polar Maping) Changement de repère : cartésien vers logarithme-polaire.

LSB (Less Signifiant Bits) Bits les moins significatifs.

MER (Message Error Rate) Taux d'erreur par message.

MD5 (Message Digest version 5) - Fonction de hachage générant une empreinte sur 128 bits.

MPEG (Moving Pictures Experts Group) Norme de compression vidéo.

MSB (Most Signifiant Bits) Bits les plus significatifs.

MPSNR (Masked Peak Signal Noise Ratio) Métrique dérivée du PSNR.

NMSE (Normalised Mean Square Error) Erreur Quadratique Moyenne Normalisée.

NVF (Noise Visibility Function)

PPM (Portable PixMap) Format de fichier graphique non compressé.

PSNR (Peak Signal Noise Ratio) Mesure objective de la qualité visuelle basée sur l'erreur quadratique moyenne normalisée.

RGB (ou RVB) Abréviations des 3 couleurs primaires additives (rouge, vert, bleu) utilisées pour coder les images couleur.

RNRT Réseau National de Recherche en Télécommunications.

ROC (Receiver Operating Characteristic).

RSA Méthode de chiffrement asymétrique proposée par Ronald Rivest, Adi Shamir et Leonard Adleman.

SHA Secure Hash Algorithm - Fonction de hachage générant une empreinte sur 160 bits. Plus robuste que MD5, mais également plus lent.

wPSNR (Weighted Peak Signal Noise Ratio) Métrique dérivée du PSNR.

XOR Opération logique OU exclusif.

YUV (ou YCrCb) Codage d'image couleur suivant un plan de luminance (Y) et deux plans de chrominance et Sauration (U-V).

Table des figures

Liste des tableaux

Introduction générale

La révolution numérique, l'explosion des réseaux de communication et l'engouement sans cesse grandissant du grand public pour les nouvelles technologies de l'information entraînent une circulation accrue des documents multimédia (images, vidéos, textes, son, etc.). L'ampleur de ce phénomène est telle que des questions fondamentales se posent désormais quant à la protection et au contrôle des données échangées. En effet, de par leur nature numérique les documents multimédia peuvent être dupliqués, modifiés, transformés et diffusés très facilement. Dans ce contexte, nous sommes en droit de nous interroger sur le respect des droits d'auteur, le contrôle des copies et l'intégrité d'un document. Face à toutes ces interrogations, le tatouage numérique (ou « watermarking » est très naturellement apparu comme une solution alternative ou complémentaire pour renforcer la sécurité des documents numériques.

L'idée de base du tatouage numérique consiste à cacher dans un document multimédia une information subliminale (*i.e.* invisible ou inaudible suivant la nature du document) permettant d'assurer un service de sécurité (*e.g.* droits d'auteur, intégrité, trançabilité, non répudiation, etc.). Une des particularités du tatouage par rapport à d'autres techniques, comme par exemple un stockage simple de l'information dans l'en-tête du fichier, est que la marque est liée de manière intime et résistante aux données. De ce fait, le tatouage est théoriquement indépendant du format de fichier et il peut être détecté ou extrait même si le document a subi des modifications ou s'il est incomplet.

Dans cette thèse, nous nous focaliserons principalement sur le tatouage des images numériques. Dans le chapitre 1, nous dressons un panorama des problématiques associées aux différents champs d'application du tatouage d'image. Nous y définissons la notion de tatouage au travers de ses différentes déclinaisons : fragile ou robuste, visible ou invisible, inversible ou non-inversible, etc. Nous présentons également dans ce chapitre, les techniques de base de la communauté afin de permettre au lecteur d'appréhender la problématique, les difficultés et les limites inhérentes au tatouage d'image.

Dans le chapitre 2, nous décrivons notre technique de tatouage basée sur un modèle affine d'IFS (cette technique a déjà fait l'objet d'une thèse précédente [Roc99] au sein de l'Institut

Eurécom). Un schéma complet comprenant les phases d'insertion et d'extraction du tatouage y sera décrit. L'information que l'on souhaite cacher dans l'image peut prendre la forme d'un logo binaire, d'une chaîne de caractères, d'un identifiant, etc. Cette information doit subir une opération de mise en forme préalable à son introduction dans l'image. Cette opération vise d'une part à adapter les données à transmettre au canal constitué par l'image, et d'autre part à assurer un cryptage de l'information pour empêcher son extraction par une personne non autorisée. Parmi les opérations de mise en forme du message, nous détaillerons celles qui consistent à ajouter de la redondance pour faire face aux différentes manipulations photométriques de l'image. Comme nous le verrons dans les chapitres suivants, la mise en forme des informations à cacher constitue le véritable fil conducteur de cette thèse.

Dans le chapitre 3, nous proposons une amélioration de l'algorithme de tatouage décrit au chapitre précédant. Nous discuterons principalement de l'utilisation de codes correcteurs d'erreurs, et plus particulièrement des codes produits, afin d'accroître la robustesse générale du tatouage face à des attaques de type signal. La robustesse est en effet un des points clés de tout algorithme de tatouage d'image. Pour la majorité des applications, le tatouage doit pouvoir être extrait sans erreur, même si l'image subit des manipulations involontaires ou délibérées. Nous présenterons notamment des résultats significatifs face à la compression JPEG et l'ajout d'un bruit gaussien.

Toujours dans un souci d'améliorer les performances de notre algorithme, nous décrivons au chapitre 4 deux techniques de resynchronisation permettant de pallier les effets liés aux déformations géométriques de l'image. La première méthode est dédiée aux déformations locales de faibles amplitudes comme les déformations aléatoires engendrées par *Stirmark*, un logiciel d'attaque bien connu de la communauté « watermarking ». Tandis que la seconde méthode permet de détecter et de compenser les déformations résultant d'une application linéaire (*e.g.* rotation, changement d'échelle, cisaillement, etc.). Ces deux techniques opèrent directement dans le domaine spatial et ne nécessite, à l'extraction aucune information *a priori* sur l'image tatouée originale et le message caché.

Enfin, au chapitre 5, nous aborderons une application très particulière du tatouage d'image qui est celle de la protection de l'intégrité des images. Nous y présenterons tout d'abord quelques unes des méthodes les plus significatives permettant d'assurer un tel service, ainsi que les contraintes à satisfaire. Ce type de service remet partiellement en cause certains paramètres communément établis en tatouage d'image pour assurer des fonctions plus classiques de sécurité, notamment en termes de quantité et nature des informations cachées. Nous proposerons ensuite deux techniques permettant de protéger le contenu des images. La première méthode a recours à un tatouage robuste. L'idée est d'enfouir dans l'image à protéger des informations sur son propre contenu. Lors de la vérification, ces informations sont extraites et comparées à celles de l'image testée. Les différences constatées indiquent alors les régions de l'image qui ont été manipulées. Le principal avantage de cette méthode réside dans le fait que l'image s'auto-suffit, c'est-à-dire qu'aucune information autre que l'image elle-même n'est nécessaire. La deuxième approche est présentée comme une al-

ternative à la première. Elle se différencie principalement de cette dernière dans la mesure où les caractéristiques de l'image originale ne sont plus cachées dans l'image elle-même, mais enregistrées sous la forme d'une signature ou d'une empreinte auprès d'une tierce personne. Seul un identifiant permettant de retrouver ces informations est réellement caché dans l'image. Cette solution est certes moins souple d'utilisation que la précédente, puisqu'il est nécessaire de recourir à une tierce personne, mais les possibilités offertes et la fiabilité du système sont bien meilleures.

Nous conclurons enfin cette thèse en évoquant les perspectives offertes par les systèmes de tatouage d'image ainsi que les nouvelles applications potentielles.

Chapitre 1

Problématique générale du tatouage d'image

1. Introduction

1.1. Cryptographie et Stéganographie

La cryptologie existe depuis des siècles. Depuis l'invention de l'écriture, le besoin de sécurité est motivé par les problèmes de confidentialité et d'intégrité : on souhaite éventuellement que l'information écrite ne soit accessible qu'à certaines personnes et qu'elle ne soit pas modifiée volontairement dans un but de mystification. La cryptologie regroupe à la fois la cryptographie, qui désigne l'art de chiffrer le contenu d'un message susceptible d'être intercepté lors de sa transmission, et la cryptanalyse, qui consiste à casser le code protégeant un message chiffré. Depuis son origine, où elle était principalement réservée à un usage militaire et diplomatique, la cryptologie a considérablement évolué, notamment avec l'apparition de l'ordinateur, et s'étend aujourd'hui au domaine civil pour la protection des données circulant sur les réseaux informatiques. Ainsi, la cryptologie moderne est maintenant une discipline de recherche publique de l'informatique théorique utilisant des outils mathématiques sophistiqués.

Le « watermarking » (littéralement filigrane) ou tatouage d'image peut être perçu comme une branche de la stéganographie. Le mot stéganographie vient du grec « steganos » (caché ou secret) et « graphy » (écriture ou dessin), et signifie littéralement « écriture cachée ». La stéganographie consiste à cacher, de manière subliminale, un message secondaire dans un message primaire. Le message primaire reste lisible de tous, tandis que le message secondaire n'est lisible que par une ou

plusieurs personnes propriétaires d'une information secrète. Pour la petite histoire, les premières traces de stéganographie remontent à l'Antiquité. Un légataire romain voulant envoyer un message à César, le camoufla dans une amphore qu'il lui envoya en guise de cadeau. Une autre forme de stéganographie, elle aussi très rudimentaire, consistait à raser le crâne d'un esclave. On y tatouait alors le message, et l'esclave était envoyé lorsque ses cheveux avaient repoussés. Le destinataire n'avait plus qu'à le faire raser de nouveau pour faire apparaître le message. Une autre forme de stéganographie très connue est le principe de l'encre invisible. Cette technique était très utilisée au moyen âge pour envoyer des messages secrets. A l'époque, l'encre était fabriquée simplement à base de jus d'oignons et de chlorure d'ammoniac. L'écriture était alors rendue visible en approchant le papier d'une flamme de bougie.

La stéganographie se distingue de la cryptographie dans la mesure où l'objectif principal de la cryptographie est de rendre illisible le message à toute personne ne possédant pas l'information secrète adéquate. De plus, alors que la cryptographie offre une sécurité plutôt *a priori* (par exemple, contrôle d'accès), la stéganographie offre une sécurité plutôt *a posteriori*, dans la mesure où le message secondaire est supposé rester accessible après recopies et manipulations du message primaire.

Contrairement à un stockage simple d'informations dans l'en-tête du fichier associé à une image, le tatouage est intimement lié aux données. De ce fait, il est donc théoriquement indépendant du format de l'image. Le tatouage permet une vérification ou une extraction efficace et automatique de certaines informations liées à l'origine, au contenu ou même à la diffusion d'une image.

1.2. Historique

Les premiers articles sur le sujet sont apparus au début des années 90. Très vite, de nombreux laboratoires et industriels se sont intéressés à ce domaine. Depuis 1995, le nombre de publications et de brevets a fait du tatouage un domaine majeur en traitement d'image (voir tableau 1).

Année	1992	1993	1994	1995	1996	1997	1998	1999	2000	2001
Publications	2	2	4	13	29	64	135	232	334	376

Tableau 1.1 – *Nombre de publications de 1992 à 1999 (source INSPEC, septembre 2002)*

Plusieurs domaines des télécommunications sont concernés par le tatouage d'image. Les principaux sont les suivants :

Le codage de source vidéo

Plusieurs problèmes rencontrés en tatouage, comme ceux liés à la notion de visibilité du marquage, ou encore relatifs à la robustesse face à une compression avec pertes, sont bien connus par la communauté compression d'images. Des schémas de tatouage d'image incluant des aspects psychovisuels définis en codage de source ont d'ailleurs été proposés; en particulier, certains auteurs [DVM98a] proposent de s'appuyer sur une décomposition sous-bandes de l'image en canaux perceptifs visuels afin de bien gérer le compromis visibilité *vs* robustesse.

La cryptographie

La sécurité des algorithmes de tatouage ne peut pas se baser uniquement sur un secret algorithmique ; il est donc nécessaire, à un moment ou un autre, de recourir à une notion de clé secrète [Ker83]. Les schémas de tatouage peuvent être également soumis à des attaques non spécifiques au tatouage comme l'attaque par collusion par exemple (*c.f.* attaques cryptographiques page 27).

Codage de canal et théorie de l'information;

Plusieurs schémas sont utilisés pour modéliser la problématique du tatouage [BBRP99], [RD97], [SC96], comme :

- Signature - Bruit - Bruit : une signature peut être vue comme un signal noyé dans du bruit. On distingue alors deux types de bruit : l'image originale d'une part et les attaques d'autre part.

- Signature - Porteuse - Bruit : l'image originale constitue une onde porteuse vis-à-vis du signal à transmettre qui est la signature. Les éventuelles attaques sont comme précédemment modélisées par un bruit.

De plus, des techniques classiques sont fréquemment utilisées :

- Des codes correcteurs (*e.g.* BCH, Reed-Muller, Turbo Codes, etc.) sont employés [CC98], [DDNM98] afin de pallier à une faiblesse du détecteur de marques, et réparer quelques bits erronés de la marque obtenue après extraction ;

- L'étalement de spectre [CKLS97], [TRS+93] est souvent utilisé pour insérer au mieux (faible visibilité et haute robustesse) le tatouage dans l'image.

La demande des utilisateurs est telle que même si la technologie est encore très imparfaite et

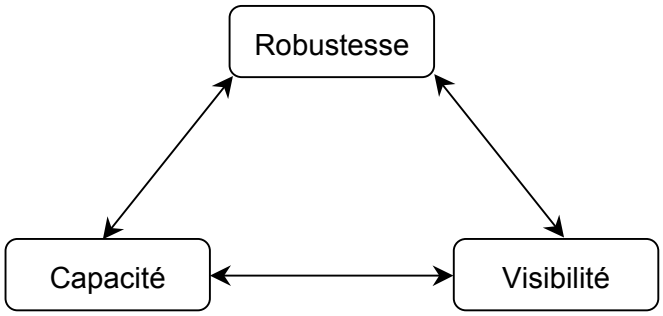


Figure 1.1 – *Compromis à réaliser en tatouage d'image*

immature, des logiciels commerciaux sont d'ores et déjà disponibles. Le mot clé « watermarking » a en outre fait son apparition dans les instances internationales relatives à JPEG-2000 [Jpeg00], MPEG-4 [Mpeg97], ou encore DVD [Dvd97].

Les principaux produits et logiciels actuellement disponibles sur le marché sont : DigiMarc [Digi], SureSign [Sure], EikonaMark [Eiko], SignIt! [Siga], Giovanni [Giov] Signafy [Sigb], SysCop [Sys], etc. Des évaluations préliminaires, semblent montrer que très peu de ces produits résistent actuellement à la dernière version de Stirmark (voir paragraphe 6.1.2) ; même si les tatoueurs SignIt!, Digimarc et SureSign réussissent un peu mieux le test que les autres. Ces résultats sont à considérer avec prudence, car comme nous le verrons plus loin, il est très difficile d'évaluer un tatoueur d'images; et a fortiori de le comparer avec d'autres. Notons enfin que le leader sur le marché est actuellement Digimarc, qui est déjà intégré dans plusieurs logiciels de retouche d'images bien connus comme Adobe Photoshop, Paint Shop Pro ou bien encore Corel Draw.

2. Principe général d'un système de tatouage d'image

2.1. Définition

Le tatouage d'image, que l'on peut sommairement décrire à l'aide de la figure 1.2, consiste à introduire, généralement de manière invisible, une information dans une image, puis à tenter de la récupérer après que l'image ait éventuellement subi des manipulations de natures variées.

2.2. Points clés d'un algorithme de tatouage

Les principales contraintes techniques à prendre en compte pour concevoir un algorithme de tatouage performant sont les suivantes :

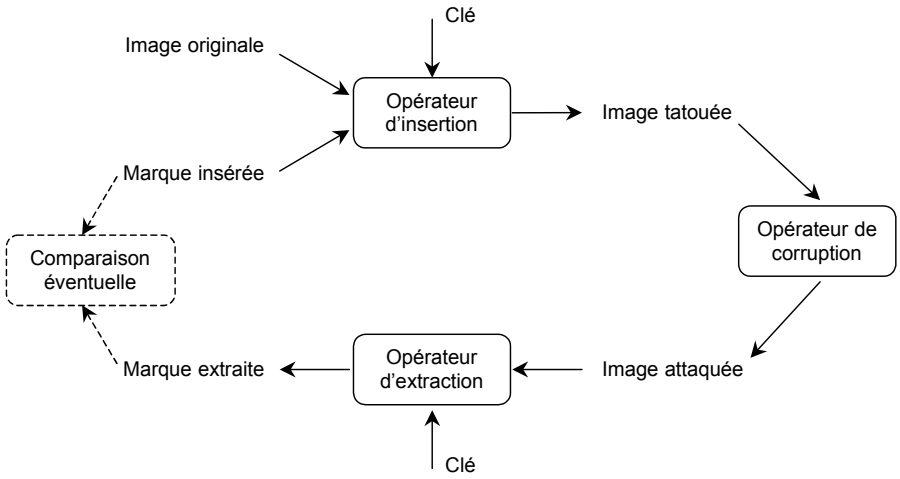


Figure 1.2 – *Dispositif générique d'un système de tatouage d'image*

La capacité (payload) ou ratio : c'est la quantité d'informations que l'on espère cacher par rapport à la quantité d'informations associée au support audio, image ou vidéo utilisé. Ordinairement de 16 à 64 bits sont suffisants pour assurer un service de droit d'auteurs à l'aide d'un identifiant, mais pas pour cacher des informations explicites comme un logo de société, assurer des services d'intégrité ou de non répudiation. On peut distinguer les signatures qui représentent seulement un pointeur vers une information stockée à l'extérieur de l'image, de celles qui s'autosuffisent pour fournir une information significative (une chaîne de caractères, un logo visuel, etc.). Notons qu'en termes de ratio seul, il sera évidemment plus aisé de cacher un message donné dans une vidéo qu'une image fixe.

L'invisibilité : le but est de faire en sorte que l'impact visuel du marquage (*i.e.* distorsion) soit la plus faible possible afin que le document marqué reste fidèle à l'original (*i.e.* modifications imperceptibles). De nombreux algorithmes prennent d'ailleurs en compte un modèle psychovisuel (HVS) [BBCP98], [SZT98].

La robustesse : il s'agit ici de pouvoir récupérer la marque même si l'image marquée a été manipulée. Il est nécessaire de distinguer plusieurs types d'attaques selon qu'elles sont considérées comme étant biens ou malveillantes, destructives ou non (en termes de dégradations visibles inacceptables et/ou d'utilisation commerciale rendue impossible). Les attaques bienveillantes regroupent les manipulations effectuées par un utilisateur de bonne foi. On trouve dans cette catégorie la compression JPEG, les conversions de format en général, les changements de résolution (zoom), etc. Il n'est pas possible d'énumérer l'ensemble des attaques bien ou malveillantes ne dégradant pas l'image de façon significative mais qui néanmoins sont capables de « lessiver » l'image marquée afin de retirer la marque, ou plus simplement d'empêcher de l'extraire correctement. Il existe des logiciels libres spécialisés dans le lessivage : Unzign [Unzi] et

Stirmark [Stir], officiellement présentés comme logiciel d'assistance pour la mise au point et l'évaluation d'algorithmes de tatouage. Actuellement, selon les auteurs de ces programmes, la quasi totalité (pour ne pas dire tous) des produits commerciaux ou R&D connus sont piégés sans trop de difficulté.

Il est facile de remarquer que ces trois critères sont contradictoires (voir Figure 1). Si l'on augmente par exemple la force de marquage dans le but de rendre le tatouage plus robuste, cela aura en contrepartie pour effet de rendre ce dernier également plus visible. De la même manière, si l'on augmente la taille du message à cacher cela se fera au détriment de la robustesse. Il est donc nécessaire de trouver le meilleur compromis possible entre ces trois paramètres en fonction de l'application visée. Il serait, par exemple, absurde de mettre au point un algorithme de tatouage de grande capacité si seulement quelques messages différents sont cachés en pratique. Ce qui est typiquement le cas pour le contrôle de copies (voir paragraphe 3.3) où deux bits suffisent pour coder les trois messages « copy-always », « copy-once » et « copy-never ». De même, certaines applications ne requièrent pas l'usage d'un tatouage robuste ; la fragilité du tatouage peut être alors exploitée afin d'assurer par exemple l'intégrité des images (voir chapitre 5). De ce fait, si la marque est altérée ou lessivée, l'image n'est plus considérée comme intègre.

2.3. Modes d'extraction du tatouage

Il existe plusieurs modes pour l'extraction du tatouage : le mode non-aveugle, le mode semi-aveugle et le mode aveugle (figures 1.3, 1.4, 1.5). Ces modes spécifient l'information *a priori* dont dispose le module d'extraction pour la vérification du tatouage. L'utilisation de tel ou tel mode dépendra de l'application visée et des protocoles utilisés.

Mode non-aveugle : (ou tatouage privé) le récepteur dispose de l'image ainsi que du tatouage original. Ce contexte est bien évidement incompatible avec des applications visant à vérifier l'intégrité de l'image, ou à assurer la vérification en temps réel du copyright (problème de temps d'accès à la base de données contenant les informations originales).

Mode semi-aveugle : (ou tatouage semi-privé) le tatouage original est supposé connu lors de l'extraction et utilisé le plus souvent *via* un score de corrélation.

Mode aveugle : (ou tatouage public) il s'agit du seul mode où l'on peut réellement parler d'extraction du tatouage (par opposition à la vérification intervenant dans les deux précédents modes) puisque l'on ne présuppose ni la connaissance du tatouage, ni la connaissance de l'image originale. C'est le mode d'extraction le plus intéressant, mais également le plus difficile à mettre en œuvre.

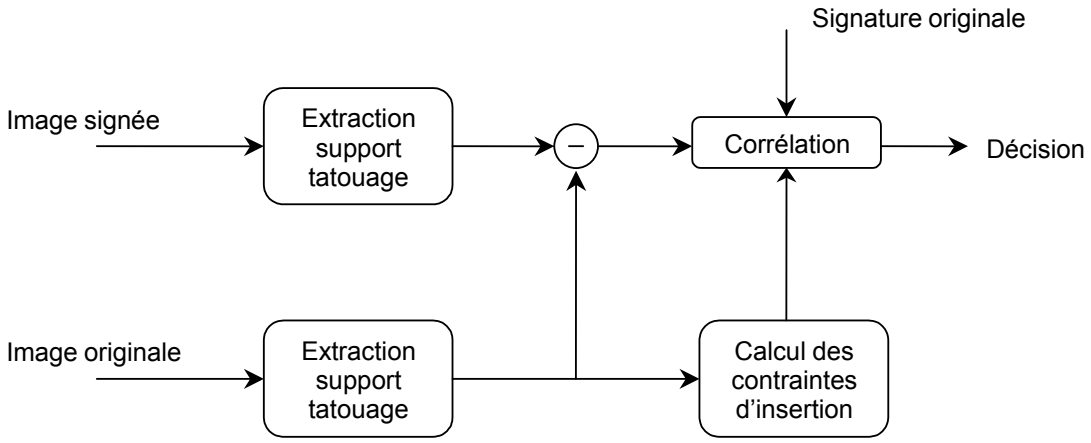


Figure 1.3 – *Mode d'extraction non-aveugle*

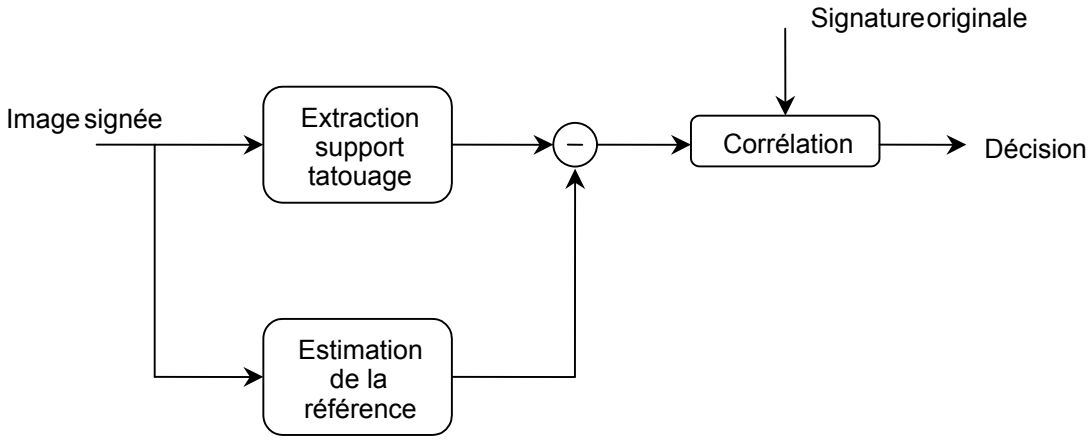


Figure 1.4 – *Mode d'extraction semi-aveugle*

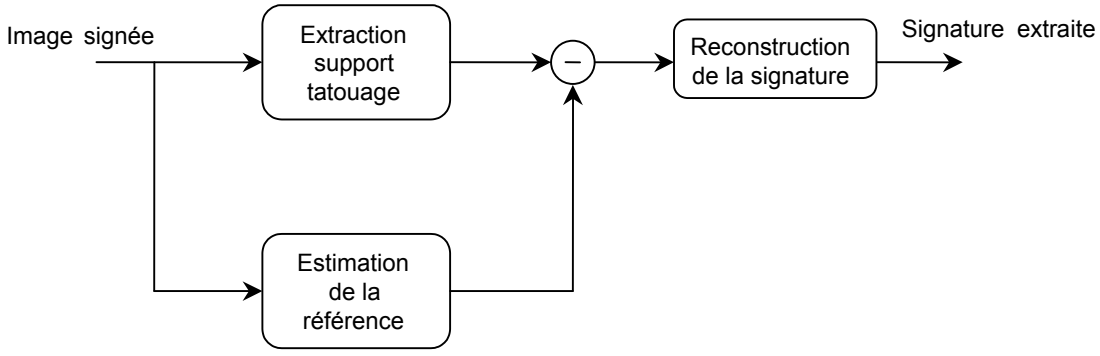


Figure 1.5 – *Mode d'extraction aveugle*

3. Applications visées

3.1. Protection des droits d'auteur

La protection des droits d'auteur a été une des premières applications étudiée en tatouage d'image. Ce service reste cependant toujours d'actualité et concerne encore la majorité des publications. L'objectif est d'offrir, en cas de litige, la possibilité à l'auteur ou au propriétaire d'une image d'apporter la preuve qu'il est effectivement ce qu'il prétend être, et ce même si l'image concernée a subi des dégradations par rapport à l'original. La mise en place d'un tel service doit respecter les trois contraintes suivantes :

Préserver la qualité de l'image : la distorsion liée à l'insertion de la marque dans l'image doit être la plus faible possible, de manière à ce que la qualité visuelle de l'image tatouée soit quasi identique à l'originale. Malheureusement cette notion d'invisibilité est très subjective et difficile à modéliser. D'autant plus qu'elle dépend de nombreux facteurs, tels que : la nature de l'image à tatouer (peinture, image médicale, photo satellite, etc.), la qualité du document original (plus une image est de bonne qualité, plus il est difficile de garantir l'invisibilité de la marque) et des conditions de visualisation (problème bien connu en codage de source avec pertes).

Garantir la non-ambiguïté de la preuve : le tatouage doit constituer une preuve irréfutable. Pour cela il convient d'assurer l'unicité (éviter les problèmes de collision) et de l'authenticité de l'identifiant, mais également de dater le dépôt (au cas où l'image aurait été tatouée plusieurs fois avec des marques différentes). Pour cela, il convient de définir des protocoles stricts excluant toute ambiguïté.

Assurer la robustesse des éléments de preuve (tatouage) : l'algorithme utilisé doit être capable d'extraire correctement la marque cachée, même si l'image a été manipulée. Dans le paragraphe 6 de ce chapitre, nous détaillons plus précisément cet aspect du tatouage d'image, ainsi que les différentes familles d'attaque.

3.2. Vérification de l'intégrité du contenu d'une image

L'idée de base consiste à utiliser les techniques de tatouage d'image afin de cacher dans certaines zones de l'image des informations sur d'autres zones. Ces informations servent à alerter l'utilisateur face à une éventuelle modification ou découpe de l'image par une personne non autorisée et à localiser précisément les régions manipulées, voire éventuellement à les restaurer. Ce service remet partiellement en cause les paramétrages usuellement établis dans le cadre d'un service classique de droits d'auteur, notamment en termes de quantité et nature des informations à cacher, de

robustesse, etc. Nous aborderons plus en détail les différents aspects de ce service dans le chapitre 5. Mais on peut déjà se demander s'il est préférable d'utiliser un tatouage fragile (ou semi-fragile), un tatouage robuste, ou opter au contraire pour une technique faisant appel à une signature externe.

3.3. Gestion du nombre de copies d'une image

Contrairement aux données de nature analogique pour lesquelles une succession de reproductions entraîne rapidement une perte significative de la qualité, les données numériques peuvent être dupliquées quasiment à l'infini. Dans ce contexte, une personne ayant accès à ce type de données et au matériel adéquat, est potentiellement capable de les reproduire bit à bit à l'identique. Il est évident que cette personne, si elle est malintentionnée, peut ensuite redistribuer illégalement des copies avec une qualité égale au document d'origine. Face à cette nouvelle situation, certaines instances proposent d'utiliser des techniques de tatouage afin de limiter l'ampleur de ce phénomène. C'est le cas des systèmes DVD, où un filigrane numérique indiquera si la vidéo peut être lue et/ou recopiée. Ce procédé n'est bien entendu fiable que si tous les constructeurs de lecteurs et d'enregistreurs de DVD tiennent compte de l'indicateur de copie. Le système de marquage, n'est pour l'heure pas encore arrêté. Le *Data-Hiding Sub-Group* (DHSG) du *Copy Protection Technical Working Group* (CPTWG) du *DVD Forum* a reçu sept propositions de watermarking qui ont depuis fusionné en deux : IBM, NEC, Macrovision, Digimarc, Philips (galaxy group) et Hitachi, Pioneer, Sony.

3.4. Non répudiation d'accès et suivi de copies

Ce service est une bonne illustration de la complémentarité entre les techniques de cryptographie et le tatouage d'image. Soit une image diffusée sous forme chiffrée, interdisant qu'une personne puisse avoir accès à son contenu lors de la transmission de celle-ci. Au niveau du destinataire, on procède simultanément au décryptage et au tatouage de l'image de telle sorte que si l'utilisateur remet illégalement en circulation cette image, il sera alors possible de remonter à la source du délit grâce à l'identifiant du destinataire (ou « fingerprint ») caché dans l'image. Ce type d'application ne va pas sans poser de sérieux problème en matière de sécurité, dans la mesure où il existe plusieurs copies d'une même image contenant des tatouages différents. En effet, sous l'hypothèse que l'ensemble des contributions des tatouages soit à moyenne nulle, plusieurs clients malhonnêtes peuvent tenter de reconstruire l'image originale en calculant une image moyenne à partir de leur image respective (*c.f.* paragraphe 6.2 traitant des attaques par collusions).

3.5. Contrôle d'accès

L'objectif est d'ôter tout intérêt commercial à l'image en y superposant un tatouage (voir

Figure 1.6 – *Contrôle d'accès par masquage partiel d'une image*

Figure 1.6). Seules les personnes ayant les droits d'accès sont en mesure d'inverser le processus de marquage de manière à reconstituer l'image originale. L'avantage de cette méthode par rapport à un système de cryptage classique où l'image est totalement inintelligible, réside dans le fait que le tatouage peut être porteur d'informations relatives à l'image. On peut, par exemple, y faire figurer l'adresse où commander l'image en clair, le nom de la société, etc.

3.6. Autres services

Bien évidemment il existe d'autres applications possibles en dehors de celles décrites précédemment, et dans des domaines autres que des services de sécurité. On peut imaginer utiliser une technique de tatouage d'image pour faciliter la recherche dans une base de données multimédia en cachant par exemple dans le document des informations textuelles sur son contenu. Le tatouage d'image trouverait également sa place dans un système de montage vidéo où il pourrait servir par exemple à étiqueter les différentes séquences. Un des avantages de cette technique par rapport à des méthodes traditionnelles, est qu'elle offre la possibilité de retrouver facilement l'origine d'un extrait à partir d'un enregistrement quelconque. Récemment, certains auteurs [RM01], [BMPB01] ont proposé d'utiliser les techniques de tatouage afin de détecter et corriger des erreurs de transmission dans des systèmes de transmission de vidéos numériques. D'une manière générale, le principal avantage du tatouage d'image par rapport à d'autres formes de marquage, par le biais de fichiers d'en-tête par exemple, réside dans son indépendance vis-à-vis du support de l'image et des manipulations éventuelles.

4. Etat de l'art des techniques de tatouage d'image

Cette section n'a pour objectif de dresser un panorama complet et exhaustif des différentes techniques de tatouage d'image. Le but de cette partie est de présenter simplement, dans leurs grandes lignes, les méthodes les plus significatives du domaine, afin de familiariser le lecteur avec les notions clés classiquement utilisées en tatouage d'image et d'appréhender les contributions incluses dans les prochains chapitres. Même si la plupart des techniques présentées ici ont connu des améliorations significatives ces dernières années, elles permettent néanmoins d'appréhender la problématique, les difficultés et les limites inhérentes au tatouage d'image.

Les algorithmes de tatouage se distinguent les uns des autres essentiellement par les quatre points clés suivants :

- La manière de sélectionner les points (ou blocs) dans le document hôte qui porteront l'information cachée ;
- Le choix d'un espace de travail pour réaliser l'opération d'enfouissement (dans le domaine spatial ou transformé comme DCT, ondelettes, Fourier-Melin, etc.) ;
- La stratégie utilisée pour mettre en forme l'information à cacher avant son enfouissement : redondance, codes correcteurs, bits de resynchronisation (la contribution de cette thèse portera essentiellement sur cet aspect) ;
- La manière de mélanger intimement le message avec le signal hôte (modulation) ; l'idée de base consiste le plus souvent à imposer une relation binaire entre les bits du message et des caractéristiques choisies de l'image porteuse.

Il existe principalement deux grandes familles de méthodes : celles qui opèrent dans le domaine spatial et celles qui opèrent dans un domaine transformé ; plus quelques méthodes originales.

4.1. Choix des éléments de l'image recevant l'information de signature

4.1.1. Le domaine spatial

L'algorithme « Patchwork » a été proposé par Bender *et al.* en 1995 [BGM95]. Cet algorithme opère directement dans le domaine spatial, c'est-à-dire au niveau même des pixels. Cette technique appartient à la famille des méthodes de tatouage à réponse binaire. Elle permet de répondre par oui ou par non à la question : une personne est-elle en possession de l'information secrète ayant permis de générer le tatouage ? Dans cette méthode on ne cherche en aucun cas à extraire le tatouage.

Le principe du « patchwork » est de sélectionner, à l'aide d'une clé secrète K_s, une séquence S_a de n couples de pixels (A_i, B_i), puis de modifier très légèrement l'image en augmentant d'une unité le niveau de gris des pixels de type A_i et en diminuant d'un niveau de gris les pixels de type B_i. Considérons la somme S des différences de luminance des couples de pixels sélectionnés. Une personne ne disposant pas de la clé sera incapable de régénérer la bonne séquence S_a et obtiendra S=0 (hypothèse de travail). Seule la personne disposant de la clé sera en mesure d'obtenir « la bonne valeur » de S, c'est-à-dire $2\times n$. Ces propos doivent être modérés par l'objection suivante : un individu malveillant peut tout à fait appliquer le même algorithme avec une autre clé $K'_s \neq K_s$ et ainsi créer une nouvelle image signée. On tombe ici sur un problème de signatures multiples qui dépasse largement le cadre de cet algorithme.

Cette méthode de base n'est bien évidemment pas très robuste ; cependant, différentes extensions de cet algorithme ont vu le jour [BQM95]. Elles permettent par exemple d'accroître la résistance du système à des opérations de filtrage sur l'image en considérant non plus des couples de pixels mais des couples de blocs. L'emploi de plusieurs séquences aléatoires orthogonales dans le but de dissimuler plusieurs bits (1 bit par séquence aléatoire) a également été proposé.

4.1.2. Le domaine DCT

De nombreuses méthodes ont été développées à partir des connaissances acquises auparavant en codage de source. Les auteurs de ces méthodes espèrent ainsi en travaillant dans le domaine DCT [RY90], anticiper et prévenir au moins les attaques liées à une compression Jpeg. Ils espèrent également pouvoir travailler plus rapidement en couplant le tatouage d'images avec le codage de source. En d'autres termes, le tatouage est réalisé directement sur le flux compressé. Le dernier point opérant en faveur d'un tatouage dans le domaine DCT est qu'il est possible de bénéficier, au moins en partie, des études psychovisuelles déjà menées en codage de source pour gérer les problèmes de visibilité.

L'algorithme suivant a été proposé par Koch et Zhao en 1995 [KZ95]. Elle est à la base de nombreux travaux, notamment de projets de recherche européens comme le projet Talisman [Tal98]. Les étapes d'insertion et d'extraction peuvent se résumer comme suit :

a) Algorithme d'insertion

1. Soit une séquence de k bits (b_1, …, b_k) à cacher dans l'image.
2. Sélectionner dans l'image, selon une clé secrète, k blocs B (B_1, …, B_k) de taille 8×8.
3. Calculer les coefficients DCT (a_{11}, …, a_{88}) de chaque bloc sélectionné.
4. Pour i allant de 1 à k :
   Soient a_{mn} et a_{op} deux des coefficients DCT du bloc B_i, et b_i le bit à cacher.

- Si (b_i = « 1 » ET $(a_{mn})_i > (a_{op})_i$) OU (b_i = « 0 » ET $(a_{mn})_i > (a_{op})_i$), alors ne rien faire.
- Sinon modifier les valeurs des coefficients $(a_{mn})_i$ et $(a_{op})_i$ pour que la relation précédente soit vérifiée.

5. Calculer la DCT inverse à partir des valeurs ainsi modifiées afin d'obtenir l'image tatouée.

L'opération d'extraction est duale de l'opération d'insertion, et se déduit immédiatement :

b) Algorithme d'extraction

1. Retrouver les blocs marqués grâce à la clé secrète.
2. Calculer les coefficients DCT associés aux blocs sélectionnés.
3. Comparer les valeurs de $(a_{mn})_i$ et $(a_{op})_i$ afin de déterminer si le bit caché était un « 0 » ou un « 1 ».

Cette méthode pose plusieurs problèmes. Tout d'abord, elle utilise la notion de blocs, ce qui la limite à cacher au mieux un bit par bloc. Ensuite, cette méthode s'appuyant sur un découpage prédéfini de l'image, sera rapidement mise en difficulté face à des attaques géométriques, même très simples. D'autre part, le compromis visibilité *vs* robustesse est difficile à régler. Si on choisit les coefficients (m, n) et (o, p) dans les basses fréquences, la marque sera robuste mais très visible. Inversement, des coefficients pris dans les hautes fréquences, conduiront à une marque invisible mais très peu robuste face à des attaques photométriques, même faibles. Il est à noter qu'entre la version présentée ici et la version actuelle de cette méthode, de nombreuses améliorations ont été apportées, comme l'utilisation d'un coefficient DCT supplémentaire permettant une mise en œuvre plus souple.

4.1.3. L'espace engendré par la transformée de Fourier-Mellin

Des transformations géométriques de l'image tatouée conduisent fréquemment à l'impossibilité d'extraire le tatouage pour de nombreux algorithmes. Ce constat a conduit à envisager l'implantation du tatouage dans un espace transformé présentant une invariance aux opérations géométriques usuelles de l'image. Dans l'article [RP97], Ó Ruanaidh *et al.* préconisent l'usage de la transformée de Fourier-Mellin (*c.f.* Figure 7) pour assurer la restitution du tatouage malgré que l'image ait subi une translation et/ou une rotation et/ou un changement d'échelle. L'espace invariant est obtenu ; d'une part grâce à la propriété de la transformée de Fourier qui répercute une translation de l'image exclusivement sur la phase et laisse invariant l'amplitude ; et d'autre part, par un changement de repère, de cartésien vers logarithmique-polaire, qui ramène les opérations de rotation et de changement d'échelle à une translation (*c.f.* Figure 1.8).

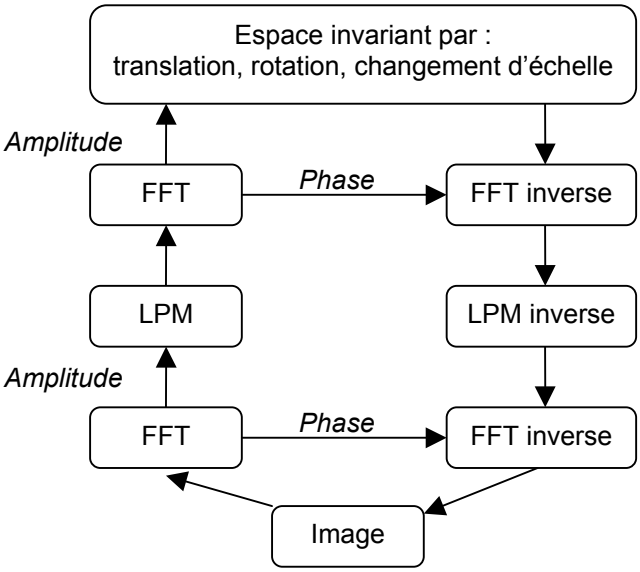


Figure 1.7 – *Construction d'un espace invariant par translation, rotation et changement d'échelle*

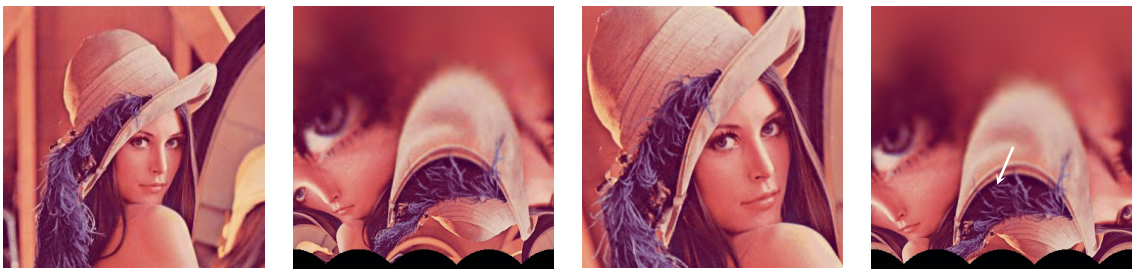


Figure 1.8 – *Exemple de transformation par LMP dans le domaine spatial*

4.1.4. Décomposition de l'image en canaux perceptifs

L'évaluation de la qualité des images est une préoccupation constante des utilisateurs de systèmes de traitements d'image. Les modèles psychovisuels introduits pour évaluer cette qualité considèrent communément le système visuel humain comme un ensemble de canaux [DVM98a], [Win98], [WLB95] par lesquels sont transmis différents types d'informations au cerveau. Les techniques de tatouage, dans le but d'améliorer l'invisibilité de la signature dans l'image, ont cherché à utiliser ces travaux et en particulier les effets de masquage [BBCP98]. Delaigle *et al.* [DVM97], [DVM98b] ont développé un modèle perceptif permettant d'évaluer analytiquement la visibilité ou l'invisibilité d'une marque afin de pouvoir éventuellement rétroagir sur l'algorithme de tatouage.

L'algorithme proposé réalise une décomposition de l'image originale en canaux. La détermination de chaque canal est faite sur la base de caractéristiques fréquentielles (module et phase) ainsi que de la localisation dans le champ de vision. Toute la difficulté consiste à identifier des canaux en adéquation avec les critères perceptifs humains. L'hypothèse sous-jacente consiste à admettre que deux signaux à l'intérieur d'un même canal ne pourront être distingués par l'œil humain.

Le domaine ondelette : les transformées en ondelettes qui, tout comme la transformée DCT fait l'objet de nombreuses études dans le contexte du codage, ont également trouvé un écho dans la communauté du tatouage d'image [KH98a], [WJ98], [XBA98], [ZLL99]. Cet intérêt repose d'une part sur les analyses en termes psychovisuels menées afin d'optimiser les tables de quantifications des codeurs, d'autre part sur l'aspect multi-échelle de telles transformées propice à une répartition plus robuste du tatouage.

4.2. Ajout de redondance à la signature

La taille des signatures nécessaires à l'identification sans ambiguïté d'un individu est de l'ordre de quelques octets, or une image constitue un volume d'information binaire bien supérieur. Il est donc légitime d'ajouter de la redondance à la signature originale afin d'accroître la robustesse du tatouage face à des manipulations de l'image. Les méthodes présentées ci-après sont d'ailleurs très largement inspirées des techniques utilisées dans le domaine des communications numériques.

4.2.1. Etalement de spectre

Les techniques d'étalement de spectre ont été introduites pour résoudre des problèmes de communications sur des canaux bruités entre plusieurs utilisateurs [PSM82]. En complément du gain en robustesse que procurent ces techniques vis-à-vis des imperfections du canal de transmission, elles permettent d'assurer la confidentialité entre les différentes communications via un même canal de transmission.

Etalement par séquence directe : cette technique réalise l'étalement directement dans le domaine temporel (ou spatial). Un signal à bande étroite S peut être étalé spectralement par modulation à l'aide d'un signal à large spectre PN (s'apparentant à un bruit blanc). Cette modulation confère au signal résultant S_e les caractéristiques spectrales de PN. Si le signal de transmission présente un évanouissement dans la bande étroite où se situe le signal à transmettre S, la technique d'étalement permettra d'assurer une bonne transmission de ce signal. La connaissance du signal PN permet de démoduler le signal S_e et de reconstruire le signal S.

Etalement par saut de fréquence (frequency hopping) : Le principe consiste à moduler le si-

gnal d'origine par une porteuse dont la fréquence varie de manière aléatoire. Le signal résultant est ainsi réparti dans l'ensemble de la gamme de fréquence où est choisie la porteuse. Cette technique permet également d'assurer un cryptage du message. En effet, la démodulation du signal nécessite la connaissance de la porteuse qui a été utilisée pour porter le signal or celle-ci dépend d'une clé secrète.

4.2.2. Codes correcteurs

Des articles [DDNM98], [DVM98b], [DBQM96], [HPRN98] font référence à une utilisation potentielle de codes correcteurs d'erreurs afin d'augmenter les performances en termes de robustesse des algorithmes de tatouage. L'emploi de tels codes apparaît en effet naturel si l'on examine le problème de la robustesse du tatouage sous l'angle de la communication d'un signal sur un canal bruité. L'usage des codes correcteurs dans le cadre du tatouage d'image reste un problème ouvert, requérant la conception de codes compacts capables de prendre en compte la diversité des attaques. Nous aborderons de manière plus détaillée l'utilisation de codes correcteurs dans le processus de mise en forme du message dans le chapitre 3.

4.3. Fusion des données : image et signature

4.3.1. Techniques de modulation

Modulation de phase

La transformée de Fourier d'une image réelle est généralement de nature complexe ; elle possède donc un module et une phase. Des études expérimentales ont montré que l'information contenue dans la phase était prépondérante sur celle contenue dans l'amplitude dans la représentation de l'image. Cette constatation conduit à introduire le tatouage au niveau de la phase pour, d'une part s'assurer qu'une tentative de suppression du tatouage engendrera inévitablement des dégradations importantes de l'image ; d'autre part les techniques de modulation de phase sont reconnues comme étant plus robustes au bruit que les techniques de modulation d'amplitude. Un tel système privilégie a priori l'aspect robustesse sur l'aspect visibilité.

Modulation d'amplitude

Dans l'article [KJB97], il est proposé, pour des raisons de visibilité, de réaliser l'insertion de la

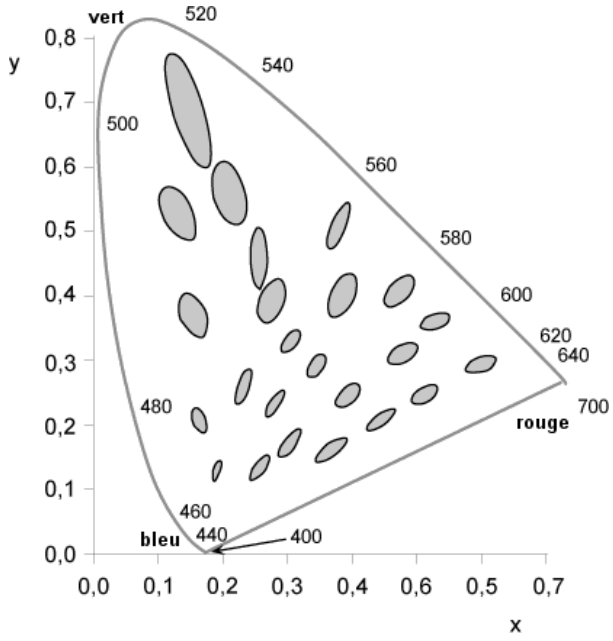


Figure 1.9 – *Ellipses de Mac Adam*[1]

signature par une modulation d'amplitude de la composante bleue d'une image couleur (RGB) plutôt que dans la composante de luminance. Ce choix peut sembler, à première vue, injustifié dans la mesure où l'œil humain est particulièrement sensible aux variations de tons de bleus (le pouvoir de discrimination des couleurs du système visuel humain n'est pas uniforme- ellipses de Mac Adam – *c.f.* Figure 1.9). L'explication est que l'œil est plus sensible à des variations de luminosité et de contraste qu'à des variations de nuances de couleur. En effet, si l'on considère la contribution des différentes composantes RGB dans la composante de luminance :

$$L = 0{,}299 \times R + 0{,}587 \times G + 0{,}114 \times B \tag{1.1}$$

On s'aperçoit rapidement que la composante bleue est minoritaire. Par conséquent, l'impact en termes de distorsion, sera plus faible que si l'on avait modulé directement la luminance.

[1] *Ellipse ≡ plus petite différence perceptible entre deux couleurs proches (i.e. toutes les couleurs appartenant à une même ellipse sont jugées identiques à la couleur centrale)*

4.3.2. Tatouage par quantification des coefficients DCT

Modification de la fonction d'arrondi

Dans l'une de leurs méthodes, Matsui et Tanaka [MT94] proposent d'introduire la signature binaire lors de l'étape de quantification des coefficients DCT. Plus précisément, ils modifient la fonction d'arrondi. Par rapport à la fonction classiquement utilisée dans un codeur Jpeg, on ne considère plus l'entier le plus proche mais l'entier pair (respectivement impair) le plus proche lorsque l'on désire introduire un bit de signature à 1 (respectivement 0). L'erreur de quantification ainsi créée est donc directement corrélée avec la signature. Les auteurs admettent que la dégradation engendrée par cette erreur est suffisamment faible pour ne pas entraîner de gêne visuelle. Si tel n'est pas le cas, il est possible de réduire le pas de quantification des tables de coefficients DCT pour se positionner à un niveau de dégradation acceptable. Malheureusement, cette opération entraîne une moindre résistance du tatouage. Le pas de quantification fournit donc un paramètre de réglage du compromis robustesse *vs* visibilité.

Définition d'une relation de N-uplet de coefficients

La technique exposée précédemment introduit un bit de tatouage au niveau de chaque coefficient DCT en ne tenant pas compte des coefficients voisins. Koch et Zhao [KZ95], [Zha96] ont cherché à rétablir une notion de voisinage en proposant une modulation différentielle des coefficients DCT.

Superposition des coefficients DCT de l'image et du tatouage

Cette technique inspirée des méthodes stéganographiques [JJ98a], [JJ98b] est particulièrement indiquée lorsque le tatouage est de même nature que les données à tatouer. Autrement dit, dans le contexte des images, si le tatouage est lui-même une image (par exemple un logo).

4.3.3. Tatouage par substitution de blocs : codage fractal

La plupart des méthodes de tatouage introduisent la signature dans l'image par le biais d'une perturbation de la quantification de certaines grandeurs caractéristiques de l'image. Dans le cadre par exemple des méthodes basées sur la quantification de certains coefficients DCT, les effets visuels résultant de cette manipulation sont parfois difficilement maîtrisables. Le laboratoire de Traitement des Signaux de l'EPFL propose une approche différente reposant sur le codage fractal [PJ96]. Le codage fractal est basé sur la définition d'une association entre différentes régions de

l'image. Cette association est réalisée selon un critère d'auto-similitude fondé sur la minimisation de l'erreur quadratique entre les blocs cibles et les blocs sources transformés. Pour un bloc cible donné, la recherche du bloc source associé s'effectue dans deux fenêtres de recherche centrées sur le bloc cible. La méthode de tatouage proposée modifie cette recherche en définissant deux sous fenêtres comme indiquées sur la figure 1.10. L'insertion du message consiste à :

a) Algorithme d'insertion

1. Tirer aléatoirement $N=r\times l$ blocs cibles dans l'image ; l étant le nombre de bits du message et r le nombre de répétitions.
2. Pour chacun des N blocs cibles, effectuer la recherche du bloc source associé dans la fenêtre de recherche de type 0 (respectivement de type 1) si le bit associé du message à pour valeur 0 (respectivement 1).
3. Pour chacun des blocs cibles non précédemment traités, la recherche du bloc source s'effectue sans contrainte sur la fenêtre de recherche. En d'autres termes, la recherche est réalisée dans la fenêtre constituée de l'union des fenêtres de type 0 et 1.
4. A partir du code IFS obtenu lors des deux précédentes étapes, effectuer le processus de décodage standard aux techniques de codage fractal afin d'obtenir l'attracteur qui constitue l'image tatouée.

L'extraction du message est réalisée de manière duale de l'insertion.

a) Algorithme d'insertion

1. Grâce à la clé secrète on régénère le signal aléatoire donnant accès aux N blocs cibles potentiellement porteurs des bits message.
2. Pour chacun des N blocs cibles on fait une recherche du bloc source associé par minimisation de l'erreur quadratique. Cette recherche s'effectue dans la région définie par l'union des fenêtres de type 0 et 1.
3. La décision sur la valeur du bit extrait est prise en fonction de la région d'appartenance du bloc source. Si le bloc source appartient à la fenêtre de type 0 le bit associé vaut 0, sinon le bit prend la valeur 1.

L'intérêt de cette approche est de mettre à profit certaines propriétés d'invariance propres aux fractales afin de pouvoir prévenir certaines attaques et récupérer la marque sans avoir recours à l'image originale. Cette propriété est à relativiser dans la mesure où l'image tatouée est potentiellement sujette à des manipulations de type compression par exemple, dès lors, on ne dispose plus rigoureusement de l'attracteur au moment de l'extraction du message.

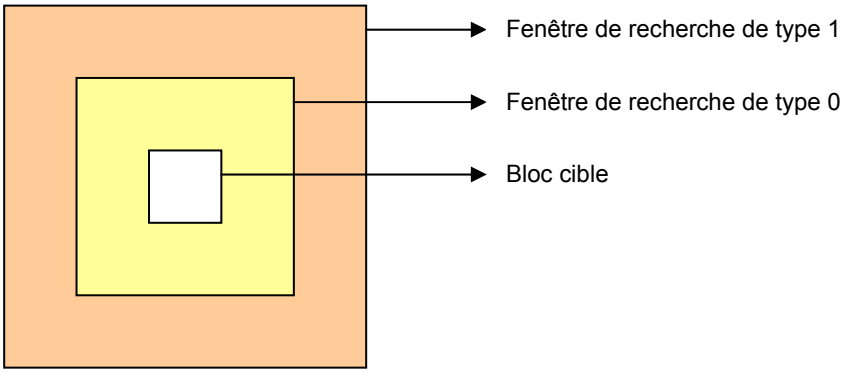


Figure 1.10 – *Bloc cible et les deux sous fenêtres de recherche associées*

4.4. Optimisation du détecteur

L'extraction de la signature est composée d'opérations duales de l'insertion, auxquelles il faut ajouter diverses techniques propres à la phase d'extraction visant à accroître la robustesse du tatouage. Certains algorithmes [KL98] pratiquent un filtrage de l'image tatouée avant d'entreprendre la vérification du tatouage. D'autres algorithmes [Kut98] confectionnent un tatouage comprenant des bits dont les valeurs sont prédéfinies afin de permettre une première estimation de l'attaque qu'a pu subir l'image. Ces derniers algorithmes sont propices à la mise en place de seuils de décisions adaptatifs. Enfin, les tests d'hypothèses font également parti des outils usuellement utilisés dans le cadre de problèmes où une prise de décision intervient. En tatouage d'image, ils trouvent particulièrement leur intérêt lorsque la signature est connue et qu'il s'agit de vérifier sa présence dans telle ou telle image, le plus souvent par corrélation.

4.5. Remarques concluantes

Cet état de l'art nous a permis de dégager les grandes lignes conduisant à la conception d'un système de tatouage et a révélé la diversité des techniques mises en œuvre. Néanmoins, la robustesse du tatouage semble surtout être vérifiée vis-à-vis des standards de compression, elle n'est pas assurée face à une combinaison d'attaques de nature différente. La plupart des efforts se sont portés vers une gestion appropriée du compromis robustesse *vs* visibilité, le plus souvent au détriment d'autres aspects comme la capacité d'insertion ou le mode d'extraction. Les algorithmes les plus robustes réalisent rarement l'extraction en mode aveugle (*i.e.* on ne présuppose ni la connaissance de l'image originale et ni la connaissance du tatouage) ou ont une capacité nulle (aucun bit de caché).

5. Evaluation de la distorsion introduite par le tatouage

Afin de réduire la distorsion visuelle introduite par l'insertion d'une marque dans une image, les algorithmes de tatouage tirent généralement partie des caractéristiques du système visuel humain (HVS) en cachant la marque dans les régions les moins sensibles de l'image (*e.g.* les contours et les zones de textures). Une bonne métrique de qualité d'image se doit également de prendre en compte les caractéristiques HVS. Malheureusement, le PSNR, qui est la mesure la plus couramment utilisée (comme l'ensemble des métriques basées pixel) ne tient absolument pas compte du contenu fréquentiel de l'image. Bien que servant toujours de référence, le PSNR n'est pas la métrique la plus appropriée au contexte du tatouage d'image. Certains auteurs préconisent alors l'utilisation d'autres métriques plus adaptées, telles que le wPSNR (weighted PSNR) [VPIP01] (ou PSNR pondéré), le MPSNR (masked PSNR) [BF96] ou bien encore la mesure de Watson [Wat93].

5.1. PSNR pondéré (wPSNR)

La métrique classique du PSNR est donnée par la formule suivante :

$$PSNR = 10.\log_{10} \frac{\max(x)^2}{\|x'-x\|^2} \tag{1.2}$$

où x' représente l'image tatouée et x l'image originale. Cette définition du PSNR pénalise l'ajout de bruit (i.e. le tatouage) de la même manière quelles que soient les régions de l'image. Alors qu'en raison des phénomènes de masquage, la perception d'un bruit est plus importante dans les régions uniformes que dans les zones constituées de contours ou texturées. Le wPSNR (1.4) se distingue du PSNR en différenciant des régions visuellement différentes. Une manière simple d'attribuer un poids à une zone de l'image en fonction de son contenu fréquentiel est d'utiliser fonction NVF (Noise Visibility Function). La fonction NVF dans le cas où l'on considère que l'image suit un modèle non stationnaire gaussien est la suivante :

$$NVF(i,j) = \frac{1}{1+\sigma_x^2(i,j)} \tag{1.3}$$

où $\sigma_x^2(i,j)$ représente la variance locale de l'image dans une fenêtre centrée sur le pixel de coordonnées *(i,j)*.

$$wPSNR = 10.\log_{10} \frac{\max(x)^2}{\|NVF(x'-x)\|^2} \tag{1.4}$$

5.2. Mesure de Watson

Le modèle de Watson [Wat93, CMB01] a été développé à l'origine pour évaluer la qualité des images compressées par Jpeg. Ce modèle estime la distorsion perceptible par l'œil humain dans le domaine TCD (Transformée en Cosinus Discrète). Son application s'effectue sur des blocs DCT 8×8 de l'image. L'erreur de quantification est pondérée par un seuil de visibilité qui dépend essentiellement de trois facteurs : un modèle de sensibilité de l'œil (*i.e.* table déterminée expérimentalement donnant les réponses de l'œil à des stimuli isolés), ainsi que deux modèles de masquage (un adapté à la luminance et un autre au contraste). Pour obtenir une mesure globale de l'erreur perceptible, les erreurs de quantification pondérées pour chaque couple de fréquences sont sommées sur chacun des blocs constituant l'image. Les résultats sont ensuite sommés sur l'ensemble de l'espace TCD en utilisant la sommation de Minkowski. La distorsion est alors exprimée en terme de nombre différences perceptibles (JNDs).

6. Evaluation de la robustesse d'un tatouage

Pour toutes les applications du tatouage d'image, à l'exception peut-être de l'intégrité, la robustesse de la marque est un des critères fondamentaux à prendre en compte lors de la mise au point d'un algorithme. En effet la marque doit pouvoir résister d'une part à des manipulations liées à l'utilisation ou la diffusion de l'image, telle qu'une conversion de format ou une impression ; et d'autre part à des attaques malveillantes, plus spécifiques, dont le but est de la détruire ou rendre impossible son extraction. De plus, la plupart de ces attaques sont très faciles à mettre en œuvre à l'aide d'outils classiques de traitement d'image. Il faut également noter que la notion de robustesse d'un tatouage est intimement liée à l'impact visuel engendré par les différentes manipulations subies par l'image. Il est clair que des attaques qui entraîneraient des dégradations trop importantes, rendant l'image totalement inexploitable, auraient très peu d'intérêt, même si elles étaient susceptibles de conduire à la neutralisation du tatouage. On distingue principalement trois types d'attaques : les attaques liées au signal, les attaques de nature cryptographique, et les attaques de protocoles. Actuellement, la majorité d'entre elles opèrent directement au niveau du signal, c'est-à-dire au niveau de l'image elle-même.

6.1. Attaques liées au signal

6.1.1. Manipulations courantes en traitement d'image

Il est impossible de dresser un inventaire exhaustif des manipulations pouvant être appliquées à une image tant elles sont nombreuses. De plus ces manipulations peuvent être aisément combinées

entre elles de manière à créer des attaques plus complexes. Nous nous contenterons simplement de présenter brièvement les plus couramment usitées.

Transformations photométriques

- *Ajout de bruit (gaussien, aléatoire uniforme)* – l'ajout involontaire ou délibéré d'un bruit dans l'image peut, lorsqu'il est suffisamment important, avoir pour effet de masquer la marque.

- *Filtres* – les filtres sont un des outils de base du traitement d'image. Ils sont principalement utilisés pour améliorer l'aspect d'une image, en rendant, par exemple, celle-ci plus « douce » (filtres passe-bas, médian, « anti-aliasing », etc.), ou en faisant ressortir des détails (réhaussement des contours). Les filtres ont généralement pour effet d'atténuer le tatouage dans l'image.

- *Compression* – tout système de tatouage d'image doit pouvoir résister jusqu'à un certain niveau de compression. En effet, la majorité des données qui circulent sur les réseaux sont sous forme compressée (avec ou sans perte). Jpeg est l'un des algorithmes de compression d'image les plus utilisés actuellement.

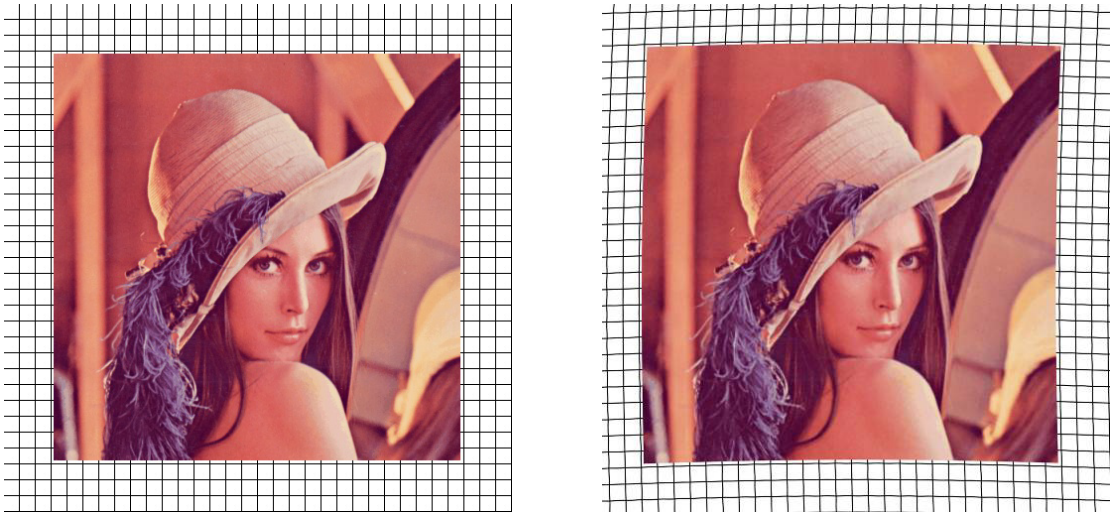
- *Quantification des couleurs* – cette opération est couramment appliquée lorsque l'on convertit une image dans un format avec une palette de couleur réduite (Gif par exemple). Bien souvent, elle est suivi d'un « dithering » (tramage) de manière à simuler des couleurs supplémentaires, qui a pour effet de rajouter du bruit à l'image.

- *Correction gamma et égalisation d'histogramme* – ce type d'opération est fréquemment utilisé pour améliorer le contraste d'une image.

- *Passage en niveaux de gris* – la conversion d'une image couleur en niveaux de gris peut poser des problèmes entre autres aux systèmes de tatouage d'image qui insèrent la marque dans une composante de l'image autre que la luminance (comme le bleu par exemple).

Transformations géométriques globales et locales

Les manipulations géométriques, mêmes très simples, sont des attaques particulièrement sévères, face auxquelles beaucoup d'algorithmes de tatouage d'image se révèlent inefficaces, en particulier lorsque l'on impose une extraction en mode aveugle. En effet, pour la majorité des méthodes proposées, l'opérateur d'extraction a besoin de connaître la position exacte de la marque dans l'image. Or, les distorsions géométriques ont pour effet d'introduire une désynchronisation entre le tatouage contenu dans l'image et le détecteur. De ce fait, bien que la marque soit encore présente,



(a) Image originale **(b)** Image attaquée

Figure 1.11 – *Illustration des déformations géométriques aléatoires engendrées par Stirmark.*

les bits extraits ne correspondent plus à ceux qui ont été cachés.

Parmi les attaques de nature géométrique, on distingue :

- *Les transformations affines (locales ou globales)* – les principales sont les translations, les rotations et les changements d'échelle.
- *Les symétries axiales (horizontale et verticale)* – elles ne sont pas forcément décelables si l'image présente naturellement un axe de symétrie ou aucune information textuelle, mais peuvent suffire à piéger l'algorithme d'extraction, si celui-ci ne les prend pas en compte.
- *Recadrage et extraction* – ces opérations visent à ne préserver que la partie « intéressante » de l'image. Elles sont généralement problématiques dans la mesure où elles introduisent une désynchronisation (similaire à celle produite par une translation) et où la taille de l'image recadrée risque d'être insuffisante pour contenir une marque robuste.
- *Suppression de lignes et de colonnes* – ces manipulations sont généralement invisibles, mais suffisent à créer un décalage significatif pouvant rendre difficile l'extraction du tatouage.

Manipulations combinant des transformations photométriques et géométriques

Voici également quelques exemples courants de manipulations combinant des transformations photométriques et géométriques.

- *Montage* - cet effet est très utilisé en infographie. Il consiste à extraire des éléments de différentes images et à les combiner judicieusement ensemble de manière à créer une nouvelle image. Le challenge en tatouage d'image, est de pouvoir retrouver, à partir du montage, la provenance des différents éléments le constituant.

- *Conversion Numérique / Analogique / Numérique* - dans le cadre d'images fixes, ce type de conversion correspond principalement à une impression suivie d'une numérisation à l'aide d'un scanner [LC99]. Les distorsions subies par l'image sont multiples et dépendent de la qualité des équipements utilisés. Néanmoins, on peut modéliser cette opération comme étant composée généralement : d'un double ré-échantillonnage, d'un ajout de bruit lié principalement aux capteurs du scanner, d'une requantification des couleurs, ainsi qu'un certain nombre de déformations géométriques liées au positionnement du document.

6.1.2. Manipulations malveillantes

Il s'agit ici de manipulations spécifiques dont le but est de détruire délibérément le tatouage contenu dans l'image. Parmi les outils ou « crackers » proposant ce type d'opération, on distingue ceux qui perturbent l'image de telle manière que, même si la marque reste présente dans l'image tatouée, celle-ci est très difficile à extraire sans avoir recours à l'image originale. En effet, l'image est simplement déformée de telle sorte que l'opérateur d'extraction n'est plus en phase avec l'opérateur d'insertion. Parmi les programmes réalisant de telles perturbations, les plus référencés actuellement sont Unzign et surtout Stirmark. UnZign modifie sensiblement la taille de l'image, quant à Stirmark, il propose une attaque qui génère des distorsions géométriques locales et aléatoires dans l'image (voir Figure 1.11). Ces déformations de l'image ont la particularité d'être différentes à chaque fois, rendant ainsi l'attaque totalement imprévisible. De plus les distorsions engendrées sont, pour la majorité des images, imperceptibles, et suffisent à mettre en défaut la grande majorité des algorithmes de tatouage. D'autres outils tentent au contraire de lessiver la marque (processus de « dewatermarking » [RDDC02]). Dans ce cas, le problème est encore plus grave dans la mesure où s'ils y parviennent, il n'y a plus aucun espoir de récupérer le tatouage, puisque ce dernier n'est plus présent dans l'image. Enfin, on peut également imaginer des algorithmes pouvant modifier une image tatouée afin de substituer une marque par une autre. A l'extraction, malgré une clé secrète pourtant associée à un marquage donné, le détecteur retournerait une autre marque que celle attendue.

Attaque mosaïque

Bien qu'il ne s'agisse pas à proprement parlé d'une manipulation au sens du traitement d'image, la décomposition d'une image sous forme de mosaïque doit être prise en compte, dans la mesure où elle pose de graves problèmes aux systèmes automatiques de détection de tatouage [PA98]. Une

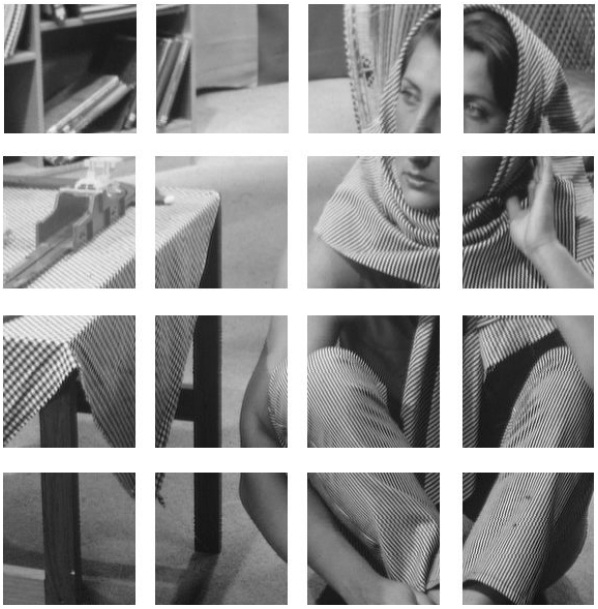


Figure 1.12 – *Illustration de l'attaque « mosaïque ».*

mosaïque est formée par un partitionnement de l'image originale en sous images, chacune des sous images étant juxtaposée aux autres, afin de reconstituer visuellement l'image d'origine (voir Figure 1.12). La mosaïque est réalisée de telle sorte que la taille des sous images soit insuffisante pour contenir un tatouage robuste. De ce fait, les robots traqueurs ou *spiders* sont dupés par la mosaïque, car ils sont incapables de considérer l'image dans sa globalité, et ne peuvent par conséquent détecter le tatouage.

Attaque par auto-similarités

Cette attaque [RDDC02] exploite les auto-similarités naturellement présentes dans les images afin de perturber le tatouage. Les auto-similarités peuvent être vues comme une forme particulière de redondance. En effet, au lieu de rechercher la corrélation entre les pixels adjacents, on s'intéresse ici à des corrélations entre des parties plus ou moins espacées dans l'image. L'idée des auto-similarités a déjà été exploitée avec succès pour la compression fractale [Fis94].

Le principe de base de cette attaque consiste à substituer des parties de l'image par d'autres parties similaires d'elle-même (voir Figure 1.13), modulo des transformations photométriques (*e.g.* dilatation, contraction et décalage) et géométriques (*e.g.* rotations, symétries, changement d'échelle). L'objectif est de distordre le signal de tatouage sans pour autant introduire de déformations géométriques au niveau de l'image. L'image est donc parcourue bloc par bloc (range blocks). Chacun de ces blocs est substitué par un bloc cible (domain block) qui lui ressemble le plus au sens de l'erreur

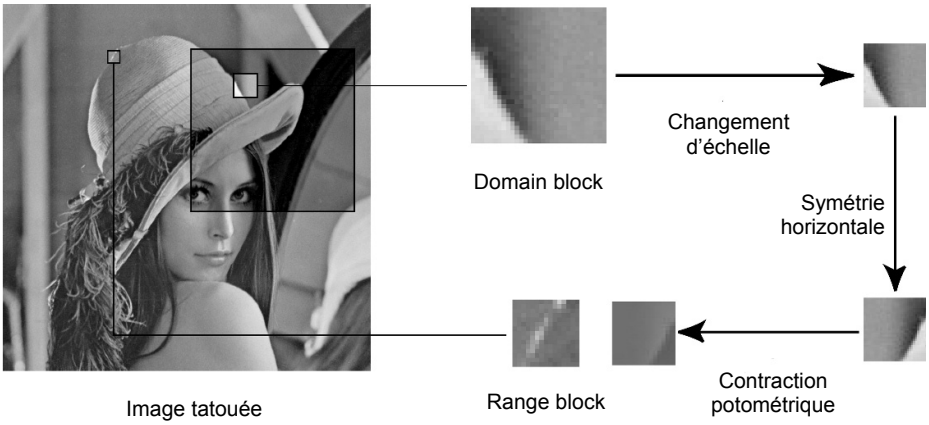


Figure 1.13 – *Principe des auto-similarités*

quadratique. La technique de codage par les auto-similarités doit cependant être adaptée. Dans le contexte de l'attaque, la reconstruction de l'image tatouée ne doit pas être parfaite au risque de ne pas retirer le tatouage. Il est donc nécessaire lors de la recherche des blocs cibles de rajouter une condition supplémentaire sur l'erreur minimale d'appariement des blocs. Chaque bloc de l'image tatouée est donc remplacé par le « domain block » qui obtient la plus petite erreur quadratique au dessus d'un certain seuil. La valeur de ce seuil dépend principalement de l'algorithme de tatouage utilisé et du contenu de l'image. Ce seuil peut être choisi de manière empirique afin de trouver le meilleur compromis entre l'efficacité de l'attaque et la qualité de l'image attaquée. Afin de minimiser également la distorsion introduite par l'attaque dans le cas d'images en couleur, seules la ou les composantes contenant le tatouage dans un espace colorimétrique donné sont attaquées.

6.2. Attaques de nature cryptographique

Moins courantes, ces attaques suivent le modèle des attaques classiques en cryptographie. Certaines de ces attaques, telles que les « Brute Force Attacks », ont pour objectif de découvrir la clé secrète utilisée pour insérer la marque, en essayant de manière exhaustive toutes les clés possibles. Bien évidemment, ce genre d'attaque est très coûteux en temps de calcul, et n'est réellement efficace que sur des algorithmes utilisant des clés de petite taille.

L'attaque oracle, quant à elle, est plus spécifique aux algorithmes de tatouage asymétriques. Si un pirate dispose du décodeur public, il peut appliquer de petites modifications successives à l'image jusqu'à ce que le décodeur ne décèle plus de marque. De cette façon, il a l'assurance de ne pas avoir dégradé l'image plus que nécessaire. Cette attaque, suggérée pour la première fois par Perrig [Per97], a depuis fait l'objet de nombreuses études, et des analyses théoriques ainsi que des contre-mesures possibles ont été récemment publiées.

Enfin, les attaques par collusion [BS98] sont sans doute les attaques cryptographiques les plus difficiles à se prémunir en tatouage d'image. La collusion fait référence à un ensemble d'utilisateurs malveillants qui mettent en commun leurs connaissances (*i.e.* leurs images tatouées) afin de produire illégalement des images non tatouées. Il existe deux types de collusions :

- **Collusion de type I** : le même tatouage est inséré dans des images différentes. La collusion consiste alors à estimer (*e.g.* en faisant la différence entre l'image tatouée et une version filtrée passe-bas), dans un premier temps, le tatouage contenu dans chacune des images marquées, puis à faire une combinaison linéaire des différentes estimations. Une fois le tatouage correctement estimé, il suffit de le soustraire aux images tatouées pour obtenir des images ne contenant plus de marque. De la même manière, il est possible de tatouer une image en lui ajoutant simplement le tatouage estimé.

- **Collusion de type II** : différents tatouages sont insérés dans différentes copies de la même image. Dans ce cas, la collusion consiste simplement à faire une combinaison linéaire des différentes images tatouées (*e.g.* moyenne) afin d'obtenir une nouvelle image ne contenant plus aucune marque. En effet, généralement, la moyenne de différentes marques tend vers zéro.

6.3. Attaques de protocoles

Les attaques de protocoles se distinguent des autres familles d'attaques, dans la mesure où leur but n'est pas de détruire ou d'empêcher la détection de la marque par des manipulations de l'image. Ces attaques s'en prennent directement aux protocoles de l'application elle-même. Une des premières attaques de ce type a été proposée par Craver *et al.* [CMYY98]. Les auteurs introduisent la notion de tatouage inversible, et montrent que pour assurer certains services de sécurité, il est impératif d'utiliser des tatouages non inversibles. Ce qui signifie en terme de tatouage d'image qu'il ne doit pas être possible d'extraire une signature depuis une image qui n'a pas été tatouée.

Signatures multiples

Un autre problème, classiquement rencontré dans un système de protection des droits d'auteur, est celui des signatures multiples ou du sur-marquage des images. Bien que la plupart des produits commerciaux refusent (théoriquement) de tatouer une image contenant déjà une signature, le problème existe bel et bien, et peut, dans certaines circonstances, conduire à une situation de « deadlock ». En effet, rien n'empêche une personne malintentionnée de trouver un moyen de court-circuiter ce test ou tout simplement de retatouer l'image avec un autre logiciel (la plupart se contentant uniquement de vérifier si l'image a été tatouée par leur algorithme).

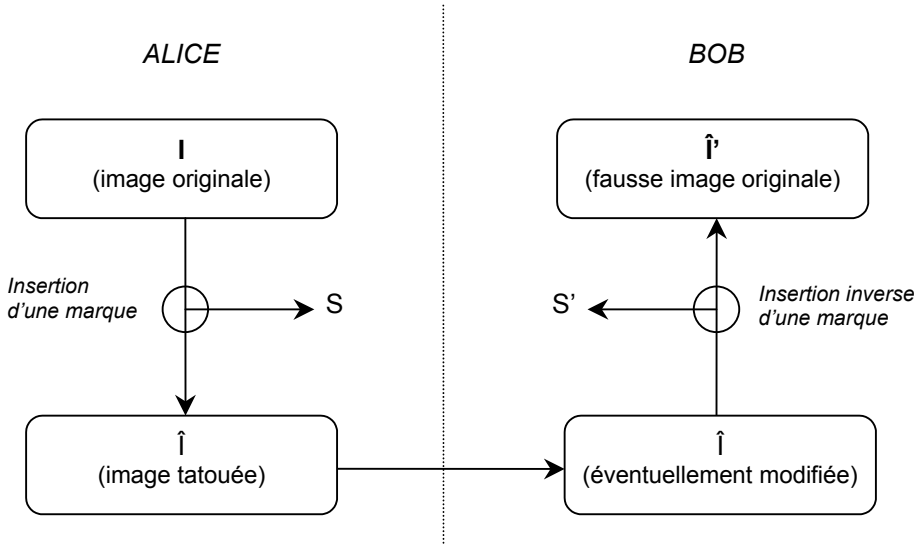


Figure 1.14 – *Le problème du "Deadlock"*

Intéressons-nous maintenant d'un peu plus près au cas d'une image contenant plusieurs signatures. Il est d'usage en sécurité d'utiliser les prénoms Alice et Bob plutôt que les lettres A et B.

Alice a une image *I*, elle la marque avec une signature *S* et génère ainsi l'image tatouée $\hat{I}$ qu'elle rend publique. Bob marque à son tour l'image $\hat{I}$ avec une signature *S'* et obtient l'image $\hat{I}'$. Il est alors évident qu'à la fois Alice et Bob peuvent réclamer la paternité de l'image $\hat{I}'$. Pour résoudre ce problème on a recourt à l'image originale. En effet, à partir de l'image *I* Bob n'est pas en mesure d'extraire sa signature, alors qu'au contraire Alice peut exhiber la sienne à partir de la supposée image originale de Bob à savoir $\hat{I}$.

Afin de prendre en défaut la protection mise en œuvre par Alice, Bob peut tenter de fabriquer une pseudo image originale $\hat{I}'$ à partir de *I* tel que $\hat{I}$ et *I* contiennent sa signature. On rétablirait ainsi une symétrie parfaite entre les deux prétendants à la propriété de l'image. La tâche de Bob paraît, au premier abord, délicate, puisqu'il n'a à aucun moment accès à la véritable image originale. Comment pourrait-il donc y introduire sa marque ? Le principe est de créer un processus de tatouage inverse. Bob ne cherche plus à dissimuler une marque *S'* dans l'image $\hat{I}$, mais à la soustraire, créant ainsi une pseudo image originale $\hat{I}'$. Bob a donc créé une nouvelle (fausse) image originale et n'a pas modifié l'image signée $\hat{I}$, mais est néanmoins en mesure de prouver que sa signature est présente dans $\hat{I}$ tout comme dans *I*. Ceci conduit à une situation indécidable ou « deadlock » (Figure 1.14). Le seul moyen de résoudre ce problème consiste à introduire dans le protocole une tierce personne de confiance jouant le rôle d'entité de certification.

La « Copy Attack »

La « Copy Attack », développée par Kutter [KVH00], consiste à copier la signature contenue dans une image tatouée dans une autre image. Cette attaque pose de nouveaux problèmes, particulièrement pour des applications où le tatouage est utilisé à des fins d'identification ou de copyright. Le procédé proposé pour réaliser cette opération ne présuppose aucune connaissance *a priori* sur l'algorithme de tatouage utilisé, ni d'information supplémentaire, telle que la clé secrète. L'attaque se décompose en trois étapes. Dans un premier temps, on procède à une estimation du tatouage dans le domaine spatial à l'aide de processus de débruitage. En d'autres termes, la marque prédite est obtenue par différence entre l'image tatouée et sa version débruitée. La deuxième étape du processus consiste à adapter ensuite la marque à l'image cible de manière à maximiser l'énergie du tatouage en tenant compte de contraintes d'invisibilité. Finalement, la marque ainsi obtenue est insérée par addition dans la nouvelle image.

6.4. Difficultés pour évaluer et comparer les performances des algorithmes

Cet aspect du problème du tatouage d'image a longtemps été négligé. Ceci est en partie dû à la jeunesse du domaine, mais aussi à la difficulté de procéder à une évaluation rigoureuse. De nombreuses publications [FG99a], [KP99], [PA98], [Pet98] abordent cependant le sujet et des outils et des projets de recherche Octalis [PKB99], Optimark [Opti], CheckMark [Chec] et Certimark [Cert] tentent d'y apporter des solutions. Parmi ceux-ci, un des plus attendu est sans doute le projet Certimark, qui regroupe des universitaires et des industriels européens. Un des objectifs de ce projet est de fournir un outil permettant d'évaluer les performances des algorithmes de tatouage en terme de robustesse mais en tenant compte également des aspects capacité et visibilité, ainsi que de l'application visée.

La plupart des tests proposés jusqu'à présent sont basés sur des méthodes empiriques encore incapables de gérer la complexité des problèmes (facteur subjectif humain pour l'aspect visibilité, multiplicité des attaques pour la robustesse). De plus, une confidentialité importante entoure la majorité des algorithmes de tatouage d'image. Leurs auteurs sont en effet très réticents à divulguer leurs méthodes ou à mettre un prototype à disposition pour des tests. Ce comportement est motivé d'une part par une compétition et des enjeux économiques très importants, et d'autre part par la crainte d'un acharnement qui conduirait à faire apparaître leurs faiblesses. Néanmoins il serait dangereux, selon le principe bien connu de Kerckhoffs [Ker83], de considérer un algorithme secret comme sûr.

7. Conclusion

Bien que le tatouage d'image soit un domaine relativement jeune, les acquis disponibles en codage de source, codage canal, cryptographie et théorie de l'information d'une part, ainsi que les enjeux industriels et économiques d'autre part, ont permis l'émergence rapide d'une grande variété de techniques, et font que des produits commerciaux (encore imparfaits) sont d'ores et déjà disponibles.

Même si actuellement, l'utilisation première du tatouage d'image reste la défense des droits d'auteur, de nombreuses autres applications sont envisageables. L'explosion du numérique et l'engouement grandissant pour le multimédia ont fait apparaître de nouveaux besoins en termes de sécurité et de recherche d'information. Dans ce contexte, beaucoup d'applications trouvent un intérêt à ce qu'un filigrane numérique, visible ou non, soit entrelacé aux données multimédia manipulées, afin d'assurer des services aussi variés que le contrôle d'accès, l'indexation, l'intégrité ou la traçabilité. Les possibilités dans ce domaine sont vastes et encore peu explorées. Mais déjà, les nouvelles générations d'algorithmes de tatouage d'image tendent à se spécialiser afin de répondre au mieux aux besoins particuliers de chaque application.

Cependant, bien que les techniques de tatouage d'image aient, d'un point de vue algorithmique, atteint une certaine maturité, la mise en place d'un service complet de watermarking reste encore problématique. En effet, si l'on s'intéresse par exemple au problème des droits d'auteur, on remarque que de nombreuses difficultés subsistent, tant techniques que juridiques. On peut se poser la question sur la manière de contrôler l'utilisation des images « copyrightées ». On peut également se demander qu'elle sera la valeur juridique accordée au tatouage d'image ? A qui reviendra la tâche d'assurer ce service : le propriétaire ou une tierce personne ?

Malgré toutes ces questions en suspens, il semble bien que le tatouage d'image ait acquis ses premières lettres de noblesse. Le mot clé « watermarking » figure désormais dans les documents relatifs à JPEG-2000, MPEG-4, MPEG-21 et DVD.

Chapitre 2

Algorithme de tatouage basé sur un modèle affine d'IFS

1. Rappels sur le codage fractal

Dans cette section, nous aborderons uniquement les grandes lignes du codage fractal. Le lecteur non familier avec cette technique est invité à consulter les références suivantes [Fis95, Jac90, Jac92, RD95]. Apparu à la fin des années 80, suite aux travaux de Barnsley et Jacquin [BJM+88], le codage fractal trouve son origine dans les systèmes de fonctions itérées (IFS) développés par Huchinson [Hut81].

1.1. Photocopieuse à réductions multiples

Imaginons une photocopieuse dont le but est de recopier l'entrée trois fois en divisant par deux son échelle (*c.f.* Figure 2.1). Imaginons ensuite qu'à chaque cycle de la copie, la photocopieuse fonctionne en boucle, c'est-à-dire que le document photocopié est remis en entrée de la photocopieuse. Alors, à force de recopier avec les mêmes transformations (dans notre cas, division par 2 de l'échelle puis recopie de la source 3 fois), la sortie convergera vers ce que l'on appelle un attracteur, c'est-à-dire une image qui lorsqu'elle est remise en entrée de cette photocopieuse donne en sortie la même image. La transformation utilisée en exemple est l'une des très nombreuses transformations possibles. On peut immédiatement comprendre que d'autres transformations conduiraient à d'autres attracteurs. Considérons maintenant deux images totalement différentes, et appliquons à chacune d'elles les mêmes transformations. Comme le montre la figure 2.2 , les deux images (a et b) qui au départ étaient dissemblables donnent lieu au même attracteur (c) (dans notre exemple, le triangle

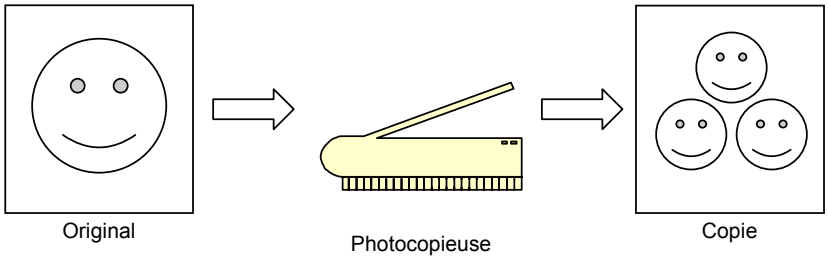


Figure 2.1 – *Photocopieuse à réductions multiples*

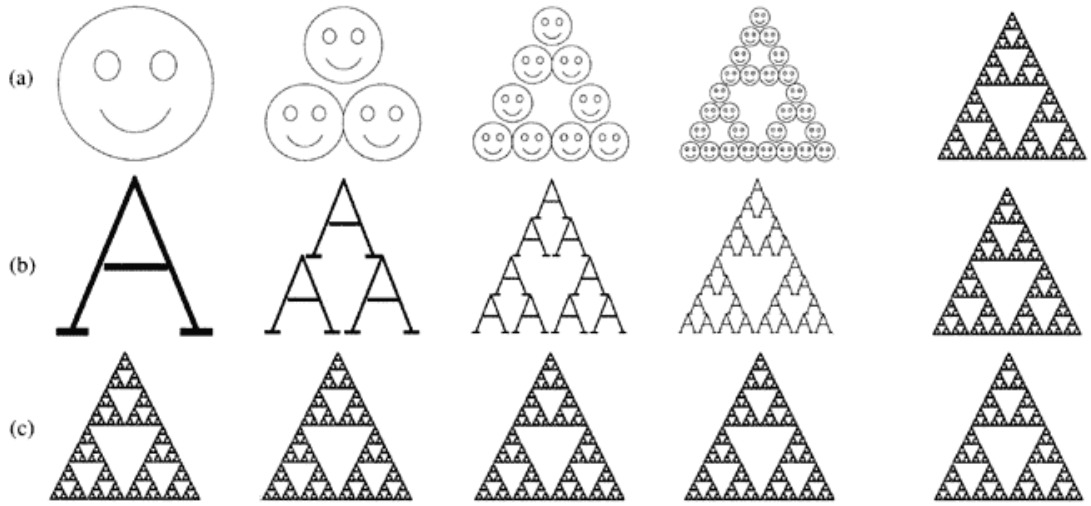


Figure 2.2 – *Fonctions itérées*

de Sierpinski). Le résultat final dépend uniquement des transformations utilisées lors de la reproduction Ainsi grâce à de telles photocopieuses il est possible de représenter des objets complexes à l'aide d'un jeu d'instructions (paramètres de transformations) relativement réduit. La fougère de Barnsley (*c.f.* Figure 2.3) en est un parfait exemple. Cette fougère peut être décrite intégralement à l'aide des quatre transformées suivantes :

$$
\begin{aligned}
w_1\begin{bmatrix} x \\ y \end{bmatrix} &= \begin{bmatrix} 0.85 & 0.04 \\ -0.04 & 0.85 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0.02 \\ 0.08 \end{bmatrix} \\
w_2\begin{bmatrix} x \\ y \end{bmatrix} &= \begin{bmatrix} -0.13 & 0.24 \\ -0.22 & 0.20 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0.12 \\ -0.27 \end{bmatrix} \\
w_3\begin{bmatrix} x \\ y \end{bmatrix} &= \begin{bmatrix} 0.18 & -0.24 \\ 0.21 & 0.20 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} -0.12 \\ -0.30 \end{bmatrix}
\end{aligned}
\tag{2.1}
$$

Figure 2.3 – *Fougère de Barnsley*

$$w_4\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0.16 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0 \\ -0.42 \end{bmatrix}$$

Il est clair que le codage de cette image pixel par pixel (bitmap) ou même sous une forme compressée représente un volume d'information bien supérieur au simple codage des coefficients des matrices de transformation. On entrevoit ici la possibilité d'utiliser un tel système pour effectuer un codage de source efficace. Il est possible de décrire une image uniquement par un ensemble de transformations.

1.2. Système d'IFS

L'IFS (Iterated Function System) est un moyen de représenter les fractales. Il est défini par un espace métrique complet et un ensemble d'applications contractantes. L'ensemble recherché, qui peut être fractal, est l'attracteur de l'IFS. Il est invariant selon toutes les applications qui composent l'IFS.

Définition 1 (Application contractante) : Soit Ω l'espace des images muni d'une métrique d. Soient *u* et *v* deux images appartenant à cet espace. On dit que la transformation *w* est contractante si et seulement si :

$$\forall u, v \in \Omega \quad d(w(u), w(v)) < d(u, v) \tag{2.2}$$

Définition 2 (Attracteur) : Soit w une transformation contractante définie sur l'espace des images (Ω, d). On appelle attracteur associé à w, l'élément x_a de (Ω, d) s'il existe défini par :

$$x_a = \lim_{n \to \infty} w^n(x_0) \tag{2.3}$$

où $w^n(x_0)$ désigne la n^{ième} composition de w par elle-même calculée au point x_0.

Théorème 1 (Invariance) : Tout attracteur x_a d'une transformation contractante w est invariant à cette transformation.

$$x_a = w(x_a) \tag{2.4}$$

La notion de transformée contractante induit une invariance de l'attracteur par rapport à l'image initiale. L'idée sous-jacente au codage fractal est de déterminer une image x_a proche de l'image originale sur le plan perceptif et pouvant être représentée par un ensemble de transformations W auquel on associe un processus itératif. Ce processus consiste, à partir de n'importe quelle image, à appliquer récursivement les transformations associées. L'image initiale permet simplement de spécifier la résolution de l'image finale.

1.3. Codage fractal

L'objectif du codage est d'assurer la convergence du processus itératif vers un point fixe (attracteur x_a) constituant une approximation aussi fidèle que possible de l'image originale x_c. Le problème du codage peut être formulé en termes d'optimisation sous contraintes dont :

1. La première contrainte est constituée par le modèle de transformations W compact adopté. Généralement il s'agit d'un modèle affine comprenant les 8 isométries du plan (les rotations 90, 180 et 270 degrés, les symétries horizontale, verticale et diagonales, ainsi que l'identité), un sous-échantillonnage et une transformation photométrique ($s.z+o$) où s (contraction/dilatation) et o (décalage) sont des paramètres à estimer et z le niveau de gris.

2. La seconde stipule que les fonctions recherchées doivent être contractantes afin d'assurer la convergence du processus itératif de décodage.

Le principe du codage consiste d'une part à écrire la propriété d'invariance d'un attracteur et d'autre part à postuler l'existence d'un attracteur proche de l'image que l'on souhaite coder. On est alors amené, sous les contraintes (1) et (2), à chercher parmi l'espace des solutions, la transformée W tel que $d(W(x_c), x_c)$ tend vers zéro.

Ce problème d'optimisation présente une complexité combinatoire beaucoup trop élevée pour être résolu directement. Dans la pratique, on procède à des simplifications en substituant des transformations locales W_k blocs à blocs à la transformation globale W. Le nouveau problème consiste à déterminer les paramètres s_k, o_k et l'isométrie, associés à chaque bloc B_k constituant une partition de l'image. Le flux codé est constitué d'une liste d'indexes faisant référence pour chaque bloc à la transformation associée (l'ensemble des transformées étant comprise dans un dictionnaire).

1.4. Décodage fractal

Le décodage consiste à dérouler le processus itératif. Pour ce faire, à partir des indexes contenus dans le flux compressé, on identifie parmi le dictionnaire de transformations, celle qui doit être appliquée au bloc B_k donné. Une fois cette opération réalisée pour chaque bloc B_k, on applique les transformées W_k et l'on réitère le processus. L'initialisation du processus correspond au choix de l'image de départ sur laquelle sont appliquées les transformées.

2. Algorithme de tatouage

L'algorithme de tatouage utilisé a été développé à l'Institut Eurécom au cours de la thèse de Stéphane Roche [Roc99] et a fait l'objet de plusieurs dépôts de brevets [DR99a, DR99b]. Son principe repose sur un modèle affine d'IFS et l'utilisation de la notion d'auto-similarité pour décrire une image. L'idée est d'utiliser les propriétés d'invariance propres au codage fractal, telles que l'invariance par réhaussemment du niveau de gris moyen, l'invariance par réhaussemment du contraste et les pseudo-invariances par translation et changement d'échelle, pour assurer la robustesse du marquage. Nous décrivons dans ce chapitre uniquement les grandes lignes de l'algorithme.

2.1. Insertion du tatouage

L'algorithme d'insertion du tatouage se décompose en trois grandes étapes :

- Détermination du support du tatouage,
- Mise en forme et cryptage de la marque,
- Fusion du tatouage avec l'image.

2.1.1. Détermination du support du tatouage

Soit une image originale I_{orig}, on peut considérer que tout algorithme de codage avec pertes décompose cette image en la somme de deux images. La première I_{codage} représente l'image codée

avec pertes et la seconde I_{erreur} correspond à l'image résiduelle.

$$I_{\text{orig}} = I_{\text{codage}} + I_{\text{erreur}} \tag{2.5}$$

Considérons un modèle de transformée affine similaire à celui utilisé en codage :

$$W(I) = A.I + O_{moy} \tag{2.6}$$

La matrice A représente les opérations d'amplification de contraste ainsi que les transformations géométriques reliant les blocs sources et les blocs cibles au cours d'une itération du processus de décodage. Le vecteur O_{moy} représente le rehaussement ou l'abaissement du niveau de gris moyen entre les deux blocs. Le théorème du collage stipule que :

$$I_{orig} \approx A.I_{orig} + O_{moy} \tag{2.7}$$

Dans le contexte d'un codage par IFS, I_{codage} porte le nom d'attracteur I_{attract} et est défini comme la limite du processus de décodage itératif. L'image I_{erreur} apparaît naturellement comme un support favorable à la dissimulation d'un tatouage invisible.

2.1.2. Mise en forme et cryptage du tatouage

L'unité d'insertion de la signature dans l'image étant le bit, la première étape consiste donc à convertir le message à cacher (une chaîne de caractères ou un logo) en une image binaire.

Cette opération terminée, on rajoute ensuite de la redondance à la signature, afin de se prémunir contre une perte partielle d'information. Cette duplication des bits du message est effectuée à deux niveaux. Tout d'abord localement, par un sur-échantillonnage (d'un facteur 3 par défaut) de l'image binaire originale (*c.f.* Figure 2.4). Ceci a pour effet de décaler le tatouage vers les basses fréquences, puisque la valeur d'un bit est la même pour toute la période de sur-échantillonnage, rendant ainsi la marque plus résistante aux filtres passe-bas. Le second niveau de redondance est global. Il consiste à dupliquer la signature sur-échantillonnée horizontalement et verticalement, constituant ainsi un pavage de mêmes dimensions que l'image à tatouer (*c.f.* Figure 2.5). Cette forme de répétition est recommandée pour pallier à une défaillance locale de l'algorithme. En effet, il peut arriver que des régions de l'image soient trop endommagées au cours de certaines manipulations ou tout simplement inadéquates (pour des raisons de visibilité) pour recevoir le tatouage.

Pour illustrer la mise en forme de la signature, nous avons choisi un logo binaire qui, outre son aspect pratique pour la protection des droits d'auteur d'une image, permet d'illustrer les différentes étapes de la mise en forme de la marque.

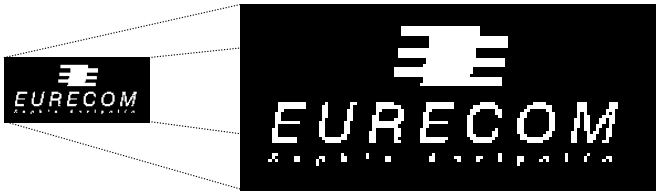


Figure 2.4 – *Sur-échantillonnage du logo d'un facteur 3*

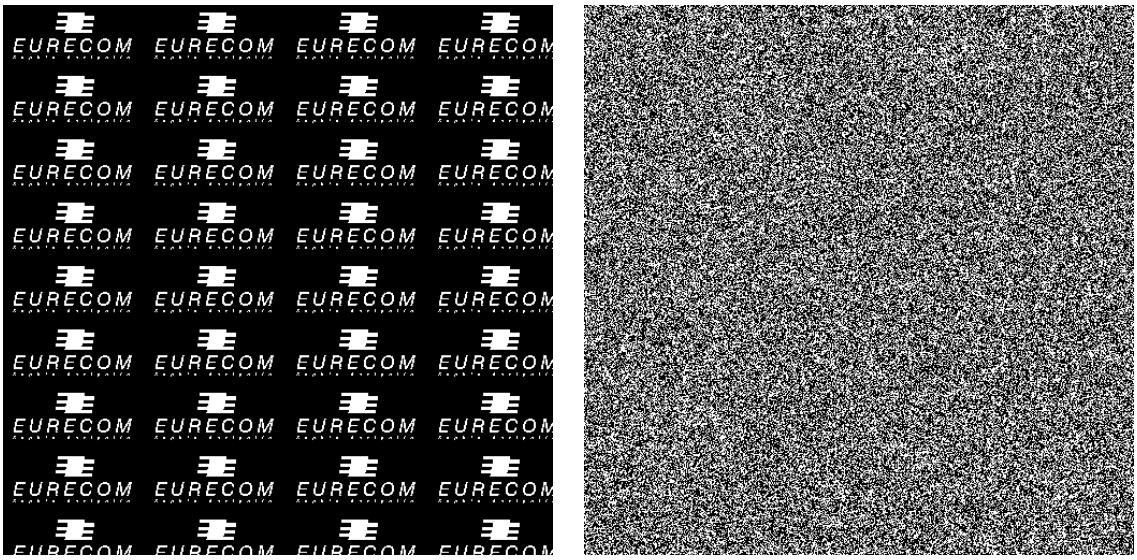


Figure 2.5 – *Duplication et cryptage du logo*

Finalement, la marque binaire obtenue est bruitée globalement. Cette opération est effectuée en réalisant un « ou exclusif » entre la signature et une séquence binaire pseudo-aléatoire (dépendant de la clé secrète) de même fréquence. Les raisons de cette opération sont multiples : sécuriser le message par un cryptage global, minimiser l'impact visuel en supprimant les motifs répétitifs, et assurer une répartition équitable entre les bits à « 0 » et les bits à « 1 ».

2.1.3. Incertitude sur la localisation du tatouage

Bien que l'opération « ou exclusif » réalisée précédemment, place toute personne ne disposant pas de la clé secrète, dans l'incapacité de connaître la valeur des bits de la signature cachée, cela ne garantie en aucune manière que le tatouage est protégé contre une attaque visant à l'altérer ou à le rendre irrécupérable. En effet, pour certains types de signatures, par exemple un identifiant sous la

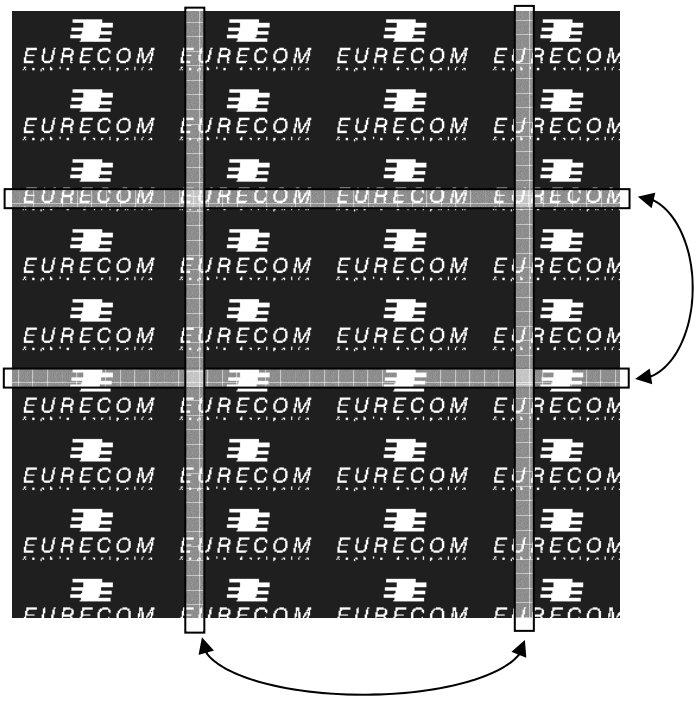


Figure 2.6 – *Incertitude sur la localisation du tatouage*

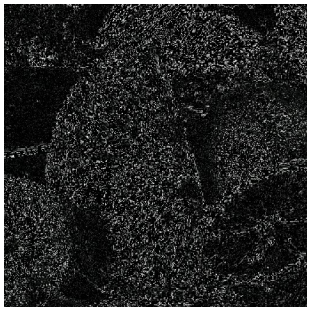
forme d'un numéro (*e.g.* plaque d'immatriculation), la perte d'un seul bit entraîne une invalidité du tatouage. De ce fait, on peut imaginer une attaque triviale, mais néanmoins efficace qui consiste à choisir un bit quelconque de la signature et à perturber l'ensemble de ses occurrences dans l'image tatouée en se basant sur la mise en forme de la signature. Il apparaît nécessaire de mettre en place un système cryptographique permettant de casser l'association directe entre un bit du message et ses différentes duplications dans l'image. Une solution simple consiste à permuter de manière pseudo aléatoire (en fonction de la clé secrète) des lignes et des colonnes de la marque mise en forme introduisant ainsi une incertitude sur la localisation du tatouage. Malheureusement, ce procédé cryptographie basé sur le principe de confusion ne va pas sans poser de problème de robustesse. En effet, certaines manipulations de l'image (*e.g.* translation, recadrage, etc.) peuvent fausser complètement le processus inverse lors de l'extraction rendant impossible la récupération du tatouage. En conclusion, le processus pseudo-aléatoire chargé d'associer les bits du tatouage avec les pixels de l'image doit être totalement indépendant d'éventuelles modifications de l'image.

2.1.4. Fusion du tatouage avec l'image

Enfin, la dernière étape consiste à introduire le tatouage T_{bin} dans l'image. Pour cela on procède à la modification du support en fonction des valeurs des bits de la marque mise en forme. Le codage utilisé est un codage direct, bit à pixel, qui établit une correspondance entre la valeur du bit

image originale

image tatouée



différence absolue (x20)

Figure 2.7 – *Exemple de tatouage*

de tatouage que l'on souhaite cacher et le signe du support. Les règles de modulation sont les suivantes :

$$\text{Si } \left|I_{\text{supp}}(i)\right| > \delta_h \text{ alors } I_{\text{mod}}(i) = I_{\text{supp}}(i) \qquad \text{(pour des raisons de visibilité)}$$

$$\begin{aligned} &\text{Si } T_{bin}(i) = 1 \text{ et } I_{\text{supp}}(i) > 0 \text{ alors } I_{\text{mod}}(i) = I_{\text{supp}}(i) \\ &\text{Si } T_{bin}(i) = 0 \text{ et } I_{\text{supp}} < 0 \text{ alors } I_{\text{mod}}(i) = I_{\text{supp}}(i) \\ &\text{sinon } I_{\text{mod}}(i) = 0 \end{aligned} \tag{2.8}$$

Une fois le support modulé, on le recombine avec l'attracteur, par simple addition, de manière à obtenir l'image tatouée (*e.g.* Figure 2.7).

$$I_{\text{tatouée}} = I_{\text{attract}} + I_{\text{mod}} \tag{2.9}$$

Remarque : il est possible de forcer une erreur minimale au niveau du support afin d'accroître la robustesse du tatouage, en particulier pour les images peu texturées. L'opération consiste, par exemple, à ajouter *a posteriori* une erreur aux pixels du support de faible amplitude. Bien évidemment ce gain en robustesse se fera au détriment de la qualité visuelle de l'image tatouée.

2.2. Extraction du tatouage

L'algorithme d'extraction du tatouage est le dual de l'algorithme d'insertion, et se décompose en trois grandes étapes :

- Séparation des signaux image et support du tatouage,
- Décryptage et resynchronisation,
- Reconstruction de la marque.

2.2.1. Séparation des signaux image et support du tatouage

Cette étape est quasiment identique à celle qui a permis de déterminer le support du tatouage lors de la phase d'insertion. On commence par déterminer l'attracteur IFS de l'image tatouée, puis on calcule le support du tatouage. Le support obtenu est ensuite « seuillé » de manière à éliminer les valeurs de trop forte ou trop faible amplitude, qui ne sont pas porteuses d'information relative au tatouage. Pour cela, on fixe deux seuils (δ_b, δ_h), une borne inférieure et une borne supérieure. Seules les amplitudes comprises, en valeur absolue, entre ces deux bornes sont porteuses d'information (*c.f.* Figure 2.8). On définit ainsi une image ternaire T_{extrait} identifiant les bits valides {+1, -1}.

$$\begin{cases} \text{Si } \delta_b < I_{\text{supp}}(i) < \delta_h \quad \text{alors} \quad T_{extrait}(i) = 1 \\ \text{Si } -\delta_h < I_{\text{supp}}(i) < -\delta_b \quad \text{alors} \quad T_{extrait}(i) = -1 \\ \text{Sinon } T_{extrait}(i) = 0 \quad (\text{i.e. aucune information}) \end{cases} \tag{2.10}$$

Remarque (approximations réalisées lors de l'extraction du tatouage) : Lorsque l'on se place dans un contexte sans attaque (*i.e.* l'image ne subit aucune modification entre la phase de tatouage et la phase d'extraction de celui-ci), les différences numériques entre les images originale et tatouée peuvent avoir un impact sur l'extraction du tatouage. En effet, en mode d'extraction aveugle, l'attracteur n'est plus estimé à partir de l'image originale, mais à partir de l'image tatouée. De ce fait, le support extrait n'est pas identique en tout point au support modulé. Cette variation de l'approximation fractale peut être vue comme l'ajout d'un bruit au niveau du support. Toutefois, nos études ont montré que ce bruit « fractal » reste relativement faible et a peu d'impact sur la robustesse du tatouage (pour des capacités inférieures à 1000 bits).

2.2.2. Décryptage et resynchronisation

Pour pouvoir décrypter correctement la marque contenue dans le support de l'image tatouée, il convient de recaler parfaitement la séquence pseudo-aléatoire avant le XOR. La nature du tatouage permet certes de compenser certaines imperfections dans le positionnement relatif du bruit par rapport au support modulé, mais ne permet pas de corriger des déformations dont l'amplitude dépasse quelques pixels. De ce fait, il est nécessaire de mettre en œuvre un système permettant de remettre en phase le bruit de référence avec le support du tatouage. Ce processus sera discuté plus en détail dans le chapitre 4, où nous aborderons les problèmes de désynchronisation liés à des transformations géométriques globales et locales. Nous présenterons dans ce chapitre deux méthodes originales permettant de détecter et de compenser de telles manipulations.

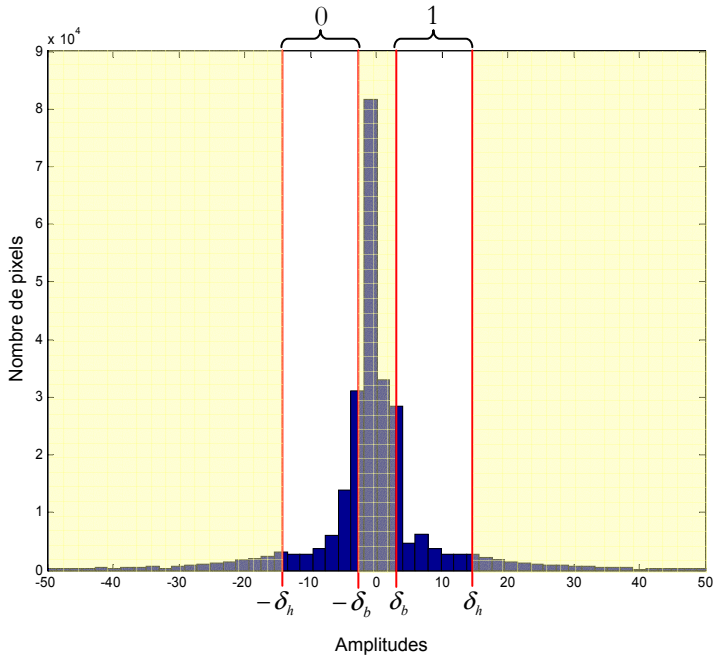


Figure 2.8 – *Histogramme du support extrait*

2.2.3. Reconstruction du message

Une fois le bruit parfaitement recalé et soustrait du support seuillé T_{extrait}, le message est reconstruit en exploitant la redondance R_k de chaque bit.

$$\forall k \in [1..K] \quad b_k = \sum_{i}^{R_k} T^k_{extrait}(i) \tag{2.11}$$

La valeur finale d'un bit m_k du message est ensuite déterminée par vote majoritaire.

$$\begin{cases} \text{Si } b_k < 0 \text{ alors } m_k = 0 \\ \text{Si } b_k \geq 0 \text{ alors } m_k = 1 \end{cases} \tag{2.12}$$

On peut également introduire une notion de cohérence locale afin d'améliorer la fiabilité du message reconstruit. Il s'agit en fait d'éliminer les bits qui ne sont pas cohérents sur la période de sur-échantillonnage. De même, il peut être intéressant de prendre en compte la validité des bits du voisinage. Un bit entouré de bits supposés erronés devra être considéré avec précaution. La valeur finale d'un bit est ensuite déterminée par vote majoritaire des bits considérés valides (non exclus par les processus d'élimination précédents).

2.3. Pertinence du tatouage extrait

2.3.1. Définition de métriques de validité

Le processus de reconstruction permet d'extraire le message contenu dans une image, mais il ne garantit en aucune manière la validité de ce dernier. Il convient donc de définir des métriques permettant d'évaluer la pertinence du message extrait. Pour cela, nous avons établi deux critères ou scores. Le calcul de ces scores repose sur l'évaluation de valeurs de confiance s_k pour chaque bit du message. La valeur s_k peut être interprétée comme le taux majoritaire des suffrages exprimés, c'est-à-dire le rapport du nombre de bits majoritaires sur le nombre de bits valides obs_k.

$$\forall k \in [1..K] \quad obs_k = \sum_{i}^{R_k} \left| T^k_{extrait}(i) \right| \tag{2.13}$$

$$s_k = \frac{|b_k|}{obs_k} \tag{2.14}$$

Le critère $score_{mean}$ est utilisé afin de déterminer si l'image a bien été tatouée. Si sa valeur est supérieure au seuil τ_{mean}, cela indiquera qu'une marque a été détectée, mais sans aucune garantie sur la validité de tous ses bits. Inversement, si le score est inférieur à τ_{mean}, cela signifiera l'absence ou la non détection du tatouage (*i.e.* « bit error rate » supérieur à zéro).

$$score_{mean} = \frac{1}{K} \sum_{k=1}^{K} s_k \tag{2.15}$$

Le critère $score_{min}$ peut être perçu comme une mesure de la fiabilité du message extrait. Si sa valeur est supérieure au seuil τ_{min}, cela voudra dire que le tatouage a été extrait avec une probabilité d'erreur très faible (*i.e.* « message error rate » proche de zéro).

$$score_{min} = Min_{k=1}^{K}(s_k) \tag{2.16}$$

2.3.2. Détermination expérimentale des seuils τ_{mean} et τ_{min}

Détermination de τ_{mean}

Afin de déterminer la valeur de τ_{mean}, nous avons réalisé l'expérience suivante. On a calculé, dans un premier temps, le critère $score_{mean}$ sur une base de 75 images non tatouées en utilisant 100 clés aléatoires par image. (*c.f.* Figure 2.9). Les résultats des tests nous ont permis de fixer la valeur de

τ_{mean} à 0.02. Puis, nous avons vérifié que la valeur du score était supérieure au seuil déterminé précédemment pour des images tatouées (*c.f.* Figure 2.10), mais aussi pour des images ayant subi des attaques (*c.f.* Figure 2.11). Dans un contexte sans attaque, la valeur moyenne de $score_{mean}$ est proche 0,25 ce qui ne laisse aucune ambiguïté sur la détection de la marque.

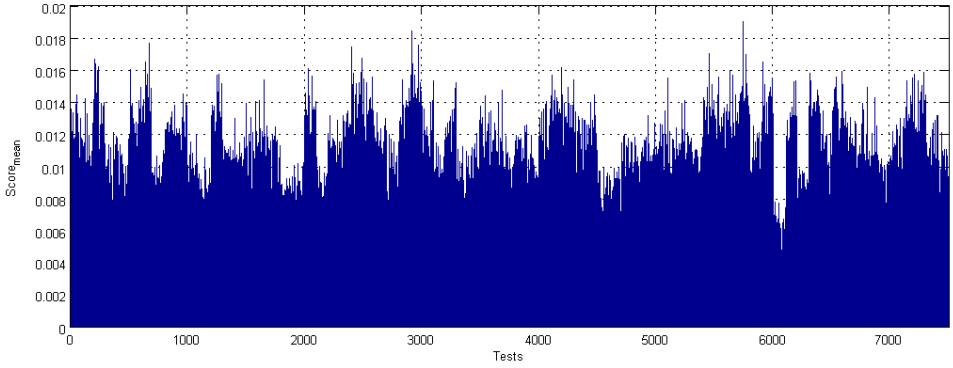


Figure 2.9 – *Valeurs de* $score_{mean}$ *pour des images non tatouées*

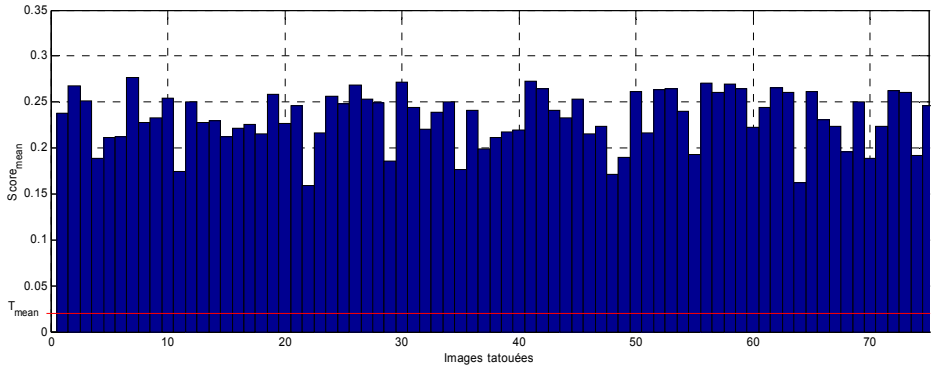


Figure 2.10 – *Valeurs de* $score_{mean}$ *pour des images tatouées (sans attaque)*

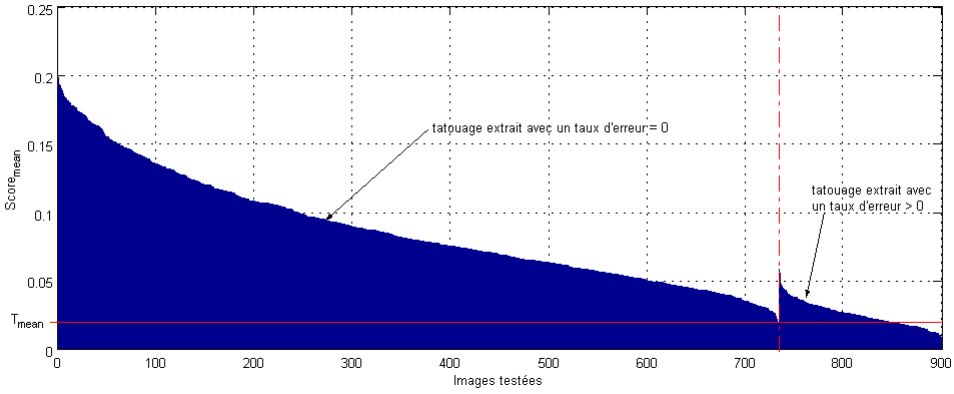


Figure 2.11 – *Valeurs de* $score_{mean}$ *rangées par ordre décroissant pour des images attaquées*

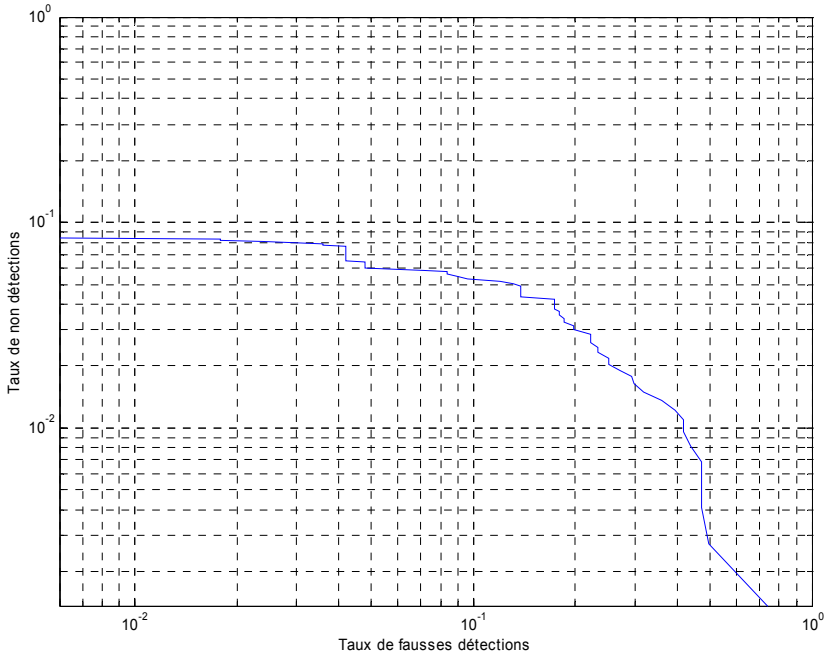


Figure 2.12 – *Courbe ROC*

Détermination de τ_{min}

La valeur de τ_{min} est plus complexe à déterminer que celle de τ_{mean}. Nous devons en effet calculer le critère $score_{min}$ dans des configurations où le tatouage est en limite de robustesse. Le protocole expérimental est donc le suivant. Il se décompose en trois grandes étapes. La première étape consiste à tatouer une base d'images, puis à faire subir, à chacune d'entre elles, des attaques de plus en plus fortes. Pour notre expérience, nous avons choisi d'utiliser différents taux de compression Jpeg. La deuxième partie du test revient alors à calculer la valeur de $score_{min}$ ainsi que le taux d'erreur par message pour chaque image testée.

Avant d'aller plus en avant dans l'analyse des résultats, il convient de définir les différents types d'erreurs possibles lors de la détection de la marque :

- Fausses détections (false positive errors) : il s'agit du cas où le détecteur indique la présence d'une marque dans une image non tatouée ou bien qu'il ne détecte pas les erreur dans la marque extraite.

- Non détections (false négative errors) : ce type d'erreur survient lorsque le détecteur ne détecte pas la marque présente dans l'image ou qu'il considère à tort que celle-ci est erronée.

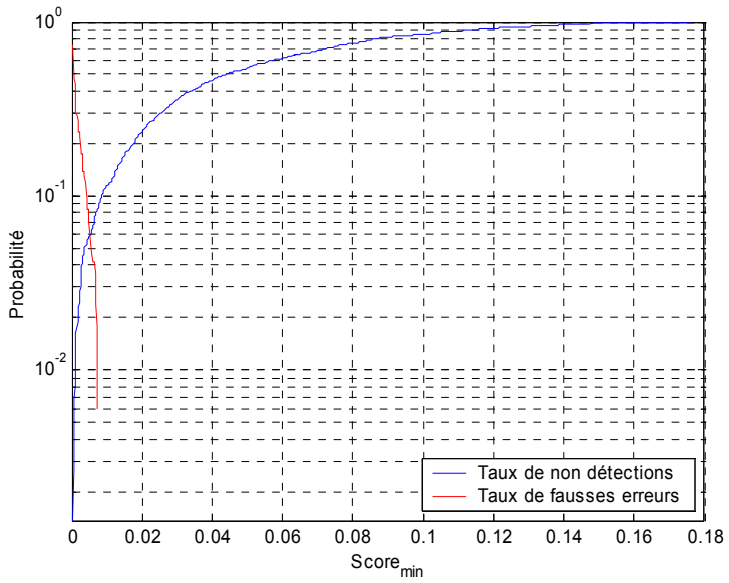


Figure 2.13 – *Taux de non détections et taux de fausses erreurs en fonction du* $score_{\min}$

Comme dans tout système de tatouage, nous devons déterminer un compromis entre le taux de fausses détections et celui de non détections. Typiquement, lorsque l'on augmente le seuil τ_{min}, le taux de fausses détections diminue, alors que le taux de non détections augmente. Pour cela nous allons représenter les résultats sous la forme d'une courbe ROC (Receiver Operating Characteristic). Une courbe ROC est une courbe paramétrique qui reporte le taux de fausses détections (en abscisse) par rapport au taux de non détections (en ordonnée) en fonction du seuil τ_{mim} choisi. Dans le cas d'une application de type protection des droits d'auteur, le message doit être extrait sans erreur, nous choisirons alors une valeur de τ_{mim} de telle sorte que le taux de fausses détections soit le plus faible possible. La valeur expérimentale retenue est de 0.01.

2.4. Cas particulier des images couleur

Le tatouage d'une image couleur nécessite des traitements supplémentaires (à l'insertion comme à l'extraction). La marque étant insérée dans la composante de luminance, il est nécessaire d'opérer un changement d'espace colorimétrique, de *RGB* vers *YUV* (2.17) ou *YCrCb*. La composante *Y* une fois tatouée est ensuite recombinée avec les composantes *U* et *V* de l'image originale. Le triplet de composantes *YUV* est alors reconverti en *RGB* (2.18) afin de produire l'image tatouée.

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0{,}299 & 0{,}587 & 0{,}114 \\ -0{,}148 & -0{,}289 & 0{,}437 \\ 0{,}615 & -0{,}515 & -0{,}100 \end{bmatrix} \cdot \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{2.17}$$

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1{,}140 \\ 1 & -0{,}395 & -0{,}581 \\ 1 & 2{,}032 & 0 \end{bmatrix} . \begin{bmatrix} Y \\ U \\ V \end{bmatrix} \tag{2.18}$$

Le choix de tatouer la composante de luminance est principalement motivé par des raisons de robustesse, notamment vis-à-vis de la compression Jpeg qui préserve mieux la luminance que les composantes de chrominance. Néanmoins, il aurait été tout à fait envisageable d'insérer le tatouage dans un autre espace colorimétrique, comme par exemple dans la composante bleue afin de minimiser la distorsion visuelle. Bien évidemment dans ce cas, le compromis capacité/visibilité/robustesse s'en trouve changé.

Remarque (perte liée aux changements d'espaces colorimétriques) : compte tenu de la structure de données utilisée pour représenter les images numériques (tableaux d'entiers) et des règles d'arrondi, il convient de souligner que les changements d'espaces colorimétriques entraînent une dégradation de la qualité de l'image. Cependant, cette perte reste négligeable au regard de celle engendrée par le processus de tatouage proprement dit.

3. Performances de l'algorithme de base

3.1. Evaluation de la distorsion visuelle

L'impact visuel du tatouage dans notre algorithme dépend principalement du seuil δ_h choisi. Plus la valeur de δ_h sera grande et plus la dégradation de l'image sera importante. D'un point de vue subjectif, malgré l'absence de masque psychovisuel, l'amplitude du tatouage varie néanmoins en fonction du contenu fréquentiel de l'image originale. Ceci est directement lié au calcul IFS de l'attracteur. En effet, dans un contexte de codage par auto-similarités, il est beaucoup plus simple de coder des zones unies que des zones de contours ou de textures. Cela se traduit au niveau de l'image d'erreur (*i.e.* du support du tatouage) par de faibles amplitudes dans les régions uniformes et des erreurs plus importantes dans les parties texturées. De ce fait, la distorsion induite par le tatouage reste faiblement perceptible.

D'un point de vue quantitatif, la distorsion visuelle introduite par le tatouage a été mesurée à l'aide de deux métriques : le PSNR et le wPSNR (ces métriques ont été décrites au paragraphe 5.1 du chapitre 1). La figure ci-dessous représente les valeurs du PSNR et du wPSNR en fonction de l'image tatouée. La dégradation visuelle moyenne est de l'ordre de 38,25 dB pour le PSNR et 51,8 dB pour le wPSNR, ce qui correspond à une compression JPEG de très bonne qualité (~Jpeg 80-90%).

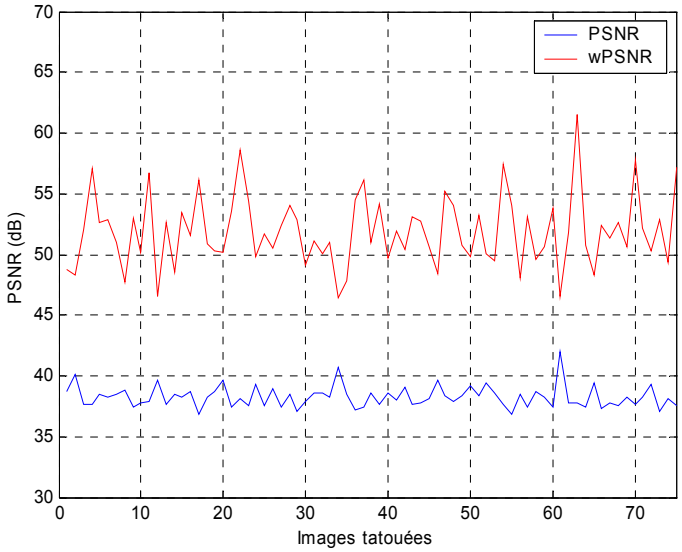


Figure 2.14 – *PSNR et du wPSNR en fonction des images*

3.2. Evaluation de la robustesse

Nous évaluons la robustesse du message en terme de taux d'erreur binaire moyen (bit error rate) et de taux d'erreur par message (message error rate). Le taux d'erreur binaire moyen correspond au rapport du nombre de bits erronés en moyenne sur le nombre total de bits du message. Ce rapport peut être considéré, sous certaines conditions, comme une bonne estimation de la probabilité d'erreur. Le taux d'erreur par message correspond, quant à lui, au rapport du nombre de messages erronés sur le nombre total de messages (*i.e.* dans notre contexte, le nombre total de messages correspond au nombre d'expériences réalisées avec les mêmes paramètres). Un message est considéré comme erroné si au moins un de ses bits est faux. Le taux d'erreur par message est une mesure de robustesse intéressante pour des applications où le message doit être extrait sans aucune erreur (*e.g.* identification de l'auteur d'une image).

3.2.1. Robustesse vis-à-vis de la compression Jpeg

Nous présentons dans ce paragraphe des résultats obtenus face à différents taux de compression Jpeg, pour différentes tailles de message (77, 121, 441 et 676 bits) avec des images de taille 512×512 pixels. Le premier graphique de la figure 2.15 représente le taux erreur binaire moyen et le second le taux d'erreur par message. On constate sans surprise que le taux d'erreur binaire et le taux d'erreur par message augmentent lorsque la taille du message s'allonge. Cela est tout à fait normal dans la mesure où la redondance de chaque bit est inversement proportionnelle à la longueur du message.

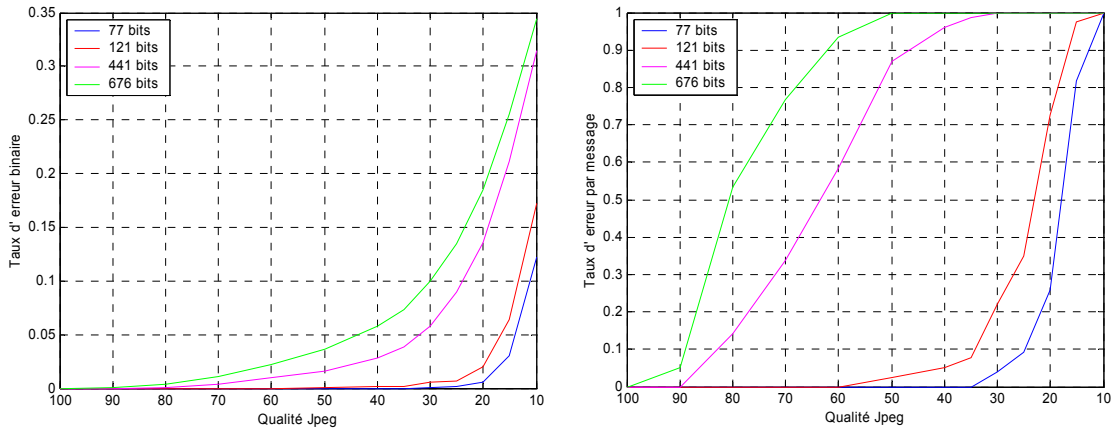


Figure 2.15 – *Robustesse vis-à-vis de la compression Jpeg*

3.2.2. Robustesse face à l'ajout d'un bruit gaussien

Nous présentons dans ce paragraphe des résultats similaires obtenus face à l'ajout d'un bruit gaussien $\mathcal{N}(0,\sigma)$, pour différentes tailles de message (77, 121, 441 et 676 bits) avec des images 512×512. Le premier graphique représente le taux erreur binaire moyen et le second le taux d'erreur moyen par message. Les performances de l'algorithme face à ce type d'attaque sont correctes, mais restent en pratique limitées par la méthode de modulation (et de démodulation) utilisée pour insérer (et extraire) le tatouage. En effet le bruit ajouté à l'image se retrouve en grande partie au niveau du support extrait, biaisant ainsi la phase de séparation des signaux décrites au paragraphe 2.2.1.

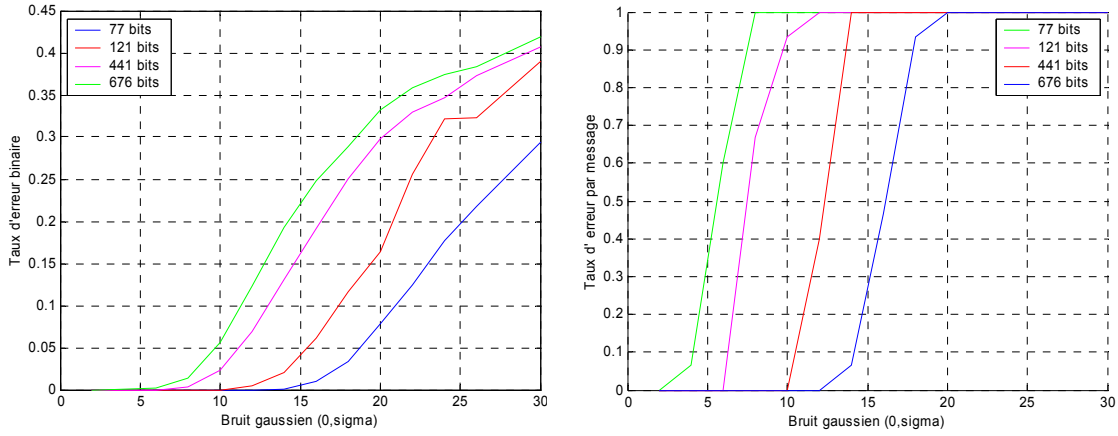


Figure 2.16 – *Robustesse face à l'ajout d'un bruit gaussien*

3.2.3. Attaques photométriques diverses

Nous présentons également quelques résultats significatifs face à des attaques photométriques courantes telles que la quantification des couleurs (typiquement conversion au format gif), l'application de filtres passe-bas, passe-haut et médian, le « print and scan » (sans désynchronisation), ainsi que quelques filtres spéciaux. La figure 2.17 illustre quelques unes des manipulations photométriques testées. Les résultats obtenus sont synthétisés dans le tableau ci-dessous. Tous les tests ont été réalisés sur une base de 75 images, hormis ceux concernant les attaques d'impression qui n'ont été effectués que pour quelques images avec un « payload » de 64 bits.

Attaques	77 bits		121 bits		441 bits		676 bits	
	ber	mer	ber	mer	ber	mer	ber	mer
moyenne 3x3	0,0008	0,04	0,004	0,267	0,055	0,987	0,088	1
moyenne 5x5	0,044	0,826	0,082	0,987	0,223	1	0,269	1
médian 3x3	0	0	0	0	0,004	0,48	0,014	0,92
médian 5x5	0,002	0,146	0,011	0,56	0,101	1	0,151	1
quantification 256c.	0	0	0	0	0	0	0,0004	0,173
quantification 16c.	0	0	0,07	0,76	0,187	0,973	0,369	1
modif. luminance	0	0	0	0	0	0	0	0
modif. contraste	0	0	0	0	0	0	0	0
correction gamma	0	0	0	0	0	0	0	0
réhaussement	0	0	0	0	0	0	0	0
Egalisation histo.	0	0	0	0	0	0	0,041	0,71
filtres couleur	0	0	0	0	0	0	0	0
effet de texture	0	0	0,0003	0,04	0,014	0,853	0,073	0,986
effet sépia	0	0	0	0	0	0	0	0
impression N&B	0	0	x	x	x	x	x	x
impression couleur	0	0	x	x	x	x	x	x

Tableau 2.1 – *Résultats de tests de robustesse faces à diverses manipulations photométriques*

L'algorithme de tatouage offre dans l'ensemble de bonnes performances en termes de robustesse face aux manipulations photométriques classiques. On constate sans surprise que le tatouage est très résistant aux modifications de luminance et de contraste, ainsi qu'aux ajustements des couleurs (*e.g.* balance des couleurs, effet sépia, etc.). Cette robustesse est due essentiellement aux propriétés d'invariance du support fractal face à ce type de manipulation. En ce qui concerne l'application de filtres passe-bas, les résultats en terme de taux d'erreur binaire sont relativement bons pour des capacités de 77 et 121 bits, par contre le taux d'erreur par message tend assez rapidement vers 1 lorsque l'on augmente la longueur du message ou la taille du filtre.

Remarque : l'application d'un filtre négatif à une image n'affecte pas l'amplitude des éléments du support, seuls les signes sont inversés. De ce fait, les bits du message extrait sont inversés. Les scores de vraisemblance utilisés ne permettent pas de lever en aveugle cette ambiguïté. Une solution simple consiste à utiliser les bits de resynchronisation (*c.f.* chapitre 4) ; leur valeur étant connue, il devient alors très facile de lever cette ambiguïté.

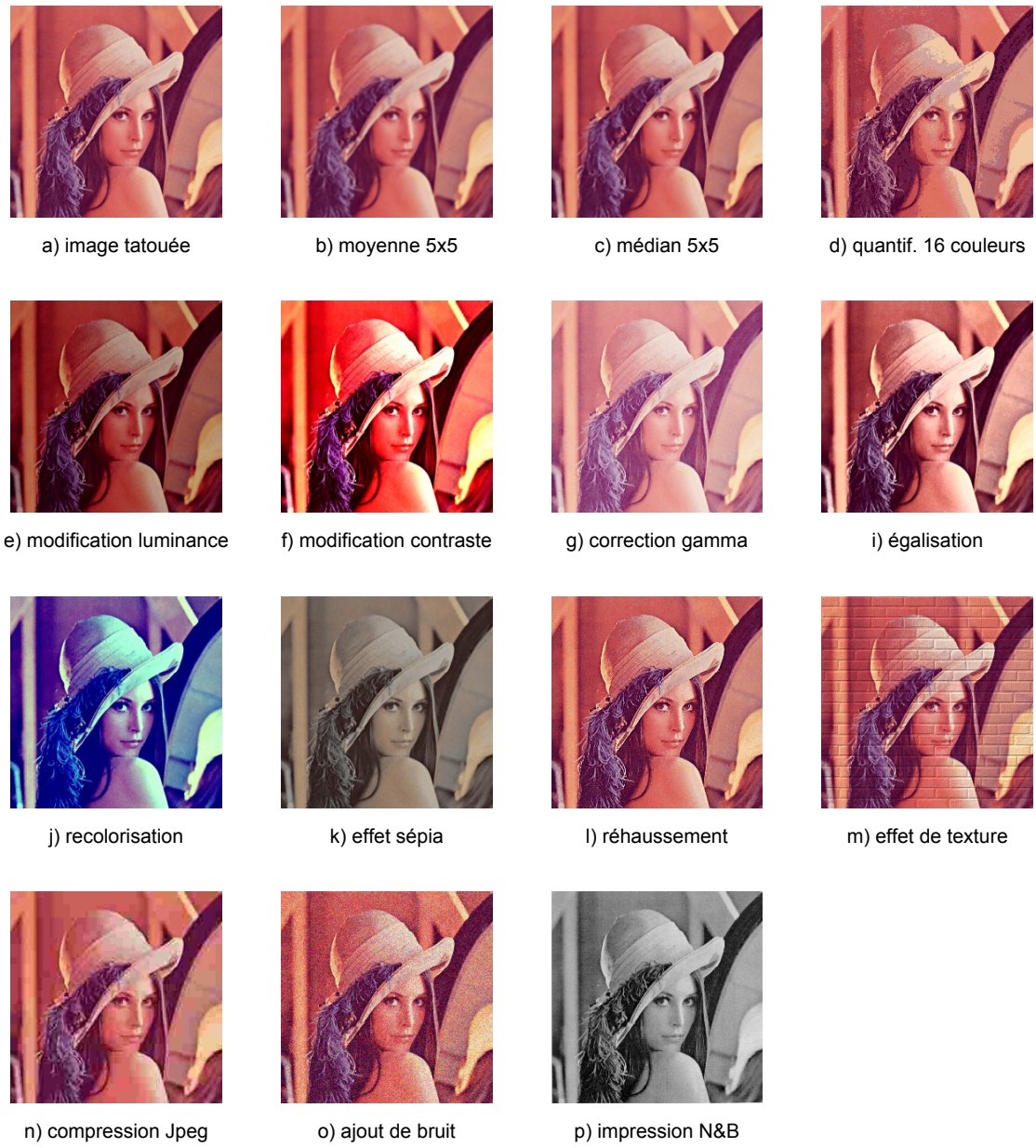


Figure 2.17 – *Exemples de manipulations photométriques courantes*

4. Conclusion

Nous avons présenté dans ce chapitre un algorithme de tatouage d'image original basé sur un modèle d'IFS. Cet algorithme, tel qu'il a été présenté ici dans sa version de base, offre déjà de bonnes performances en termes de robustesse face à des attaques de nature photométrique (*e.g.* compression avec pertes, filtrage, quantification, modification de luminance et de contraste, etc.). Ces

bons résultats sont liés d'une part à certaines propriétés d'invariance du support du tatouage, et d'autre part à la façon dont la marque a été mise en forme. Pour ce qui est des performances sur le plan visuel, notre technique de tatouage se situe dans la moyenne haute communément admise par la communauté « watermarking » ; la distorsion moyenne introduite par le marquage est de l'ordre de 38dB, ce qui correspond à une compression jpeg de très bonne qualité. Enfin, la complexité générale de l'algorithme reste tout à fait raisonnable ; l'insertion et l'extraction d'un message sont de l'ordre de la seconde sur un PC de type Pentium 4 1.7 Ghz.

Nous proposons néanmoins dans les chapitres suivants différentes optimisations afin d'accroître la robustesse du message sans que cela soit au détriment la visibilité. Dans le chapitre 3, nous préconisons d'utiliser, lors de la mise en forme du tatouage, des turbo codes en bloc en plus des codes par répétition afin d'améliorer les résultats principalement en termes de taux d'erreur par message. Et dans le chapitre 4, nous présentons deux méthodes de resynchronisation de manière à rendre le tatouage plus robuste aux déformations géométriques locales et globales de l'image. Actuellement, seul un décalage d'un pixel est toléré à l'extraction pour retrouver le message d'origine, ce qui est bien sûr insuffisant pour faire face aux manipulations courantes en traitement d'image.

Chapitre 3

Gain en robustesse par la mise en œuvre de turbo codes

1. Introduction au codage canal

Les codes correcteurs d'erreurs (Error Correcting Codes) ou le codage canal constituent un élément fondamental dans une chaîne de communication numérique. L'objectif de cette chaîne est de transmettre de manière fiable un message d'un émetteur (ou source), vers un destinataire, qui peut être éloigné dans le temps et/ou dans l'espace. Les codes visent à rajouter au message à transmettre des informations supplémentaires dans le but de détecter et/ou de corriger d'éventuelles erreurs survenues lors de la transmission. Les codes peuvent être classés en deux grandes catégories : les codes en blocs d'une part et les codes convolutifs d'autre part. Nous ne ferons dans ce chapitre qu'un bref rappel sur le principe de base des codes correcteurs, le lecteur non familier avec cette technique est invité à consulter les ouvrages suivants [GJ96, HW99].

1.1. Notion de message numérique

On définit communément un message numérique comme une suite d'éléments pouvant prendre une valeur parmi un ensemble Q appelé également alphabet. Les éléments, qui peuvent également être considérés comme des variables aléatoires discrètes, sont dits Q-aires. Dans le cas particulier où l'alphabet est constitué uniquement de deux éléments {0 et 1}, les éléments sont dits binaires. On remarquera que tout élément Q-aire peut être représenté par une suite d'éléments binaires. Dans le cadre du tatouage d'image, nous ne considèrerons que des messages formés d'éléments binaires.

1.2. Chaîne de transmission numérique

La figure 3.1 représente le schéma classique d'une chaîne de transmission numérique. Les principaux éléments sont, d'une part la source, le milieu de transmission et le destinataire, qui constituent les données du problème, et d'autre part le codage/décodage de source, le codage/décodage de canal, l'émetteur et le récepteur qui représentent les degrés de liberté du concepteur pour réaliser le système de transmission.

1.2.1. Codeur de source

Le codage de source vise à la concision maximale du message, afin de minimiser les ressources nécessaires à la transmission (*e.g.* temps, puissance, bande passante, mémoire, etc.). Ce codage a pour objectif de substituer un message aussi court que possible au message émis par la source, dans la mesure où cette opération est réversible (*i.e.* que le message initial peut être exactement restitué). Les limites du codage de source sont fixées par la théorie de l'information (premier théorème de Shannon). Au-delà de cette limite, le codage s'effectue avec perte, c'est-à-dire qu'à partir des données codées on n'est plus en mesure de restituer exactement le message d'origine.

1.2.2. Codeur de canal

Le codage de canal vise quant à lui à la protection du message contre les perturbations du canal de transmission. L'opération de codage canal consiste à ajouter au message à transmettre des éléments binaires, dits de redondance suivant une loi donnée. La nécessité d'introduire de la redondance dans le message, pour se prémunir des erreurs de transmission est démontrée par la théorie de l'information. Intuitivement, on peut concevoir que pour un message dépourvu de redondance, chaque élément binaire est unique et ainsi, toute erreur de transmission conduit à une perte d'information irréversible. Inversement, des éléments de redondance introduits astucieusement vont corréler les éléments binaires du message codé. Ainsi, sous certaines conditions, un ou plusieurs éléments binaires erronés au cours de la transmission pourront être détectés, voire même corrigés. L'introduction du codage canal conduit toujours à un accroissement de la taille du message. Il y a donc un antagonisme entre le codage de source et le codage de canal, l'objectif du premier étant de diminuer la redondance du message, celui du second d'en ajouter dans un but de protection.

1.2.3. Canal de transmission

Le canal de transmission inclut tous les éléments situés entre la sortie du codeur de canal et l'entrée du décodeur de canal, soit : l'émetteur, le milieu de transmission, le bruit et le récepteur. En considérant des messages numériques constitués d'éléments binaires, l'entrée du canal de transmis-

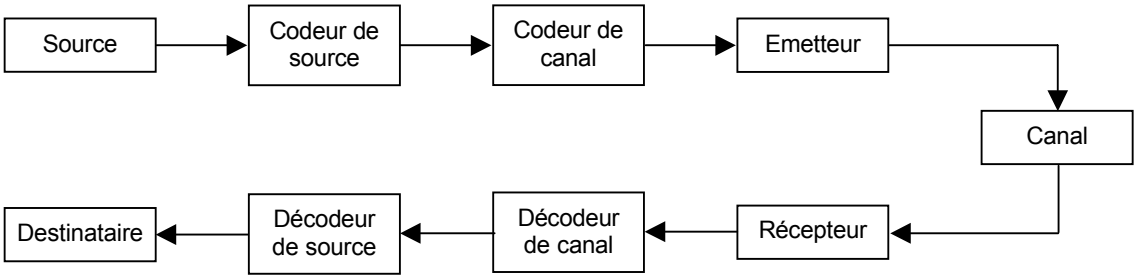


Figure 3.1 – *Chaîne classique de transmission numérique*

sion est discrète et binaire, mais sa sortie peut être discrète ou continue.

- Elle est discrète si le récepteur a pris une décision ferme (*hard decision*), c'est-à-dire s'il fournit au décodeur une suite d'éléments binaires représentative du message numérique codé. L'utilisation d'une décision ferme dans le récepteur conduit à une perte irréversible d'information pour le décodeur, mais autorise des algorithmes de décodage de mise en œuvre relativement simple, car travaillant à partir de données binaires.

- Elle est continue si le récepteur n'a pas pris de décision ferme, mais uniquement une décision dite souple (*soft decision*), c'est-à-dire s'il fournit au décodeur une suite d'échantillons analogiques prélevés généralement aux instants kT en sortie du filtre adapté. L'utilisation de décisions pondérées conduit à de meilleures performances du décodeur que l'utilisation de décisions fermes. Mais en contrepartie, cet avantage se paie généralement par un accroissement de la complexité des algorithmes de décodage.

Selon la structure discrète ou continue de la sortie du canal de transmission, on définit deux principaux modèles de canal de transmission : le canal discret et le canal à Bruit Additif Blanc Gaussien (AWGN).

Canal discret (BSC)

Un canal discret possède un alphabet d'entrée $[x_0,\ldots,x_i,\ldots,x_{n-1}]$ et un alphabet de sortie $[y_0,\ldots,y_j,\ldots,y_{m-1}]$ finis. Il est caractérisé par ses probabilités de transition p_{ij}^k définies de la manière suivante :

$$p_{ij}^k = P\left\{ Y_k = y_i \middle| X_k = x_i \right\} \quad \text{avec} \quad \sum_{j=0}^{m-1} p_{ij}^k = 1 \quad \forall i \tag{3.1}$$

où X_k et Y_k représentent respectivement les éléments discrets à l'entrée et à la sortie du canal à l'instant kT.

Lorsque les probabilités de transition sont indépendantes du temps, le canal est dit stationnaire. Il est de plus sans mémoire si l'élément Y_k ne dépend que de l'élément X_k. Un modèle simple de canal discret, stationnaire et sans mémoire est le canal binaire symétrique pour lequel les éléments X_k et Y_k sont binaires, à valeur dans l'alphabet {0,1} et dont les probabilités de transition sont symétriques. Classiquement, le canal BSC est représenté symboliquement par un diagramme en treillis à deux états (*c.f.* Figure 3.2).

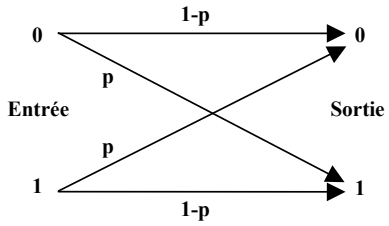


Figure 3.2 – *Représentation d'un canal binaire symétrique*

Canal à bruit additif blanc gaussien (AWGN)

Pour ce canal l'entrée X_k est discrète, généralement binaire, à valeur dans l'alphabet {0,1} et la sortie est continue, constituée d'échantillons analogiques perturbés par un bruit discret b_k, additif, blanc, gaussien, stationnaire, centré et indépendant des éléments X_k. Le canal à bruit additif blanc gaussien peut être représenté symboliquement par le diagramme de la figure 3.3

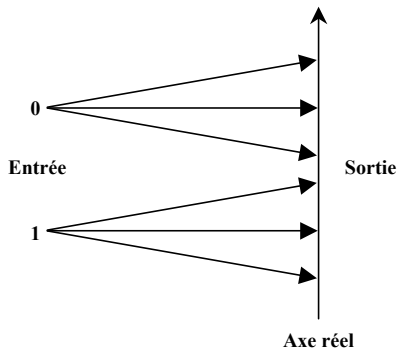


Figure 3.3 – *Représentation d'un canal à bruit additif blanc gaussien*

1.2.4. Décodeur de canal

Plusieurs stratégies différentes peuvent être utilisées par le décodeur de canal :

- *Détection d'erreurs* – Le décodeur observe la séquence reçue (ferme ou souple) et détecte la présence éventuelle d'erreurs. Cette détection peut servir à contrôler le taux d'erreur ou à mettre en œuvre des techniques de retransmission (le décodeur demande à l'émetteur de retransmettre la séquence dans laquelle une erreur a été détectée).

– *Correction d'erreurs* – Cette opération nécessite des algorithmes beaucoup plus complexes que la simple détection, et plus de redondance dans la séquence émise. Le décodeur observe la séquence reçue, détecte et corrige (si cela est possible) les éventuelles erreurs.

1.3. Code à répétition

Il s'agit d'un moyen intuitif pour protéger un message, puisqu'il consiste à répéter n fois chacun de ses éléments binaires. Ainsi, à chaque bit issu de la source, le code associe un mot de code de taille n. Le rendement R d'un tel code est de $1/n$. Le rendement caractérise la redondance introduite par le code de canal. Le décodage est quant à lui très simple, puisqu'il consiste à prendre en sortie du canal une décision majoritaire. Prenons l'exemple où n est égale à 3. Le procédé de décodage majoritaire produira une erreur de décision si deux ou trois parmi les 3 bits représentant un mot de code ont subi une inversion lors de leur transmission sur le canal BSC. Le code à répétition est un code rudimentaire, et des résultats comparables peuvent être aisément obtenus au prix d'une redondance moindre (*i.e.* un rendement plus élevé) en utilisant des codes plus élaborés.

1.4. Codes en blocs linéaires

1.4.1. Définition

Le codage en blocs consiste à associer à chaque bloc de k éléments binaires issu de la source, un bloc de n éléments binaire (avec $n > k$ pour satisfaire à la condition de redondance). Les blocs de n éléments sont appelés les mots du code. Le rapport k/n représente le rendement du code (également appelé le taux de codage) et la différence $(n - k)$ le nombre d'éléments binaires de redondance introduit par le code. Pour un code en blocs linéaire $C(n,k)$, chaque mot du code $c = [c_0 \ldots c_j \ldots c_{n-1}]$ est associé à un bloc d'information $m = [m_0 \ldots m_j \ldots m_{k-1}]$ par la relation matricielle $c = mG$, G étant une matrice à k lignes et à n colonnes appelée matrice génératrice du code. La matrice génératrice d'un code en blocs n'étant pas unique, il est toujours possible d'écrire cette matrice sous la forme suivante :

$$G = \left[I_k , P \right] \tag{3.2}$$

où I_k est la matrice identité $k \times k$ et P une matrice $k \times (n-k)$ utilisée pour calculer les $(n-k)$ éléments binaires de redondance. Ainsi écrite, la matrice génératrice G est sous la forme réduite et le code est dit systématique.

A un code en blocs linéaire $C(n,k)$, on peut toujours associer une matrice H, appelée matrice de contrôle de parité, orthogonale aux mot du code.

Pour un code en blocs linéaire systématique, la matrice de contrôle de parité est de la forme :

$$H = \left[P^T, I_{n-k}\right] \tag{3.3}$$

1.4.2. Principe de la détection et de la correction des erreurs

Nous considérons que le canal de transmission est du type binaire symétrique et que le décodeur fonctionne à partir de décisions fermes et sans désynchronisation avec le codeur. La détection des erreurs de transmission se fait en utilisant la propriété d'orthogonalité de la matrice de contrôle de parité avec les mots du code et en calculant le syndrome *s*, vecteur ligne à (*n-k*) composantes défini par :

$$s = rH^T = (c+e)H^T = eH^T \tag{3.4}$$

où *r* représente le mot reçu par le décodeur et *e* le vecteur d'erreur.

Le syndrome *s* est nul si, et seulement si, *r* est un mot du code. Un syndrome non nul implique la présence d'erreurs de transmission. Toutefois, un syndrome nul ne signifie pas nécessairement l'absence d'erreurs de transmission. En effet, le mot *r* peut appartenir aux mots du code tout en étant différent de *c*. La correction des erreurs se fait en recherchant le mot du code *c* le plus vraisemblable, c'est-à-dire celui qui est à la distance de Hamming minimale du mot reçu *r*.

1.4.3. Pouvoir de détection et de correction d'un code en blocs

La distance minimale $d_{\min}$ d'un code en blocs *C* est la distance de Hamming entre les deux mots de code les plus proches :

$$d_{\min} = \min\left\{d_H\left(c_i, c_j\right), \forall c_i, c_j \in C, i \neq j\right\} \tag{3.5}$$

Un code en blocs linéaire de distance minimale $d_{\min}$, peut détecter toutes les configurations de ($d_{\min}$-1) erreurs dans un bloc de n éléments binaires et corriger toutes les configurations de [($d_{\min}$-1)/2] erreurs.

1.5. Codes cycliques

1.5.1. Définition

Les codes cycliques représentent la classe la plus importante des codes en blocs linéaires. Pour un code cyclique, toute permutation circulaire à gauche de *j* éléments binaires d'un mot du code,

redonne un mot du code. Pour ces codes, on utilise généralement une représentation polynomiale des mots du code plutôt qu'une représentation vectorielle. Ainsi au mot c on associe le polynôme $c(x)$ de degré n-1.

$$c(x) = c_0 + c_1 x + ... + c_j x^j + ... c_{n-1} x^{n-1} \qquad c_j \in \{0,1\} \tag{3.6}$$

Pour un code cyclique les mots $c(x)$ sont des multiples d'un polynôme générateur $g(x) = g_0 + g_1 x + ... + g_j x^j + ... x^{n-k}$ diviseur de $(x^n + 1)$.

$$g(x)h(x) = (x^n + 1) \tag{3.7}$$

Le polynôme $h(x)$ de degré k est appelé le polynôme de contrôle de parité du code. Lorsque le code est sous forme systématique, le mot $c(x)$ associé au polynôme d'information $m(x) = m_0 + m_1 x + ... + m_j x^j + ... + m_{k-1} x^{k-1}$ est de la forme :

$$c(x) = v(x) + x^{n-k} m(x) \tag{3.8}$$

où $v(x)$ est le reste de la division de $x^{n-k}m(x)$ par le polynôme générateur $g(x)$.

1.5.2. Codes BCH

Les codes de Bose, Chaudhuri et Hocquenghem [BR60], communément appelés codes BCH, apparus en 1960, représentent aujourd'hui la famille la plus importante des codes cycliques. Pour tous entiers m et t, on peut construire un code BCH ayant pour paramètres :

$$n = 2^m - 1 \;\; ; \;\; d_{\min} \geq 2t + 1 \;\; ; \;\; k \geq 2^m - 1 - mt \tag{3.9}$$

n	k	t	$g(x)$
7	4	1	$1+x+x^3$
15	11	1	$1+x+x^4$
	7	2	$1+x+x^4+x^6+x^7+x^8$
	5	3	$1+x+x^2+x^4+x^5+x^8+x^{10}$
31	26	1	$1+x^2+x^5$
	21	2	$1+x^3+x^5+x^6+x^8+x^9+x^{10}$
	16	3	$1+x+x^2+x^3+x^5+x^7+x^8+x^9+x^{10}+x^{11}+x^{15}$
	…	…	…

Tableau 3.1 – *Quelques paramètres des codes BCH*

1.5.3. Codes de Reed-Solomon

Les codes de Reed-Solomon [RS60] sont un cas particulier des codes BCH. Ce sont des codes BCH dont la longueur est égale à l'ordre multiplicatif de l'alphabet des symboles. Ils sont constitués d'éléments q-aires où q est une puissance de 2 ($q=2^m$). Chaque élément q-aire d'un code de Reed-Solomon peut donc être représenté par m éléments binaires.

$$n = q - 1 \quad ; \quad d_{\min} = 2t + 1 \quad ; \quad k = n - 2t \tag{3.10}$$

Un code de Reed-Solomon peut donc corriger t éléments q-aires dans un bloc de n éléments q-aires, ou si on utilise une représentation binaire des éléments q-aires, au mieux mt éléments binaires. D'une manière générale, les codes Reed-Solomon sont bien adaptés à la correction d'erreurs binaires par paquet (ou « *burst* »).

1.5.4. Décodage des codes cycliques

Pour les codes cycliques, on calcule d'abord la version polynomiale $s(x)$ du syndrome s, puis on suit la même procédure que pour le décodage des codes en blocs. Le syndrome $s(x)$ est obtenu en évaluant le reste de la division du mot reçu $i(x)$ par le polynôme générateur $g(x)$.

$$r(x) = g(x)q(x) + s(x) \tag{3.11}$$

1.6. Codes convolutifs

Les codes convolutifs (ou récurrents) ont été inventés en 1955 par Elias [Eli55]. Ils constituent une seconde famille de codes correcteurs d'erreurs au moins aussi importante que les codes en blocs. Pour les codes convolutifs, chaque bloc de n éléments binaires en sortie du codeur dépend non seulement du bloc de k éléments binaires présent à son entrée, mais aussi des m blocs présents précédemment. Les codes convolutifs introduisent par conséquent un effet mémoire d'ordre m. La quantité $(m+1)$ s'appelle la longueur de contrainte du code. Un codeur convolutif est constitué d'un registre à $(m+1).k$ étages qui mémorise les $(m+1)$ blocs de k éléments binaires d'information, d'une logique combinatoire qui calcule les blocs de n éléments binaires fournis par le codeur et d'un convertisseur parallèle série (*i.e.* multiplexeur). La quantité $R=k/n$ est appelée le rendement du code. Si les k éléments binaires d'information présents à l'entrée du codeur sont effectivement émis, c'est-à-dire se retrouvent explicitement dans le bloc de n éléments binaires en sortie du codeur, le code est dit systématique.

Le décodage des codes convolutifs consiste à rechercher la séquence binaire la plus vraisemblable. Généralement la recherche de cette séquence est effectuée à l'aide, soit de l'algorithme de Viterbi [Vit67], soit de l'algorithme itératif de Fano [Fan63].

1.7. Codes concaténés

Les codes concaténés ont été inventés par Elias en 1954 avec l'introduction des codes produits [Eli54]. Ce concept de concaténation de deux codes sera généralisé par Forney en 1966 [For66]. Il propose une structure cascadant en série un code externe suivi d'un code interne. L'intérêt majeur de la concaténation de deux codes réside dans l'obtention d'un code de distance minimale élevée, donc puissant, tout en maintenant une complexité de codage et surtout de décodage raisonnable. A titre d'exemple, un code produit construit à partir de deux codes de même pouvoir de correction t possède une distance minimale de $(2t+1)^2$ et une complexité de décodage en 2^t. Pour obtenir le même pouvoir de correction avec un code en bloc simple, il faudrait un décodeur algébrique de complexité en $2^{2t(t+1)}$.

On distingue généralement deux types de concaténations :

- *Série* : le message est d'abord codé par le code externe, puis le mot de code résultant constitue l'entrée du codeur interne. Le rendement du code concaténé est égal au produit des rendements des deux codes élémentaires.

- *Parallèle* : le message est codé pat le code C_1 donnant une séquence c_1 et parallèlement, après un entrelacement facultatif, il est codé par un code C_2 produisant une séquence c_2. Le mot de code résultant est le couple (c_1, c_2) et le rendement est égale à $\frac{R_1.R_2}{R_1+R_2}$.

2. Les codes correcteurs en tatouage d'image

Jusqu'à présent, la majorité des algorithmes de tatouage d'image utilisent, afin de garantir un minimum de robustesse, une redondance de l'information basée sur une simple répétition de chaque bit du message. Ce codage est très simple à mettre en œuvre, mais il n'a pas un pouvoir de correction d'erreur très efficace dans la mesure ou le gain de codage est de zéro dB. Il semble cependant évident qu'il pourrait être plus avantageux d'utiliser des codes correcteurs plus élaborés qu'une simple duplication. En effet, cette approche apparaît naturelle si l'on compare le problème du tatouage d'image à celui de la transmission d'un signal à travers un canal bruité. Ce genre de modèle considère généralement l'image comme étant le canal de transmission et les différentes attaques comme l'ajout d'un bruit sur ce canal. Les codes correcteurs d'erreurs sont très largement utilisés en codage canal et peuvent se révéler très utile en tatouage d'image. Ainsi, on commence à trouver dans la littérature de récentes publications faisant état de l'utilisation de codes correcteurs beaucoup plus puissants, comme des codes convolutifs [FDM00], BCH [TP00, DSM00, BDS+01] ou la concaténation d'un code convolutif et d'un code Reed-Solomon [BQR01]. L'utilisation de turbo codes convolutifs est également mentionnée par certains auteurs [KMKM00, PVP00, BPS01]. Dans

le contexte d'un attaque Jpeg, la plupart des expériences rapportées montrent que pour de très faibles qualités de compression (*i.e.* en dessous de 20%), les codes par répétitions semblent être la meilleure solution. Néanmoins, pour de telles valeurs, la qualité de l'image compressée la rend totalement inexploitable d'un point de vue commercial et par conséquent le problème de la robustesse du tatouage n'est plus réellement indispensable. Par ailleurs, lorsque l'on compare des codes BCH avec des codes convolutifs, en considérant bien évidemment une taille de message et un rendement identiques, les résultats donnés par [BDS+01] montrent que pour des qualités de compressions inférieures à 50%, les codes BCH sont meilleurs, alors que pour des qualités supérieures les résultats sont inversés. Dans la plupart des articles traitant des codes BCH, les codes utilisés n'ont un pouvoir de correction que de 1 ou 2 et un rendement de l'ordre de 0.7. En effet plus la capacité de correction d'un code est grande, plus la complexité de son décodage est élevée, limitant par conséquent les performances de ces codes. Dans ce contexte les codes produits (i.e. codes concaténés en série) constituent une alternative attractive pour concilier un pouvoir de correction des erreurs élevé et une complexité de décodage raisonnable.

3. Mise en œuvre des codes produits dans notre algorithme

Dans le cadre de notre algorithme de tatouage, nous avons opté pour l'utilisation de turbo codes en bloc qui sont reconnus pour leurs excellentes performances. Nos travaux ont été menés en étroite collaboration avec le département Signal et Communication de l'ESNT Brest, dans le cadre du projet GET – « Turbo Watermark ».

3.1. Modèle de transmission considéré

La problématique de notre algorithme de tatouage, tel que nous l'avons décrit au chapitre 2, peut être formulée d'un point de vue codage canal à l'aide du modèle de transmission représenté sur la figure 3.4. Dans ce modèle, les éléments binaires w_{kl} sont modulés par une amplitude aléatoire positive $y_{kl}=|\Delta_{kl}|$ (*i.e.* points du support), avant de traverser un commutateur à deux états (passant ou bloquant), noté s_{kl}, indépendant de w_{kl}. L'attaque aléatoire est modélisée par un bruit blanc additif gaussien. Les hypothèses sont les suivantes. L'entrée du canal, notée w, prend ses valeurs dans {-1,1} de manière équiprobable. La variable aléatoire s vaut 0 ou 1 avec une probabilité de un demi. La variable aléatoire y est la valeur absolue d'une variable aléatoire gaussienne généralisée (comme nous le montrerons au paragraphe suivant, centrée, de variance μ^2. Le bruit blanc additif gaussien b de moyenne nulle et de variance σ^2 est ajouté à $w.s.y$, pour former l'observation r :

$$r = w.s.y + b \tag{3.12}$$

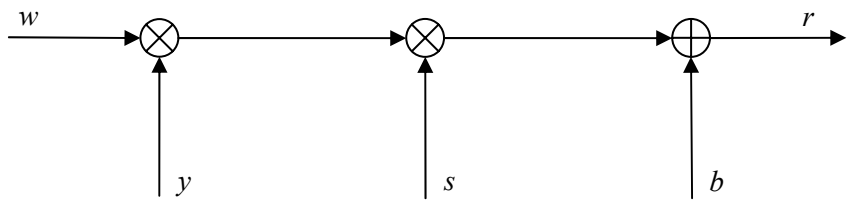


Figure 3.4 – *Modèle de transmission considéré*

En d'autres mots : w représente le signal du tatouage mis en forme, y les amplitudes du supports, s modélise le fait que statistiquement seulement un point sur deux du support est tatoué (dans la pratique cette statistique est faussée par le fait que certains échantillons de y ne peuvent être modulés avec le tatouage w pour des raisons de visibilité) et enfin le bruit gaussien b permet de modéliser d'une manière générale les attaques sur le canal de tatouage.

3.1.1. Etude du support du tatouage

Nous allons nous intéresser, dans un premier temps, aux propriétés du support du tatouage. Ainsi, quatre fonctions de densité de probabilité ont été envisagées pour modéliser la loi de distribution des éléments du support :

- *Loi gaussienne* $\mathrm{G}(\mu, \sigma)$;
- *Loi laplacienne* $\mathrm{L}(\mu, \sigma)$;
- *Loi gaussienne généralisée* $\mathrm{GG}(\mu, \sigma, \upsilon)$;
- *Mixture de deux gaussiennes* $\mathrm{GM}(\alpha_1, \mu_1, \sigma_1, \alpha_2, \mu_2, \sigma_2)$;

La loi gaussienne généralisée est donnée par la formule suivante :

$$GG_{\upsilon,\mu,\sigma}(x) = \frac{\upsilon.\alpha(\upsilon)}{2\sigma.\Gamma(1/\upsilon)} \exp\left\{-\left[\alpha(\upsilon)\left|\frac{x-\mu}{\sigma}\right|^{\upsilon}\right]\right\} \qquad \text{avec } \alpha(\upsilon) = \sqrt{\frac{\Gamma(3/\upsilon)}{\Gamma(1/\upsilon)}} \tag{3.13}$$

où $\Gamma(.)$ est la fonction gamma. Cette loi est caractérisée par sa moyenne μ, son écart type σ et son paramètre de forme υ. Il faut noter que si le paramètre υ est égal à 2, on obtient une loi gaussienne, et s'il est égal à 1, on a alors une loi laplacienne. Les paramètres υ et σ sont estimés à l'aide d'un critère ML [BBPR98].

Nous avons utilisé un test de maximum de vraisemblance (3.14) pour déterminer de manière objective la loi qui modélise au mieux les observations.

$$L = \sum_{o} \log\left(\mathrm{p}\left(\frac{o}{\theta}\right)\right) \tag{3.14}$$

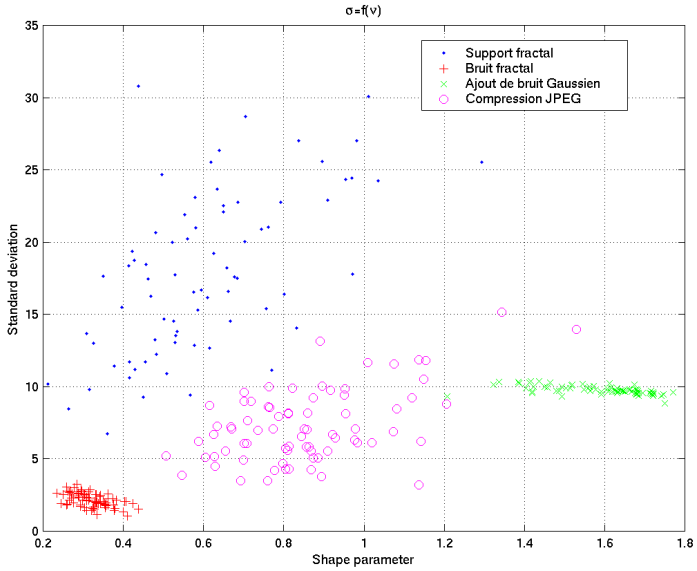


Figure 3.5 – *Ecart type en fonction du paramètre de forme*

où p est la densité de probabilité estimée, o les observations et les paramètres θ du modèle. Plus la valeur de L est élevée, plus la modélisation colle aux données réelles.

Les expériences réalisées sur une base de 75 images montrent que les éléments du support suivent une loi gaussienne généralisée. Nous avons estimé les paramètres de ce modèle avec les supports obtenus à partir des différentes images. Les résultats tendent à montrer que le support a une distribution à moyenne nulle, mais en revanche les paramètres (υ, σ) varient fortement en fonction de l'image considérée. D'autre part, ces deux paramètres ne semblent pas corrélés comme l'atteste la figure 3.5. Le paramètre de forme varie entre 0,2126 et 1,2930 et a une moyenne de 0,6035. On est donc assez éloigné d'une loi laplacienne. L'écart type varie entre 6,7628 et 30,7956 et a une valeur moyenne de 17,8832.

Remarque : il faut noter que nous avons cherché à modéliser la distribution du support fractal à l'aide d'une loi stationnaire (*i.e.* tous les éléments du support suivent la même loi quelle que soit leur position dans l'image). Néanmoins, lorsque l'on visualise le support, on se rend compte que de nombreuses disparités locales remettent en cause cette hypothèse.

3.1.2. Etude du bruit fractal

Comme nous l'avons évoqué au chapitre précédent, le calcul de l'attracteur est relativement stable vis-à-vis de l'insertion du tatouage, mais il n'est pas totalement invariant. Ces instabilités se répercutent au niveau du support du tatouage et peuvent être modélisées par l'ajout d'un bruit. Ce

bruit fractal correspond à la différence entre le support modulé I_{mod} et le support I_{supp} recalculé à partir de l'image tatouée (*i.e.* sans attaque). Expérimentalement, nous avons pu montrer que ce bruit peut être modélisé par une loi gaussienne généralisée de moyenne nulle et de paramètres υ=0,3215 et σ=2,1617. Néanmoins, on doit rester prudent vis-à-vis de cette modélisation. La figure 3.5 montre en effet que le couple de paramètres (υ, σ) obtenus pour chaque image ne forme pas un nuage de points concentrés autour de la valeur (0,3515 ; 2,1617).

3.1.3. Modélisation d'attaques photométriques sur le canal de transmission

Dans ce paragraphe nous étudions l'influence de la compression Jpeg, ainsi que celui de l'ajout d'un bruit gaussien sur le canal de tatouage. L'objectif est d'essayer de modéliser ces attaques d'un point de vue du codage canal. Nous nous intéressons dans un premier temps à l'impact de l'ajout d'un bruit gaussien sur la stabilité du codage IFS. En faisant l'hypothèse que l'attracteur reste stable, on s'attend normalement à retrouver la majorité du bruit au niveau du support. On ajoute donc un bruit gaussien de moyenne nulle et d'écart type σ sur le canal de luminance de l'image, et on estime ensuite le bruit résultant D_{gauss} (*c.f.* équation 3.14) au niveau du support de l'image bruitée $\widetilde{I}_{supp}$.

$$D_{gauss} = \widetilde{I}_{supp} - I_{supp} \tag{3.15}$$

Expérimentalement, on retrouve quasiment tout le bruit ajouté à l'image au niveau du support. Le signal de différence D_{gauss} est à moyenne nulle et a un écart type proche de celui du bruit qui a été ajouté. Cependant le paramètre de forme υ du modèle GG varie suivant les images (*e.g.* pour un σ de 10, υ varie entre 1,2070 et 1,17715 et a une valeur moyenne de 1,5848). On est donc assez loin de la valeur 2 caractérisant une gaussienne. Cet écrasement de la distribution vers 0 est liée en partie au codage IFS qui tente de minimiser les erreurs lors du calcul de l'attracteur.

En procédant de manière similaire, on observe que les perturbations introduites par la compression Jpeg sur le canal de tatouage suivent un modèle GG dont les paramètres υ, σ varient très fortement d'une image à l'autre.

3.2. Principe des codes produits

3.2.1. Codage

Les codes produits sont obtenus par une concaténation en série de deux codes en bloc réalisée de la manière suivante. Les bits d'information sont disposés dans une matrice $k_l \times k_c$. Toutes les lignes de cette matrice sont codées en utilisant un code en bloc linéaire $C_l(n_l, k_l)$ donnant une matrice

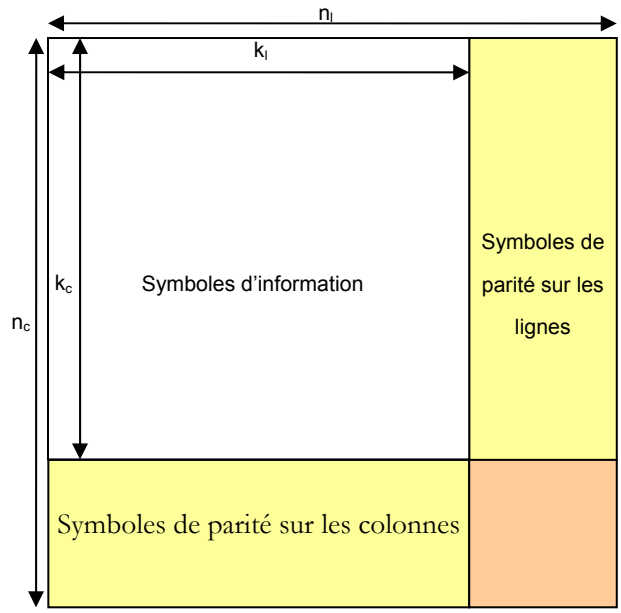


Figure 3.6 – *Codage du code produit*

$n_l \times k_c$, puis toutes les colonnes de cette nouvelle matrice sont codées par un deuxième code en bloc linéaire $C_c(n_c, k_c)$, engendrant alors un mot du code produit, une matrice $n_l \times n_c$. Le rendement R est égal à $R_l \times R_c$. Le principe de codage nous permet donc d'obtenir des codes produits ayant une très grande distance de Hamming, et ainsi de corriger un très grand nombre d'erreurs. Les codes produits possèdent des propriétés très intéressantes et en particulier le fait que toutes les lignes de la matrice codée sont des mots de code de C_l et que toutes les colonnes des mots de codes de C_c.

3.2.2. Décodage

Le décodage d'un code concaténé consiste à décoder l'un après l'autre les codes élémentaires le constituant. En utilisant des décodeurs à entrées et sorties pondérées, on peut itérer le processus de décodage. Dans notre système, nous utilisons la méthode proposée par R. Pyndiah en 1994 [PGPJ94, Pyn98] pour le décodage des codes produits. Le décodeur élémentaire est un décodeur de Chase [Cha72] qui fonctionne selon le maximum de vraisemblance. Il fournit le mot de code situé à distance euclidienne minimale du mot reçu. Le turbo décodeur est alimenté par des entrées pondérées et fournit à son tour en sortie des valeurs pondérées. Ses valeurs pondérées sont extraites du logarithme de rapport de vraisemblance, calculé pour chacun des bits. Notons r l'observation et c un mot de code quelconque. Dans ce cas, le rapport de vraisemblance correspondant au bit c_k vaut :

$$\Gamma(c_k) = \ln\left(\frac{\Pr(c_k = 1|r)}{\Pr(c_k = -1|r)}\right) = \ln\left(\frac{p(c_k = 1, r)}{p(c_k = -1, r)}\right) = \ln\left(\frac{\sum_{c\in C, c_k=1} p(r|c)}{\sum_{c\in C, c_k=-1} p(r|c)}\right) \tag{3.16}$$

Dans l'hypothèse où $r = \alpha.c + b$ et que α est connu du récepteur, on montre que ce rapport s'écrit alors :

$$\Gamma(c_k) = \ln\left(\frac{\sum_{c\in C, c_k=1} \exp\left(-\frac{\|r-\alpha c\|^2}{2\sigma^2}\right)}{\sum_{c\in C, c_k=-1} \exp\left(-\frac{\|r-\alpha c\|^2}{2\sigma^2}\right)}\right) \tag{3.17}$$

L'approximation proposée par R. Pyndiah consiste à ne considérer que deux mots de code parmi l'ensemble des mots de code. La valeur approchée du logarithme de rapport de vraisemblance est alors donnée par :

$$\hat{\Gamma}(c_k) = \frac{1}{2\sigma^2}\left(\min_{c\in C, c_k=-1}\|r-\alpha c\|^2 - \min_{c\in C, c_k=1}\|r-\alpha c\|^2\right) \tag{3.18}$$

En développant l'équation (3.18), on obtient finalement :

$$\hat{\Gamma}(c_k) = \frac{1}{\sigma^2}\left(\max_{c\in C, c_k=1}\sum_l r_l\alpha_l c_l - \max_{c\in C, c_k=-1}\sum_l r_l\alpha_l c_l\right) \tag{3.19}$$

Lorsque l'atténuation a est inconnue du récepteur, ce qui est le cas ici, on peut utiliser la métrique approchée suivante :

$$\hat{\Gamma}(c_k) = \frac{1}{\sigma^2}\left(\max_{c\in C, c_k=1}\sum_l r_l c_l - \max_{c\in C, c_k=-1}\sum_l r_l c_l\right) \tag{3.20}$$

Nous verrons au paragraphe 3.5 que l'exploitation des répliques du code par répétition nous permettra de contourner ce problème.

3.3. Modification de l'étape d'insertion

La mise en place de codes correcteurs d'erreurs dans notre algorithme de tatouage se traduit simplement, au niveau du processus d'insertion, par une modification de l'étape de mise en forme du message (*c.f.* Figure 3.8). Désormais, après avoir converti le message à cacher sous une forme binaire, nous ajoutons à ce dernier des bits de redondance. Nous avons pour cela à notre disposi-

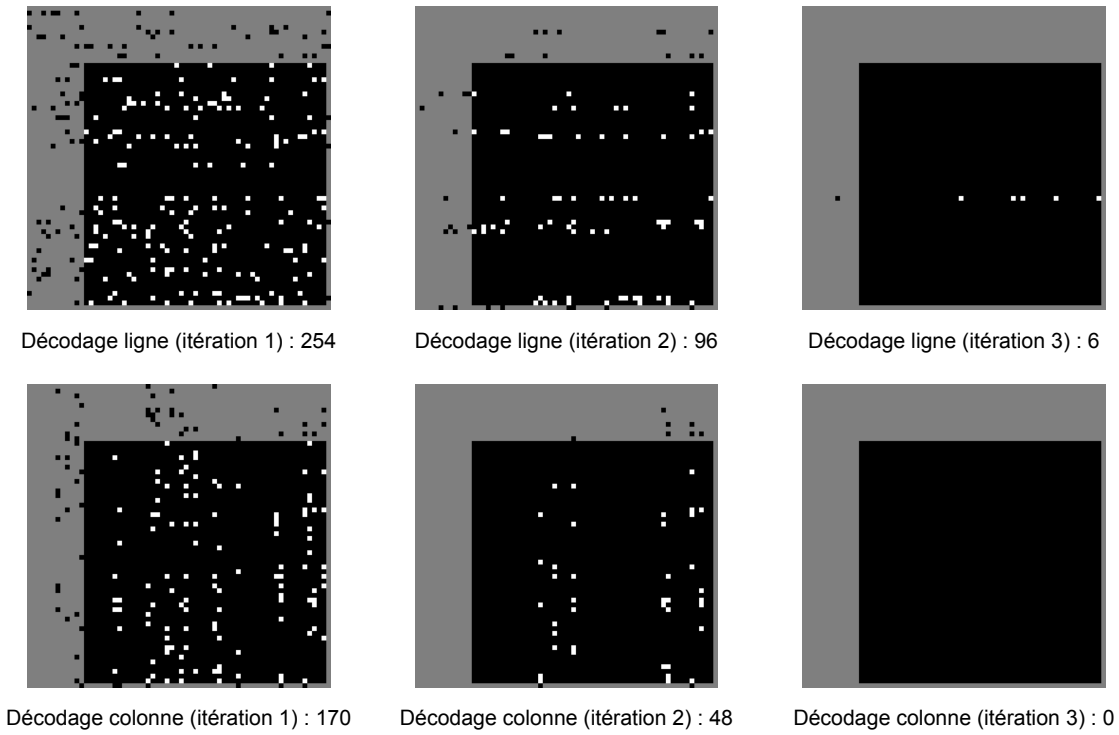


Figure 3.7 – *Evolution des erreurs en fonction du nombre d'itérations - BCH(64,51,6)²*

tion plusieurs configurations de codes produits avec différents « payloads » (voir tableau 3.2). Une fois le message codé, on obtient une image binaire carrée. La suite du processus de mise en forme de la marque ainsi que le processus de tatouage proprement dit, restent identiques à ceux décrits au chapitre 2. La marque binaire obtenue est ensuite sur-échantillonnée, dupliquée et bruitée, avant d'être modulée avec le support. En effet, au niveau de la mise en forme de la marque, les codes par répétitions restent indispensables pour deux raisons. D'une part, la modulation utilisée ne permet pas une probabilité d'insertion de un (*i.e.* statistiquement seulement un bit sur deux de la marque est modulé avec le support, mais pour des raisons de visibilité, dans la pratique l'insertion est comprise entre 0,20 et 0,5), d'autre part, il est nécessaire que toute l'image soit tatouée afin d'être résistant aux opérations de recadrage. Un compromis doit donc être trouvé entre la taille du message à cacher $k_l \times k_c$, la taille du code produit $n_l \times n_c$, le facteur de sur-échantillonnage e et le nombre de réplications L pour une taille d'image $N_l \times N_c$. Pour une taille d'image donnée, connaissant les paramètres k_l, k_c, n_l, n_c et e, le nombre de réplications par bit codé peut être borné de manière supérieure par :

$$L \leq \frac{N_l \times N_c}{n_l \times n_c \times (1+\alpha) \times e^2} \tag{3.21}$$

où le paramètre α représente le nombre de bits de resynchronisation ajouté par bit d'information. L'utilisation de ces bits sera abordée au chapitre 4.

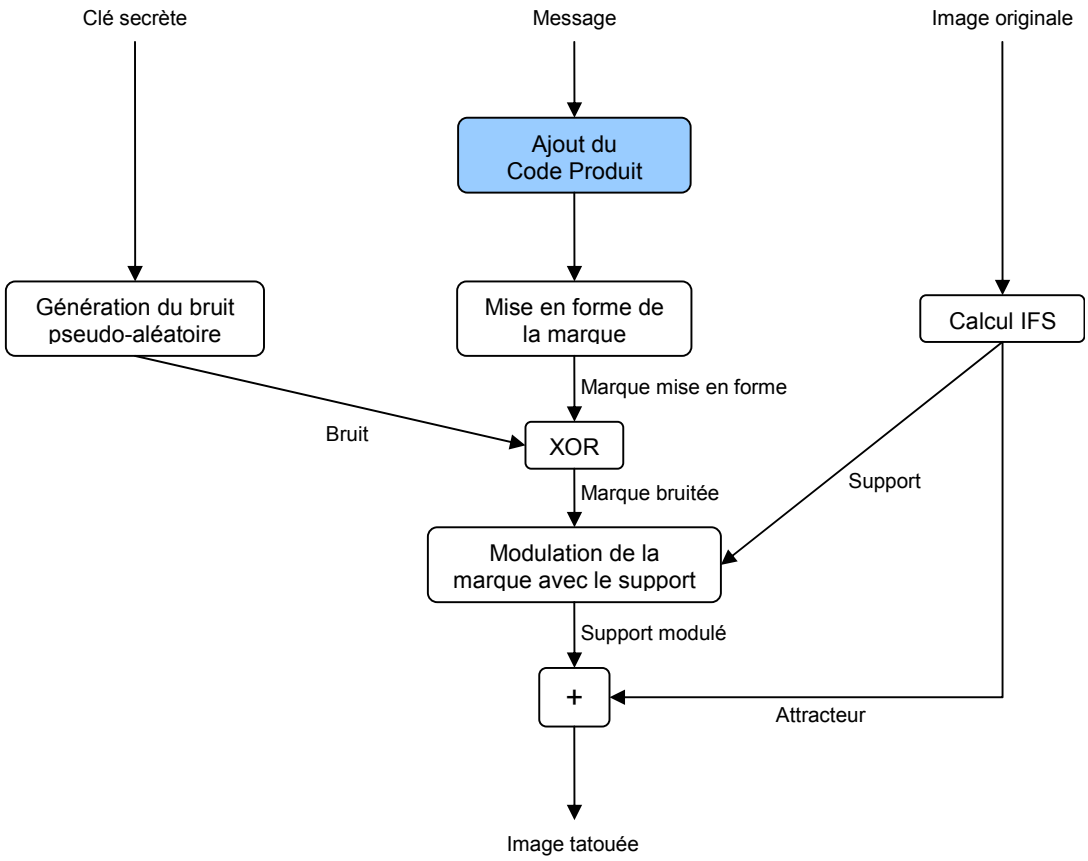


Figure 3.8 – *Nouveau schéma du processus d'insertion*

Payload	Format du message original	Codes BCH utilisés	Format du message turbo codé	Taille finale du message
77 bits	11x7	BCH(16,11)xBCH(16,7)	16x16	256 bits
121 bits	11x11	$BCH(16,11)^2$	16x16	256 bits
441 bits	21x21	$BCH(32,21)^2$	32x32	1024 bits
676 bits	26x26	$BCH(32,26)^2$	32x32	1024 bits
3249 bits	57x57	$BCH(64,57)^2$	64x64	4096 bits

Tableau 3.2 – *Différentes configurations de codes produits utilisés pour différents payloads*

3.4. Modification de l'étape d'extraction

Les modifications à apporter à l'étape d'extraction ne concernent que la phase de décodage du message caché dans l'image (*c.f.* Figure 3.9). Les calculs de l'attracteur et du support restent pour leur part inchangés. Le décodage du message est désormais réalisé en deux temps. On procède dans un premier temps au décodage des codes par répétition. Puis, dans un deuxième temps, le résultat obtenu est envoyé à l'entrée du turbo décodeur qui fournit alors le message décodé final.

3.4.1. Décodage des codes par répétition

Les codes par répétition peuvent être décodés de deux façons différentes : soit par un décodage « souple » (ou « soft »), soit par un décodage « dur » (ou « hard »). La manière dont ces codes sont décodés influe sur la suite du processus d'extraction et principalement sur la qualité du décodage des turbos codes en bloc.

Le décodage « dur » des codes par répétition correspond à celui décrit au chapitre 2 (*c.f.* paragraphes 2.2.1 et 2.2.3). Il consiste dans un premier temps à appliquer un seuillage au niveau du support T_{extrait}. Ce seuillage permet d'éliminer les éléments du support ne codant pas d'information et d'attribuer aux autres une valeur binaire {-1, 1} au sens de la règle de modulation utilisée à l'insertion. On prend ensuite une décision ferme sur la valeur finale de chaque bit, suivant un système de vote majoritaire (*i.e.* la sortie du décodeur est alors une séquence binaire).

Le décodage « souple » des codes par répétition peut être, quant à lui, réalisé de différentes manières. La première solution consiste à appliquer le même seuillage que précédemment au niveau du support, mais au lieu d'appliquer ensuite une décision par vote majoritaire, on calcule pour chaque bit d'information l'écart en nombre de voix normalisé par le nombre d'observations. La sortie du décodeur ne sera plus binaire comme dans le cas d'un décodage « dur », mais prendra ses valeurs dans intervalle [-1, 1]. La deuxième solution est similaire à la première dans son principe, à la différence près que l'on pondère la voix d'un élément du support en fonction de son amplitude. D'une manière générale, on attribuera plus d'importance à un élément si son amplitude en valeur absolue se situe au centre de l'intervalle $[\delta_b, \delta_h]$ plutôt que près des bornes (typiquement la valeur de pondération est fonction d'une gaussienne centrée sur l'intervalle $[\delta_b, \delta_h]$). Enfin, la dernière solution consiste simplement à calculer pour chaque bit du message la moyenne des amplitudes des éléments du support codant de l'information. Dans la pratique, la manière de calculer les observations à fournir à l'entrée du turbo décodeur influe très peu sur le décodage itératif des codes produits. Par simplicité nous retiendrons donc la dernière solution pour toutes les expérimentations réalisées dans la suite de cette thèse.

3.4.2. Décodage itératif du code produit

Suivant que l'on a réalisé un décodage « souple » ou « dur » des codes par répétition, l'entrée du turbo décodeur sera alimentée par une séquence binaire ou réelle. Lorsque la séquence d'entrée est réelle, cela signifie que chaque symbole 0 ou 1 est individuellement accompagné d'une information de fiabilité. Cette information peut être prise en compte pour améliorer les performances du décodage. Dans le cas où l'entrée est binaire, on perd cette information sur la fiabilité des bits d'information. Nous montrerons au paragraphe 3.5 que cela a des répercutions importantes sur la qualité du décodage turbo du code produit. Le décodage proprement dit est ensuite réalisé de ma-

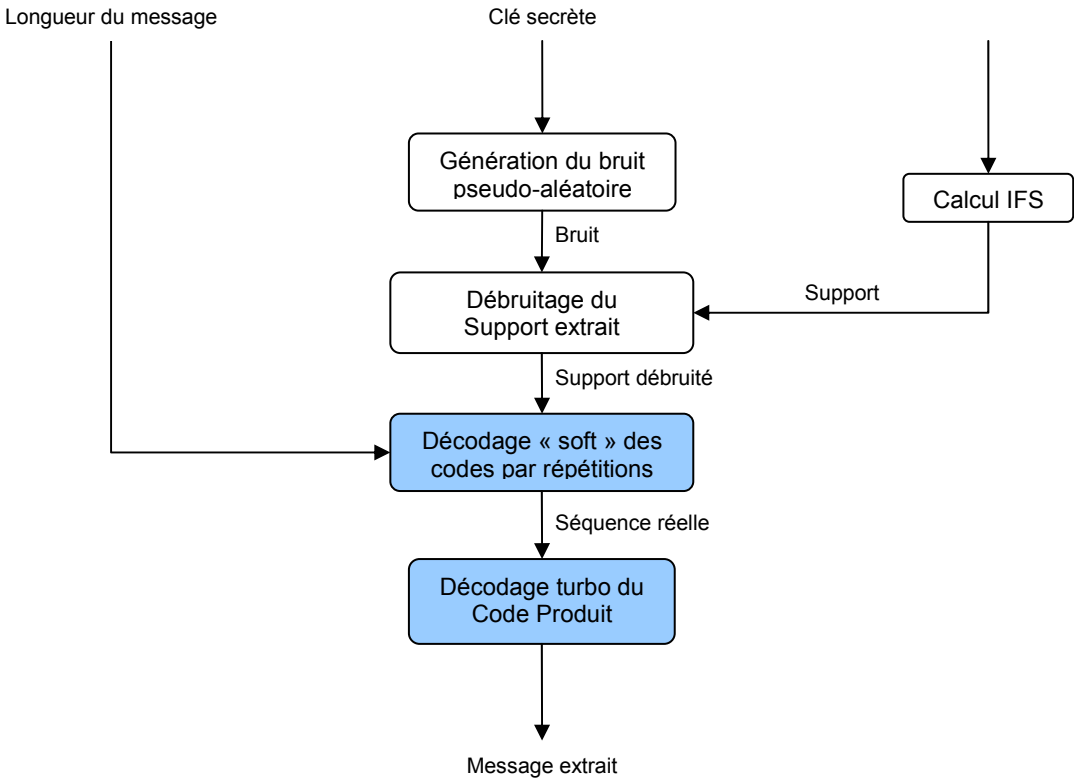


Figure 3.9 – *Nouveau schéma du processus d'extraction*

nière itérative, alternativement sur les lignes et les colonnes.

En pratique, lorsque l'on réalise un décodage souple du code par répétition, l'observation utilisée au niveau du décodeur de Chase-Pyndiah résulte de la moyenne des L observations des répliques du signal émis. Si le nombre L est suffisamment élevé, le théorème central limite s'applique et l'observation x s'écrit sous la forme :

$$x = \alpha.w + \beta \tag{3.22}$$

où α est une variable aléatoire gaussienne de moyenne $1/\sqrt{2\pi}$ et de variance $\mu^2/2L$ et β une variable aléatoire gaussienne centrée de variance $\sigma^2/2$.

α est en fait un estimateur sans biais de la moyenne des $y_i s_i$. Nous avons vu que la variance de cet estimateur est inversement proportionnelle à L. Dès lors, plus L est élevé, meilleur est l'estimateur. Pour des valeurs suffisantes de L, α peut être considéré comme proche de $1/\sqrt{2\pi}$. Dans ce cas, le problème de la connaissance de l'atténuation, soulevé au paragraphe 3.2.2, ne se pose plus. L'algorithme de décodage est alors optimal et il suffit de normaliser les échantillons pondérés par ce facteur. Par conséquent, nous considèrerons toujours une valeur de L suffisamment grande (*e.g.* supérieure ou égale à 16).

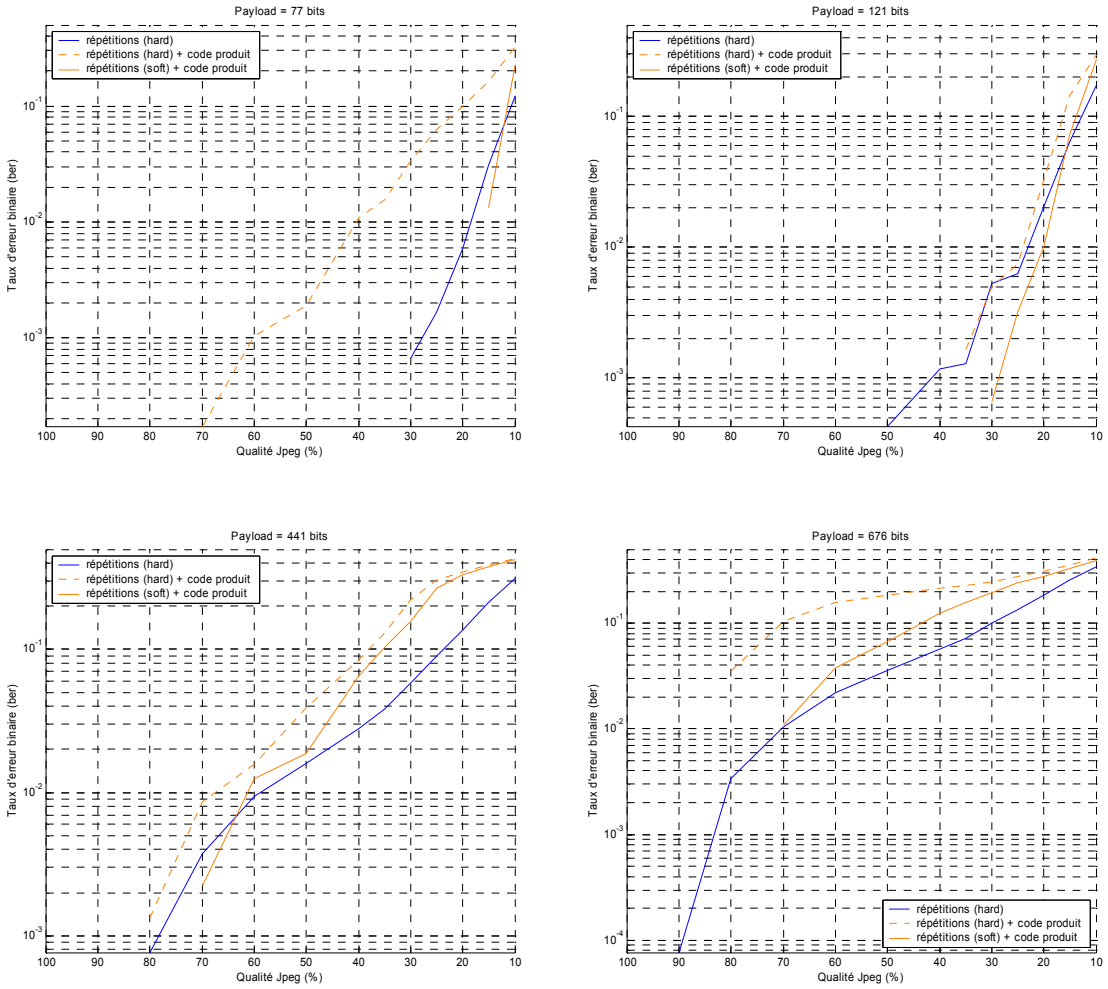


Figure 3.10 – *Taux d'erreur binaire vis-à-vis de la compression Jpeg*

3.5. Résultats

Afin d'évaluer les performances des codes produits, nous avons comparé la robustesse du message vis-à-vis de la compression Jpeg et de l'ajout d'un bruit gaussien à l'image, suivant différentes stratégies de codage (codes par répétition simples, codes BCH et codes produits), pour des tailles de messages équivalentes. Nous avons utilisé pour nos expériences une base de 75 images de taille 512×512 pixels (les images utilisées sont représentées en annexe A de cette thèse). Les résultats sont exprimés d'une part, en termes de taux d'erreur binaire moyen (BER) et de taux d'erreur par message (MER).

Remarque : les expériences ayant été réalisées sur une base limitée à 75 images, il est important de noter que la précision des résultats est de l'ordre de 10^{-2} pour le taux d'erreur par message.

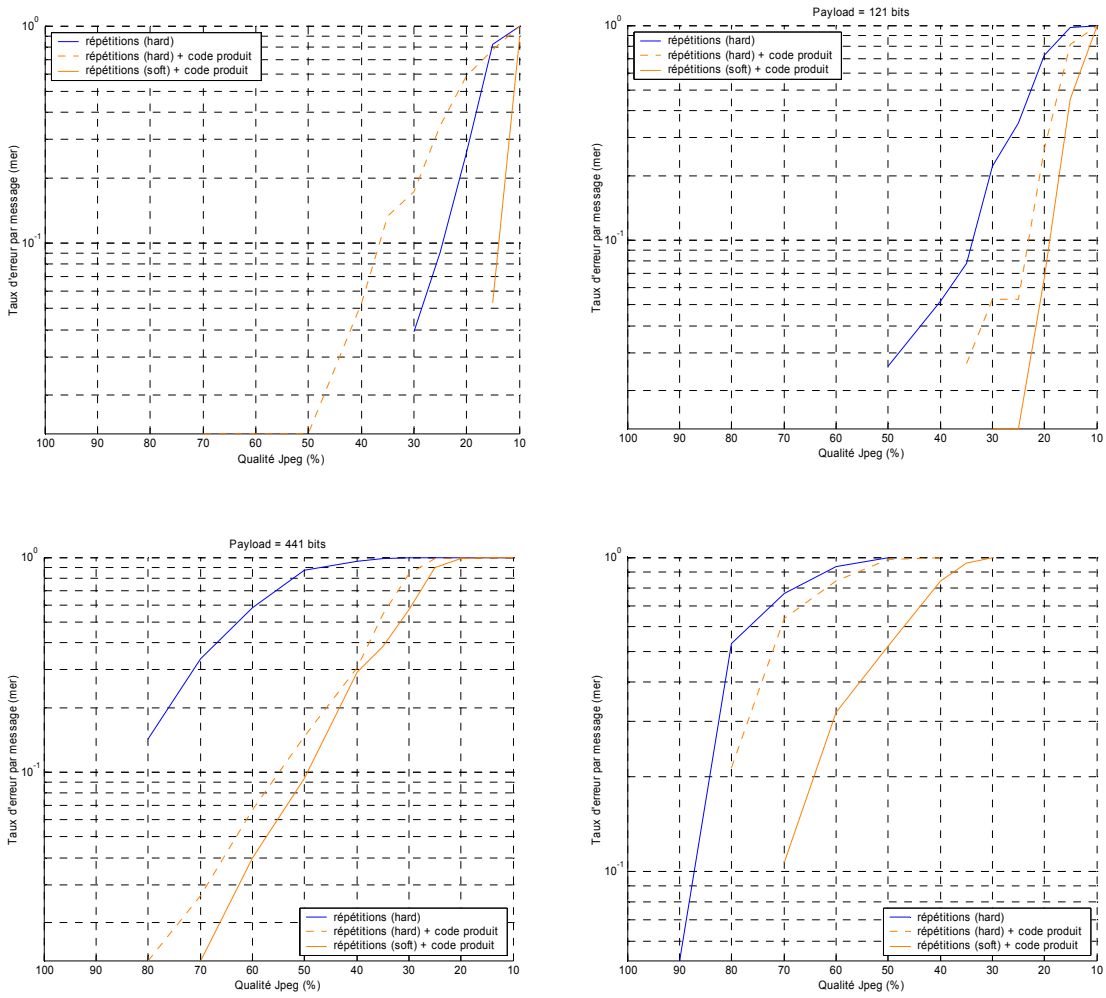


Figure 3.11 – *Taux d'erreur par message vis-à-vis de la compression Jpeg*

3.5.1. Compression Jpeg

Nous présentons dans ce paragraphe des résultats obtenus face à divers taux de compression Jpeg, pour différentes tailles de message (*i.e.* 77, 121, 441 et 676 bits). Les graphiques de la figure 3.10 représentent le taux d'erreur binaire moyen et ceux de la figure 3.11 le taux d'erreur par message. Les différentes courbes illustrent les performances obtenues expérimentalement suivant diverses stratégies de codage et de décodage. La courbe bleue correspond aux performances de l'algorithme de base, c'est-à-dire à l'utilisation d'un code par répétition simple. Les deux autres courbes (en orange) illustrent les performances obtenues en concaténant un code par répétition et un code produit. Lorsque l'on étudie ces deux dernières, on constate sans surprise un écart assez net des performances selon que l'on effectue un décodage dur (courbe en pointillés) ou souple (courbe en tait plein) des répétitions. Ces résultats mettent en évidence l'importance du traitement à

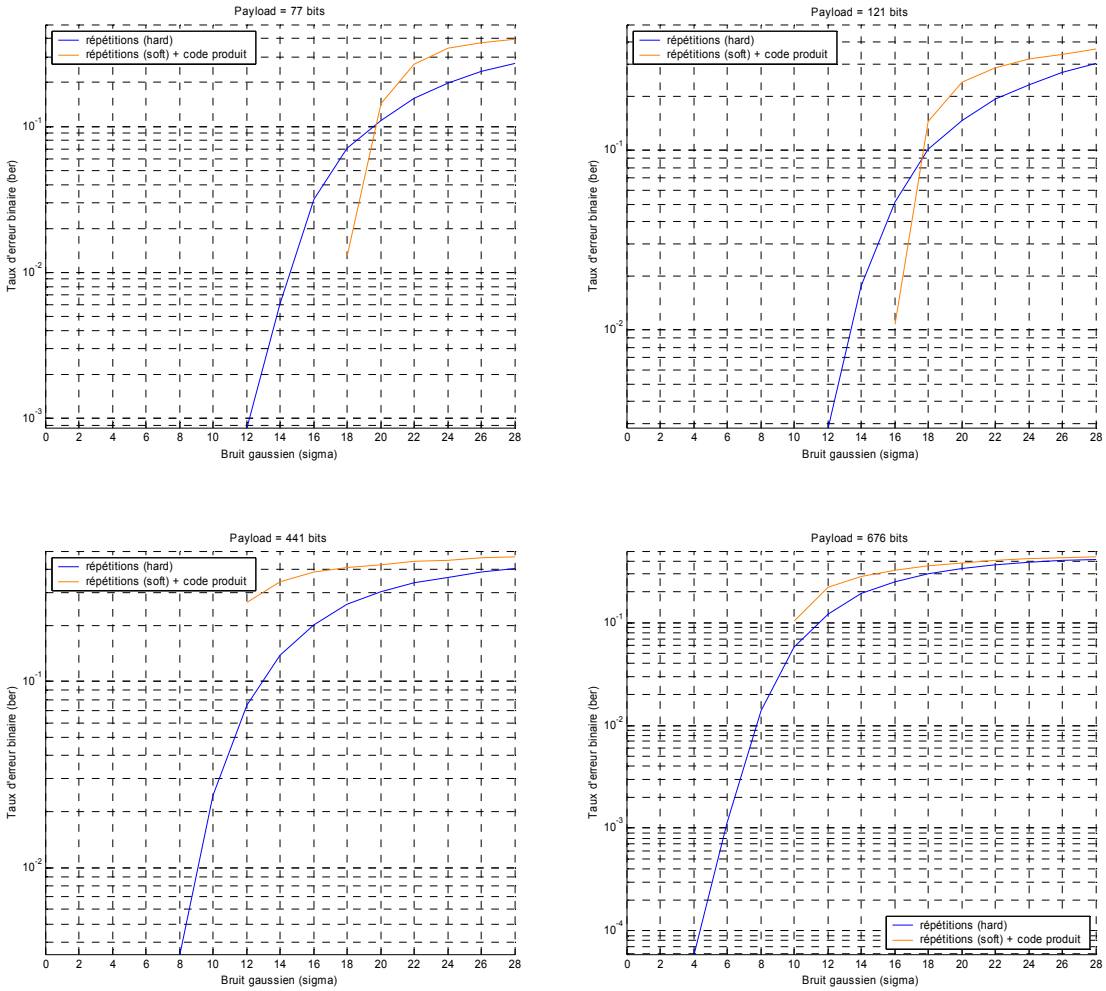


Figure 3.12 – *Taux d'erreur binaire vis-à-vis de l'ajout d'un bruit gaussien*

appliquer aux observations à l'entrée du turbo décodeur. Enfin, en analysant les courbes de taux d'erreur par message, on note un gain significatif en robustesse lorsque l'on utilise des codes produits pour les différentes tailles de séquences utilisées. A titre d'exemple, pour une taille de message de 121 bits, les bits d'informations sont extraits sans erreur jusqu'à un Jpeg de qualité 25%, alors qu'en utilisant un simple code par répétitions les premières erreurs surviennent à partir d'un facteur qualité de 50%.

Cependant pour des payloads beaucoup plus importants (*e.g.* 3249 bits), le gain est quasi-nul. Ces mauvaises performances s'expliquent par la manière dont les bits du code produit sont ensuite répliqués (*i.e.* localement et globalement) au niveau du support lors de la mise en forme du tatouage. Pour des codes produit de grandes dimensions (*e.g.* 64×64), le sur-échantillonnage d'un facteur 3 fait que les différentes duplications d'un même bit se trouvent très peu dispersées dans l'image. De ce fait, compte tenu que les différentes régions de l'image ne permettent pas toutes d'insérer avec la

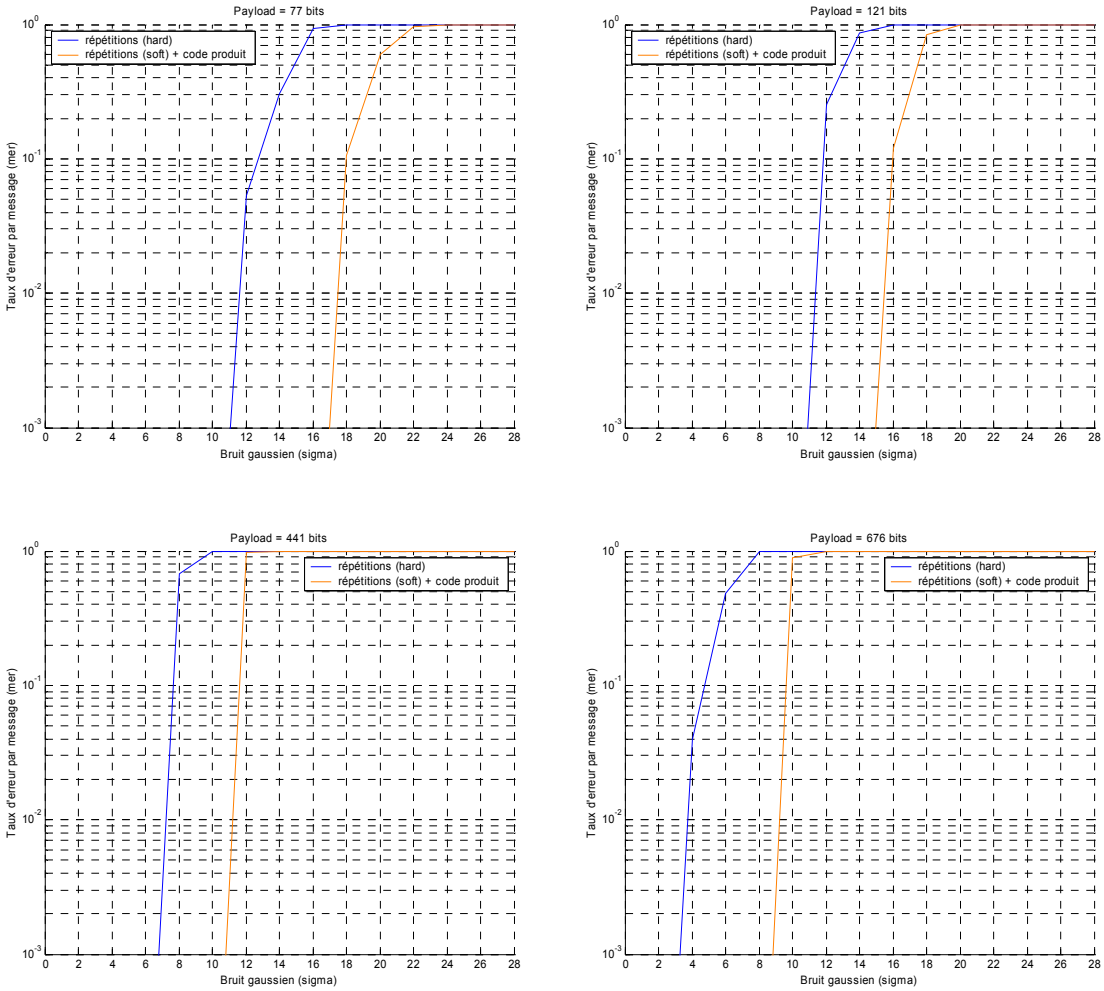


Figure 3.13 – *Taux d'erreur par message vis-à-vis de l'ajout d'un bruit gaussien*

même robustesse le tatouage (en fonction du contenu fréquentiel et de la règle de visibilité), certains bits risquent d'être peu, voire pas, cachés dans l'image tatouée. Dès lors, même sans attaque, une partie des bits d'information est totalement perdue. Dans ce cas, il est alors préférable de réduire le facteur de sur-échantillonnage afin de mieux répartir les bits dans l'image (*n.b.* le nombre de répliques reste inchangé).

3.5.2. Ajout d'un bruit gaussien

Nous présentons dans ce paragraphe des résultats similaires vis-à-vis de l'ajout d'un bruit gaussien à l'image. Les figures 3.12 et 3.13 représentent respectivement le taux d'erreur binaire moyen et le taux d'erreur par message suivant les deux stratégies de codage. Les courbes bleues matérialisent les performances de l'algorithme de base et les oranges celles correspondant à la concaténation d'un

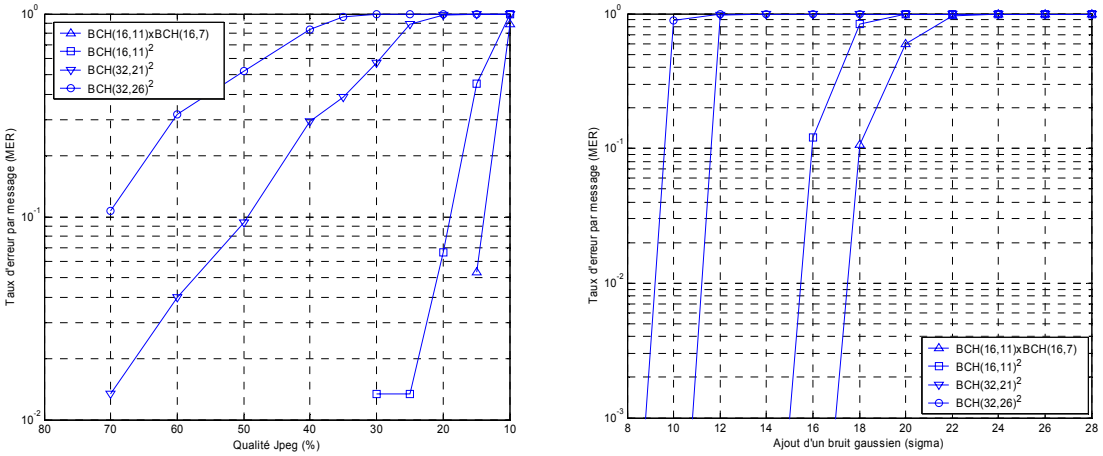


Figure 3.14 – *Evolution des performances en fonction de la longueur du message*

code produit et d'un code par répétition. Comme pour les expériences avec la compression Jpeg, l'utilisation d'un code produit apporte un gain significatif en robustesse, notamment en termes de taux d'erreur par message.

3.5.3. Comparaison en fonction de la taille du message

Les deux graphiques de la figure 3.14 montrent l'évolution du MER en fonction de la longueur du message, vis-à-vis de la compression Jpeg et de l'ajout d'un bruit gaussien. Pour des tailles fixées de mots de code (*i.e.* 256 ou 1024 bits), on observe une légère dégradation des performances lorsque le payload augmente (*i.e.* de 77 à 121 bits, ou de 441 à 676 bits). En effet, cela s'explique simplement par le fait que lorsque l'on augmente le nombre de bits d'information sans changer la longueur du code, le pouvoir de correction d'erreurs du code se trouve diminué. D'autre part, pour des codes ayant la même distance de Hamming (*e.g.* BCH(16,11)² et BCH(32,26)²), on note que cette dégradation est plus importante pour des codes de grandes dimensions. A première vue, ce résultat peut sembler aberrant. Théoriquement, lorsque l'on augmente la longueur d'un code, tout en conservant la même distance de Hamming, le gain de codage augmente. Malheureusement, dans notre cas, ce gain de codage n'est pas suffisant pour compenser la perte due à la diminution du facteur de répétition L, qui est ici divisé par quatre. L'énergie du signal reçu par bit est donc divisée par quatre et la probabilité d'insertion d'un bit varie de $(1-(1-p)^{-L})$ où p représente la probabilité d'insertion par pixel.

3.5.4. Comparaison Code Produit – BCH simple

Pour terminer, nous avons également comparé les performances des codes produits par rap-

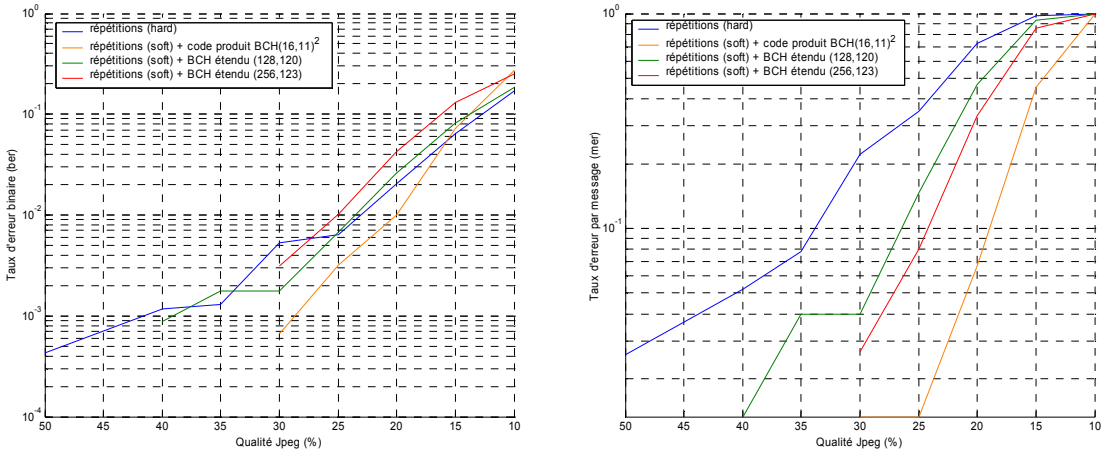


Figure 3.15 – *Comparaison code produit / BCH*

port à celles d'un BCH simple. Dans l'exemple de la figure 3.15, nous avons utilisé un code produit BCH $(16,11)^2$, et deux codes BCH étendus : le BCH(128,120) et le BCH(256,123). La taille de la séquence binaire utile est sensiblement la même pour les trois codes (*i.e.* 121 bits pour le code produit, contre 120 et 123 bits pour les BCH). Dans les deux cas, nous avons procédé à un décodage souple des codes par répétition. Les résultats obtenus vis-à-vis de la compression Jpeg montrent sans ambiguïté des performances nettement supérieures pour le code produit quelle que soit le taux de compression Jpeg. Les codes BCH se montrent, pour leur part, plus performants qu'un simple code par répétitions, mais le gain est nettement moins important qu'avec le code produit.

4. Conclusion

Nous avons apporté dans ce chapitre une première amélioration à l'algorithme de tatouage original. Cette amélioration se situe principalement au niveau de l'étape de mise en forme du tatouage. Elle consiste à encoder successivement la séquence binaire d'information à l'aide d'un code produit BCH, puis d'un code par répétitions. Cette combinaison de codes correcteurs permet, pour une longueur de message donnée, d'obtenir un gain significatif en robustesse, sans introduire de dégradation visuelle supplémentaire. Dans le cadre de notre algorithme de tatouage, l'utilisation d'un code par répétitions, *a priori* peu performant, reste cependant indispensable pour deux raisons principales. D'une part, l'image doit être tatouée dans son intégralité de manière à pouvoir récupérer le tatouage dans le cas où l'image a été recadrée (*n.b.* ceci est valable quelque soit l'algorithme de tatouage utilisé), d'autre part la probabilité d'insertion étant inférieure à 50% dans notre cas, il est nécessaire d'avoir un minimum d'observations par bit à l'entrée du turbo décodeur afin de compenser cette perte d'information.

Chapitre 4

Robustesse aux déformations géométriques locales et globales

1. Organisation du chapitre

Comme nous l'avons vu au cours du premier chapitre, la robustesse est un des points clés de tout algorithme de tatouage d'image. En effet, pour la majorité des applications, la marque doit pouvoir rester explicitement accessible, même si l'image a subi des manipulations involontaires ou délibérées. Dans ce chapitre, nous nous préoccupons plus particulièrement des problèmes de désynchronisation liés aux transformations géométriques, ainsi qu'aux moyens à mettre en œuvre pour les contrecarrer. Nous proposons, notamment, une technique originale permettant de compenser ces déformations, et plus particulièrement les distorsions aléatoires générées par Stirmark. La méthode mise au point, opère directement dans le domaine spatial et ne nécessite à l'extraction aucune connaissance *a priori* sur l'image originale et le message caché. Nous présentons également une méthode complémentaire permettant de détecter les transformations affines globales (*i.e.* rotations et changement d'échelle).

2. Contexte du problème

2.1. Problématique liée aux transformations géométriques

On distingue les manipulations géométriques locales et globales dans la mesure où les techniques de resynchronisation diffèrent. Dans le premier cas, on dispose d'un nombre d'échantillons

important pour estimer la transformation, alors que dans le second cas, on doit se contenter d'un nombre plus réduit puisque l'estimation doit être réalisée localement. Parmi les transformations géométriques globales les plus courantes, figurent les rotations de quelques degrés, les petites translations, les changements d'échelle et les opérations de recadrage. Ces décalages sont très fréquents dans une chaîne de traitement d'image et sont d'ordinaire imperceptibles si l'on ne dispose pas de l'image originale. Les transformations locales, quant à elles, dépendent généralement, soit de la position du pixel concerné dans l'image, soit d'un processus pseudo aléatoire (*e.g.* Stirmark). Elles sont habituellement accompagnées d'une fonction d'interpolation, de type bilinéaire par exemple, pour assurer une continuité au niveau des frontières des régions subissant des transformations.

Du point de vue du tatouage d'image, les manipulations géométriques ont principalement pour effet d'introduire une désynchronisation entre la signature contenue dans l'image et l'opérateur d'extraction. La marque reste présente dans l'image, le plus souvent, mais le décodeur n'est plus capable de la détecter. L'explication est que le tatouage subit généralement les mêmes distorsions que l'image. De ce fait, en l'absence de repère, le détecteur cherche à extraire les bits du message à des endroits dans l'image où ils ne s'y trouvent plus.

2.2. Méthodes classiques de resynchronisation

La difficulté pour retrouver un tatouage dans une image dont la géométrie a été modifiée, ainsi que les techniques pour y parvenir, ne sont pas les mêmes suivant le mode d'extraction utilisé. En mode non-aveugle, par exemple, il suffit d'utiliser l'image originale afin d'identifier la nature des manipulations géométriques, et de déterminer ensuite les transformations « pseudo » inverses à appliquer à l'image attaquée de manière à compenser ces décalages. Dans le cadre d'une extraction en mode aveugle, le problème est tout autre. Nous ne disposons d'aucune information sur le contenu du message caché, ni sur la géométrie de l'image originale. Ce contexte est bien évidemment très contraignant dans la mesure où l'on ne sait absolument pas de quelle(s) manière(s) l'image a été éventuellement manipulée.

2.2.1. Méthodes ayant recours à l'image originale

Davoine *et al* [DBHC99] proposent une méthode permettant de compenser les déformations géométriques engendrées par Stirmark en ayant recours à l'image originale. Il s'agit d'un prétraitement appliqué directement à l'image attaquée, indépendamment de la technique de tatouage utilisée. Le principe de base consiste à plaquer un maillage triangulaire sur l'image attaquée, puis à le déformer, en appliquant de légères translations aux sommets des triangles, de manière à approximer au mieux l'image originale (*c.f.* Figure 4.1). La méthode donne de bons résultats, mais le fait d'avoir recours à l'image originale restreint considérablement son champ applicatif.

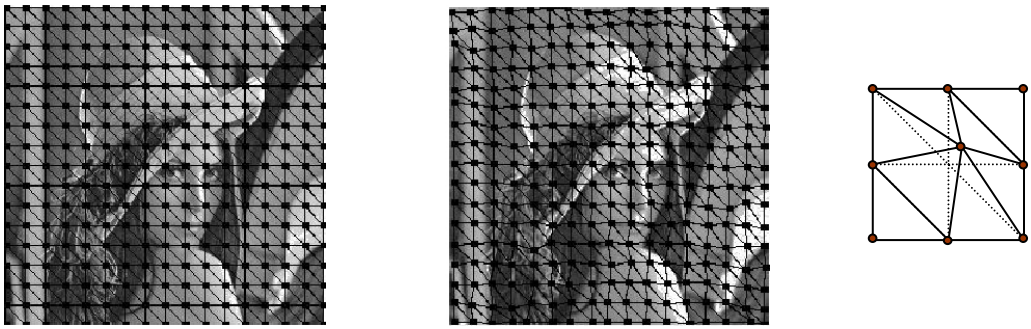


Figure 4.1 – *Compensation des déformations géométriques à l'aide de l'image originale*

2.2.2. Utilisation d'un « template »

Actuellement, une des techniques les plus couramment utilisées consiste à insérer, en plus de la marque, un motif prédéfini ou « template » dans l'image. Ce motif sert de repère, au moment de l'extraction du tatouage, afin de détecter et compenser certaines déformations géométriques telles que des rotations, des translations, ou bien encore des changements d'échelle. Pereira et Pun [PP99] proposent d'insérer un « template » dans un anneau correspondant aux moyennes fréquences de la FFT de l'image (*c.f.* Figure 4.2). Le motif est généré en augmentant l'amplitude de certains coefficients créant ainsi des pics locaux. Lors de l'extraction, la resynchronisation s'effectue en trouvant la transformation affine qui permet de faire correspondre le motif initial avec les maxima locaux détectés. Cette méthode donne de très bons résultats, malheureusement l'ajout d'un tel motif dans l'image est facilement repérable dans le domaine fréquentiel (*c.f.* Figure 4.3). Il est alors relativement aisé pour une personne malintentionnée de supprimer ces pics (grâce à le « Template Removal Attack » [HVR01]), privant ainsi le processus d'extraction de tout moyen de resynchronisation.

2.2.3. Insertion du tatouage dans un espace invariant

La solution idéale consisterait à insérer le tatouage dans un espace invariant aux déformations géométriques tel que l'espace engendré par la transformée de Fourier-Mellin [RP97] (*c.f.* chapitre 1) ou bien encore celui proposé par Lin *et al.* [LBC+00]. Cette solution est théoriquement envisageable pour des transformations affines globales simples, comme les rotations, les translations et les changements d'échelle, mais devient très rapidement inapplicable dès qu'il s'agit de déformations géométriques locales. De plus les problèmes d'approximation liés à la nature discrète des images font que l'espace d'insertion se trouve, au final, considérablement réduit pour contenir une marque robuste.

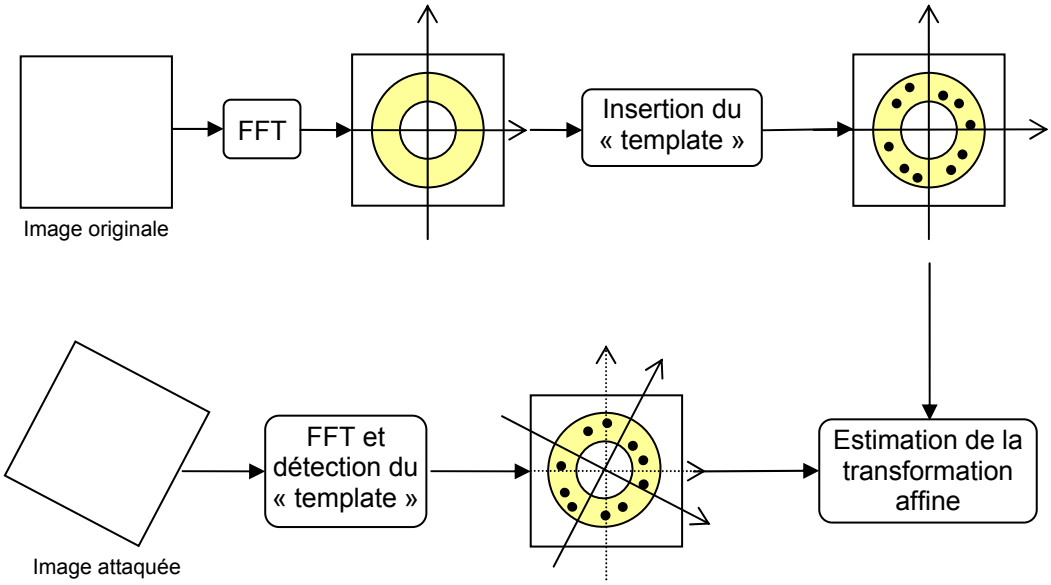
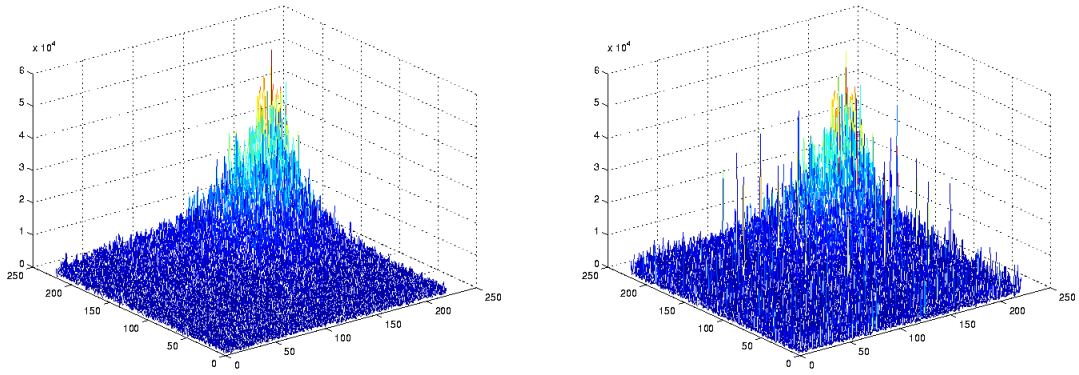


Figure 4.2 – *Principe de resynchronisation à l'aide d'un template*



(a) FFT image originale de Lena

(b) FFT image tatouée (par Digimarc) de Lena

Figure 4.3 – *Exemple de système de resynchronisation à l'aide d'un motif périodique*

2.2.4. Utilisation des caractéristiques géométriques de l'image

Une autre possibilité pour pallier aux effets des déformations géométriques, réside dans le choix des régions de l'image recevant le tatouage. La plupart des méthodes dites de « deuxième génération » proposent de tenir compte des caractéristiques géométriques de l'image. Certaines de ces méthodes [BCM00] se basent sur un découpage en régions de l'image par rapport à des points ca-

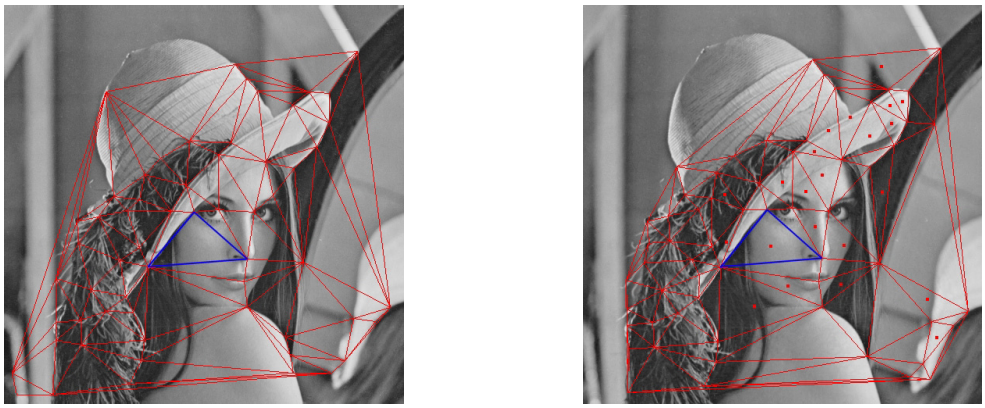


Figure 4.4 – *Tatouage d'une image en fonction de points caractéristiques*

ractéristiques (obtenus par exemple à l'aide d'un détecteur de points d'intérêt), suivi d'une triangulation de Delaunay (*c.f.* Figure 4.4). Pour chacune des régions trouvées précédemment, la marque est mise à la forme de la région cible (par transformation linéaire) et fusionnée avec celle-ci par une technique de modulation classique. L'extraction du tatouage est réalisée de manière duale de l'insertion. C'est-à-dire qu'après avoir découpé l'image tatouée en régions suivant le même procédé, on déforme chacune de ces régions de manière à retrouver la forme originale de la marque. La robustesse de cette méthode repose principalement sur la stabilité des points d'intérêt choisis face aux diverses manipulations possibles de l'image. Il est clair qu'une perte significative de ces points d'intérêt entraîne irrémédiablement la perte du tatouage.

3. Compensation des déformations géométriques locales

3.1. Description générale de la méthode

La méthode que nous proposons pour contrer les déformations géométriques locales se situe dans un contexte d'extraction en mode aveugle. C'est-à-dire que la resynchronisation est réalisée sans aucune connaissance *a priori* sur l'image originale, ni sur le contenu du message caché. Notre technique de resynchronisation consiste, tout d'abord lors de l'étape d'insertion, à adjoindre au message une information additionnelle prédéfinie. Ces bits supplémentaires sont appelés bits de resynchronisation ou bits de contrôle. Ces bits servent ensuite, lors de l'extraction du tatouage, de points de repère pour estimer et compenser les déformations géométriques locales ou globales de faible amplitude. Les déformations géométriques sont approximées à l'aide d'une méthode classique de calcul de flux optique.

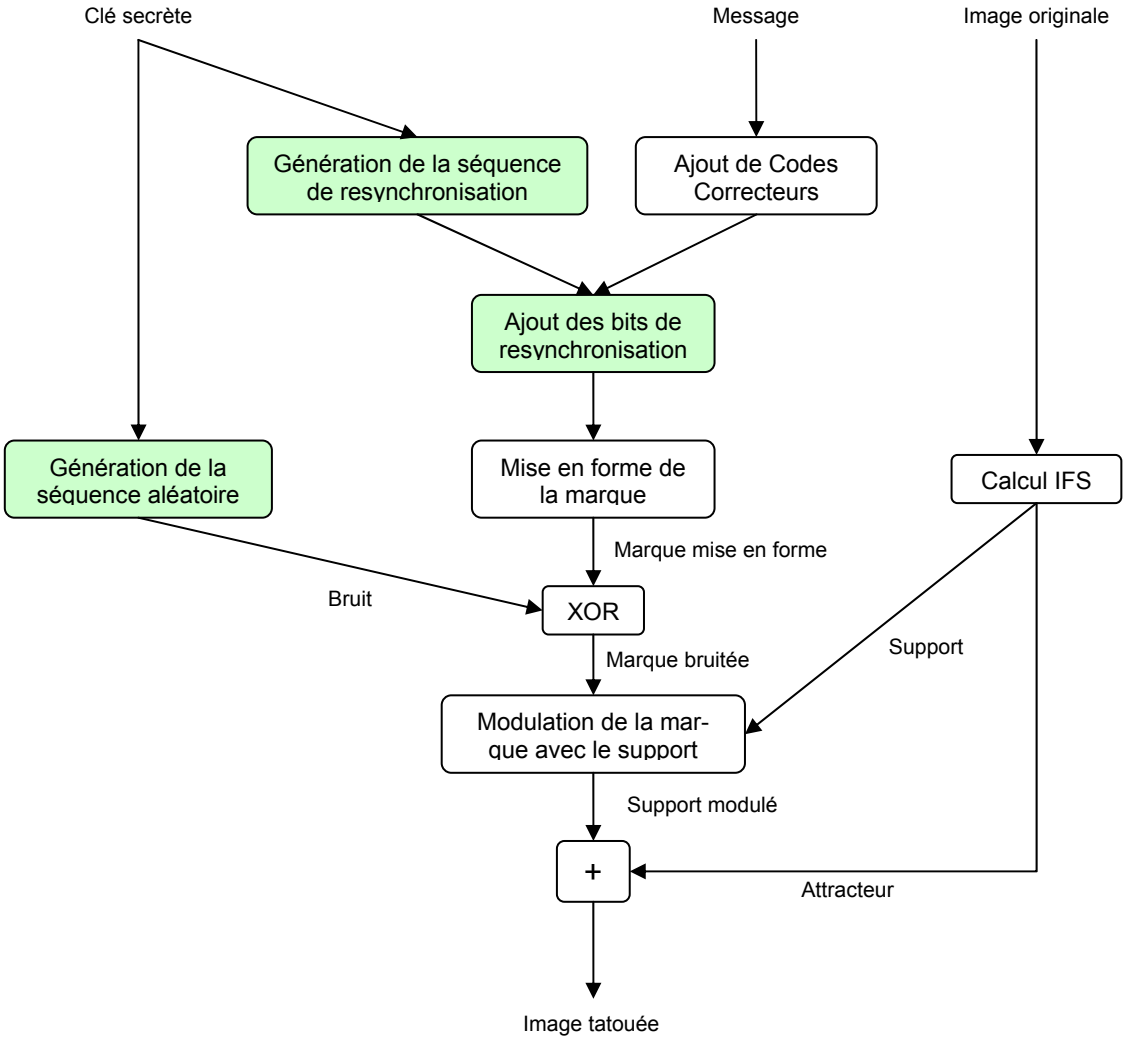


Figure 4.5 – *Schéma du processus d'insertion*

3.2. Mise en forme de la signature

La première phase de notre technique se situe en amont du processus de tatouage. Elle consiste à ajouter à la marque binaire (éventuellement encodée avec des codes correcteurs), des bits de resynchronisation. Le reste de l'algorithme d'insertion reste, quant à lui, inchangé. C'est-à-dire que les bits de resynchronisation sont modulés avec le support de la même manière que les bits d'information.

3.2.1. Détermination du nombre de bits de resynchronisation à ajouter

Le choix du nombre de bits de contrôle à adjoindre aux bits du message est crucial. En effet, il est nécessaire d'en cacher suffisamment pour que le processus de resynchronisation soit efficace.

En effet, la densité des bits de resynchronisation conditionne directement la précision de l'approximation des déformations géométriques par calcul de flux optique. Plus la densité est importante et plus le risque de mauvais appariement sera faible. Il est donc nécessaire d'ajouter un nombre important de bits, d'autant plus, que seulement un bit sur deux du tatouage est modulé avec le support, dans le meilleur des cas (*c.f.* règles de modulation décrites au chapitre 2). Cependant, il ne faut pas perdre de vue que l'ajout des bits de resynchronisation se fait au détriment de la redondance des bits d'information, ce qui a pour conséquence directe d'amoindrir la robustesse du tatouage face aux attaques photométriques et aux découpes de l'image. Il y a donc un compromis à réaliser en terme de ratio entre la quantité de bits de contrôle et la quantité de bits d'information.

Le nombre de bits de resynchronisation n_{bc} à ajouter au message est déterminé par les contraintes suivantes :

- le ratio d_{bc} de bits de resynchronisation, défini comme le rapport entre le nombre théorique de bits de resynchronisation et le nombre de bits d'information,
- le nombre de bits d'informations n_{bi},
- et la contrainte « arbitraire » d'avoir un motif de base (agencement des bits de resynchronisation et des bits d'information) de forme carrée. Le choix de cette forme est motivé par le fait que la plupart des images sont d'apparence compacte et que la duplication spatiale des bits doit être la plus équitable possible. Ce problème est particulièrement vrai pour des messages de taille importante comme ceux qui peuvent être utilisés en intégrité.

$$n_{\mathrm{bc}} = \left[\sqrt{n_{\mathrm{bi}} + d_{bc} \times n_{bi}}\right]^2 - n_{bi} \tag{4.1}$$

Remarque : à titre d'exemple, pour un message de 64 bits, nous avons établi expérimentalement que le meilleur compromis est de 57 bits de contrôle, ce qui donne comme motif de base une marque de 11 pixels de côté.

3.2.2. Répartition spatiale des bits de resynchronisation

La séquence de bits de resynchronisation est générée de manière pseudo-aléatoire à partir de la clé secrète. Les bits de resynchronisation sont ensuite intercalés uniformément aux bits d'information (*c.f.* Figure 4.6). La séquence binaire résultante est ensuite agencée de manière arbitraire suivant le chemin décrit par la figure 4.7 (d'autres types de parcours peuvent également être utilisés : en spirale, en zig-zag, etc.), de manière à obtenir une répartition spatiale relativement homogène entre les différents types de bits. La suite de la mise en forme de la marque reprend exactement le même procédé que celui énoncé au chapitre 2. C'est-à-dire que la marque est sur-échantillonnée, puis dupliquée horizontalement et verticalement, et enfin cryptée (*c.f.* paragraphe 3.2.3), avant d'être finalement fusionnée avec l'image.

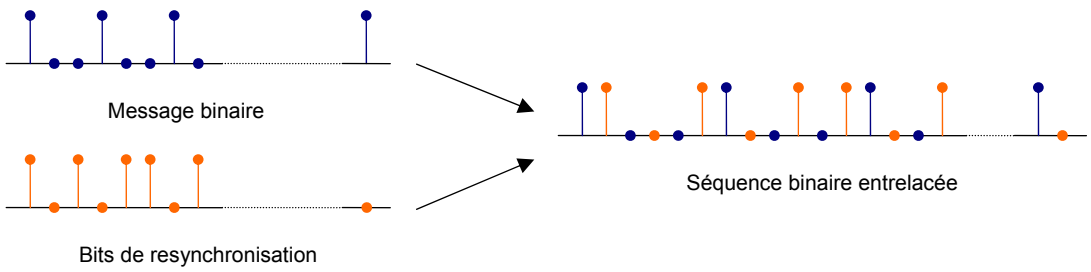


Figure 4.6 – *Entrelacement des bits d'information et des bits de resynchronisation*

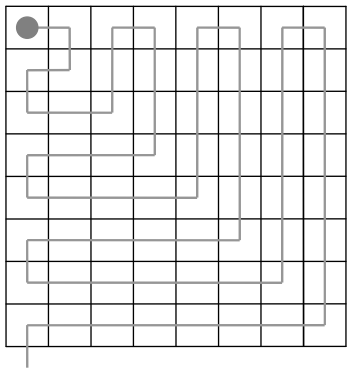


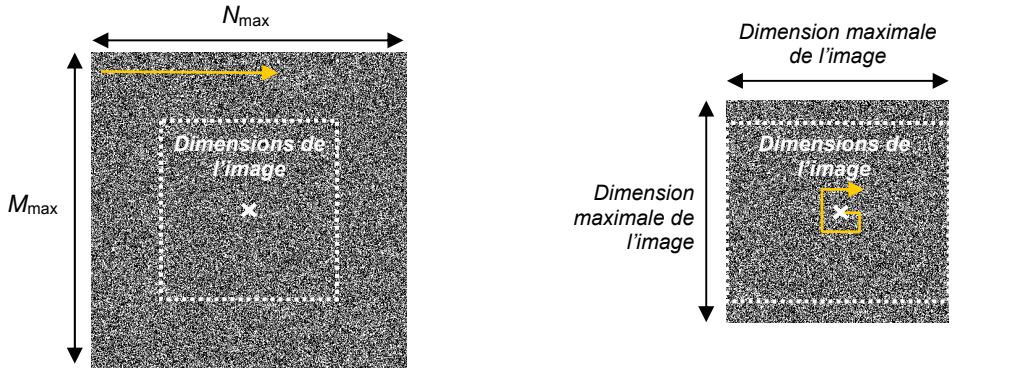
Figure 4.7 – *Schéma de répartition de la séquence binaire formée des bits de data et de resynchronisation*

3.2.3. Précautions à prendre sur la manière de générer la séquence binaire

La manière dont la séquence binaire pseudo-aléatoire bi-dimensionnelle est créée, ou plus exactement la manière dont elle est mise en forme, revêt une importance toute particulière. En effet, nous devons être en mesure de générer la même séquence à l'extraction qu'à l'insertion, et ce quelles que soient les modifications géométriques opérées sur l'image. A titre d'exemple, si la séquence 2D est mise en forme ligne par ligne, on sera dans l'incapacité de la reproduire à l' identique à l'extraction si le nombre de colonnes de l'image a été modifié (*e.g.* recadrage). Il existe différentes solutions pour pallier à ce problème, en voici deux.

La première solution, consiste à créer une séquence de base de dimensions fixes (*e.g.* 2048×2048 pixels) dans lequel on découpe au centre une région à la taille de l'image pour obtenir la séquence binaire pseudo-aléatoire finale (*c.f.* Figure 4.8a). Cette méthode à l'avantage d'être très simple à mettre en œuvre, par contre elle est très coûteuse en mémoire. De plus, le cas particulier des images de dimensions supérieures à la séquence de base doit être traité séparément, en considérant par exemple que la séquence de base est dupliquée horizontalement et verticalement.

La deuxième solution, plus élégante, consiste à générer la séquence pseudo-aléatoire 2D en spirale en partant du centre de l'image jusqu'à atteindre la plus grande des deux dimensions de l'image. La séquence obtenue est ensuite découpée à la taille exacte de l'image (*c.f.* Figure 4.8b). Cette solution à l'avantage d'être adaptée à toutes les tailles d'images et d'être beaucoup moins coûteuse en mémoire que la première. Cependant, les deux méthodes présentent le même défaut vis-à-vis des opérations de recadrage de l'image. Dans le cas d'un « crop » non centré par exemple, seule une partie de la séquence pseudo aléatoire générée à l'extraction sera, en correspondance avec la marque cryptée contenue dans l'image testée, voire aucune correspondance dans certains cas extrêmes.



a) Génération par la méthode 1 *b) Génération par la méthode 2*

Figure 4.8 – *Méthodes pour mettre en forme la séquence binaire pseudo-aléatoire 2D*

3.3. Processus de resynchronisation

Le processus de resynchronisation se décompose en 3 étapes :

- Génération du masque de référence,
- Recherche des motifs formés par les bits de contrôle par un algorithme de « Block Matching » modifié,
- Obtention du masque de référence resynchronisé.

3.3.1. Principe de base

L'idée générale de la méthode de resynchronisation est de parvenir à recaler les bits de contrôle avec le signal de tatouage désynchronisé avant d'extraire le message. L'hypothèse de travail que nous faisons stipule que si l'on est en mesure de resynchroniser les bits de contrôle, les bits d'information seront alors également remis en correspondance. Cette hypothèse est réaliste dans la mesure où les bits d'information et les bits de contrôle sont finement entremêlés. Dans ces conditions, connaissant la valeur et les positions relatives des bits de contrôle, le processus de resynchronisation consiste simplement à rechercher pour chaque portion de signal de tatouage, la séquence de bits de resynchronisation lui correspondant au mieux (*c.f.* Figure 4.10).

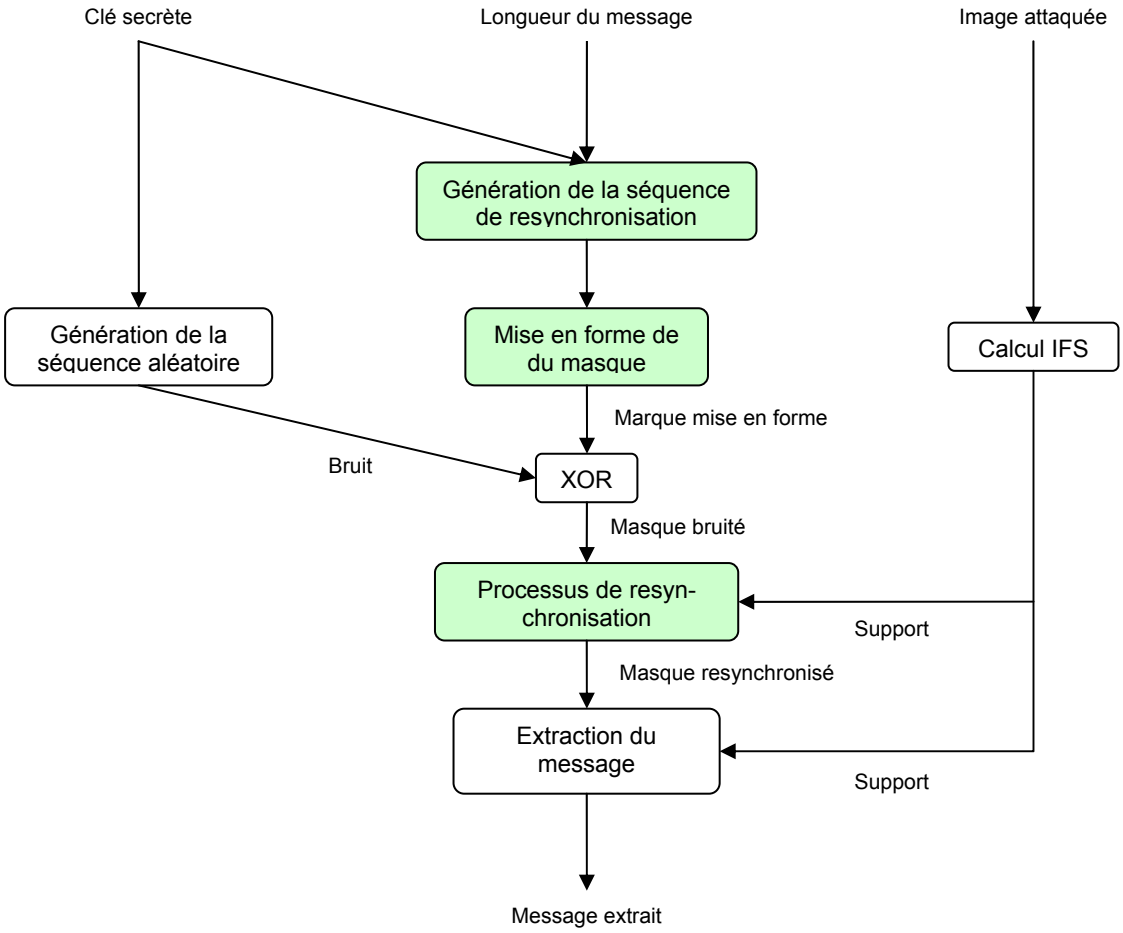


Figure 4.9 – *Schéma du processus d'extraction*

3.3.2. Génération du masque de référence

Une fois, le tatouage modulé à l'image, plus rien ne différencie les bits d'information, des bits de resynchronisation. Pour pouvoir retrouver les bits de resynchronisation, il est donc nécessaire de connaître, en plus de leur valeur, la manière dont ils ont été agencés, afin d'utiliser leurs positions relatives comme élément de repère. Pour cela, on a recours à un masque qui sert ainsi de modèle de référence indiquant précisément l'agencement des bits formant la marque originale telle qu'elle a été insérée dans l'image, ainsi que la manière dont la séquence pseudo-aléatoire a été appliquée. Le procédé de création du masque est donc similaire à celui de la marque, à la différence près que les éléments constituant le masque ne sont ni des « 0 », ni des « 1 », mais des labels identifiant les bits qui ont été cachés. Ces labels permettent d'établir la nature (bit de contrôle ou bit d'information), ainsi que le numéro d'ordre de chaque bit du message. D'autre part, ces identifiants sont signés (en fonction de la séquence pseudo aléatoire ajoutée au masque) de manière à pouvoir déterminer, lors de la reconstruction du message, la contribution, positive ou négative, de chaque point du support, et séparer ainsi la marque du bruit.

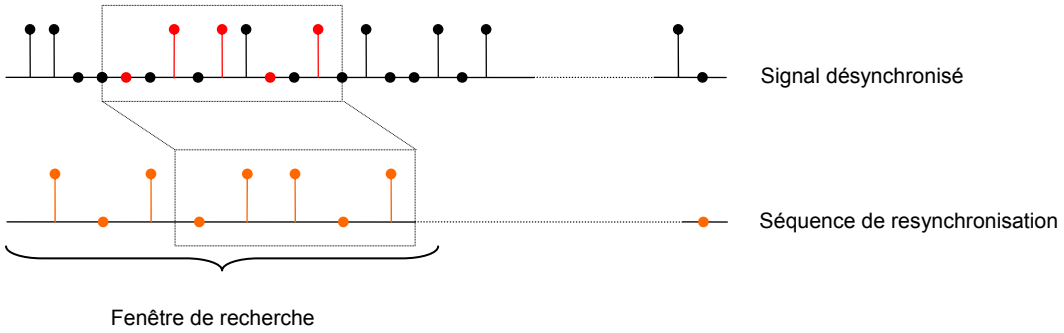


Figure 4.10 – *Principe de la méthode de resynchronisation*

Exemple : soit le message *Msg* (inconnu) à extraire, $Msg = m_1 , m_2 , m_3 , m_4 , m_5 , m_6 , m_7 , m_8 , \ldots , m_k$ et la séquence de resynchronisation $B_c = 1 , 0 , 1 , 0 , 1 , 1 , 0 , 1 , \ldots , 0$.

Les séquences *Msg* et B_c sont tout d'abord entrelacées :

$(Msg, B_c) = m_1 , 1 , m_2 , 0 , m_3 , 1 , m_4 , 0 , m_5 , 1 , m_6 , 1 , m_7 , 0 , m_8 , 1 , \ldots, m_k , 0.$

La nouvelle séquence est alors dupliquée (localement et globalement) et cryptée pour donner $M_{réf}$:

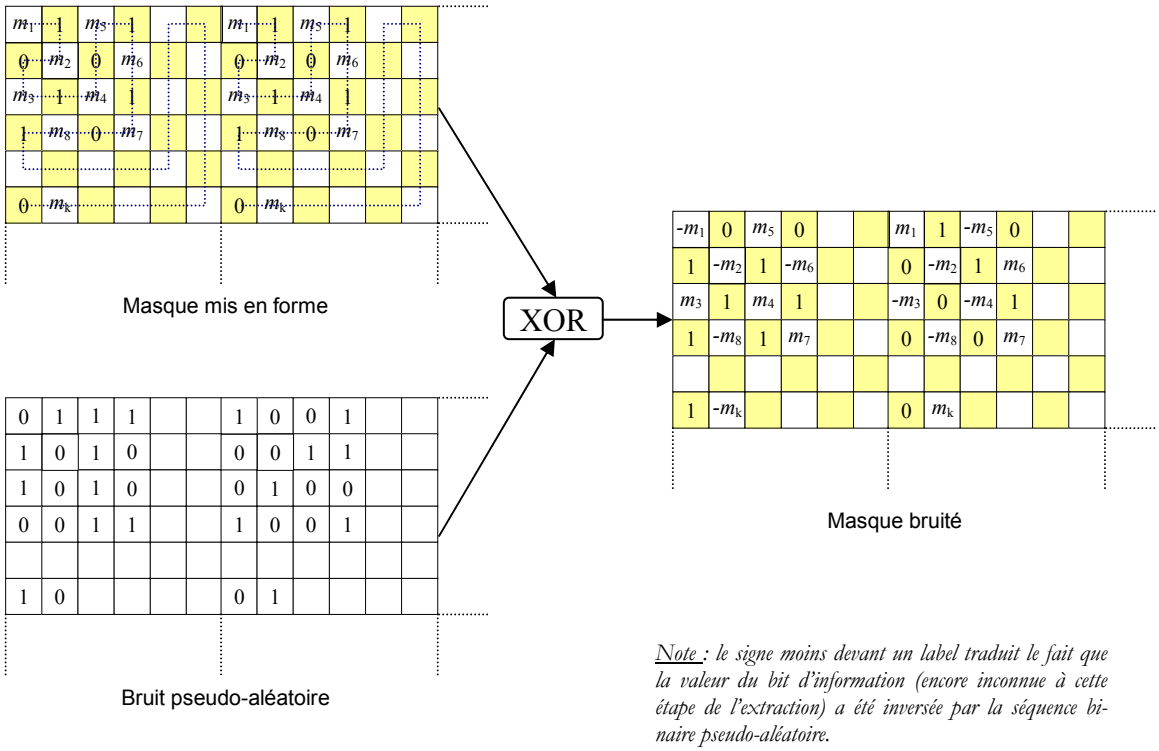


Figure 4.11 – *Création du masque de référence*

3.3.3. Estimation des déformations géométriques par calcul de flot optique

Dans le cas d'attaques géométriques, comme l'attaque Stirmark, on remarque que les déformations, bien que globalement aléatoires, sont quasiment linéaires sur des petites portions de l'image. De ce fait, il est alors possible de les approximer localement par de simples translations de blocs. Nous avons donc choisi d'utiliser un algorithme classique de « block matching » pour calculer le flot optique correspondant à l'attaque géométrique et compenser ainsi les déformations engendrées. L'algorithme de block matching est bien connu en codage vidéo où il est appliqué à la détection des mouvements entre deux trames. Dans le cadre de notre technique de resynchronisation, le block matching est appliqué entre le masque de référence et le support de tatouage extrait. Le support du tatouage a été préalablement séparé de l'image, seuillé afin d'éliminer les valeurs de fortes amplitudes, et démodulé pour ne faire apparaître que les valeurs des bits portés (*i.e.* « 0 », « 1 » ou « indéterminé »).

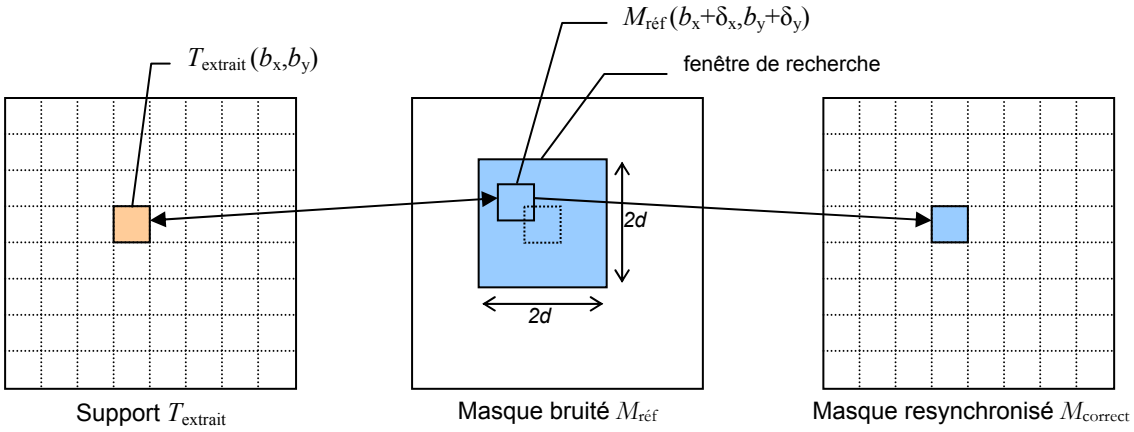


Figure 4.12 – *Processus de recherche par Block Matching*

Le principe du calcul du flot optique est le suivant. On considère le support ternaire du tatouage démodulé $T_{extrait}$ et le masque de référence $M_{réf}$. Le support du tatouage est divisé en blocs de taille $n \times n$ pixels. Pour chaque bloc $T_{extrait}(b_x,b_y)$ du support, on recherche un bloc $M_{réf}(b_x+\delta_x,b_y+\delta_y)$ de mêmes dimensions, dans le masque de référence, qui minimise une fonction de coût (fonction décrite au paragraphe 3.3.4). La recherche est effectuée exhaustivement à l'intérieur d'une fenêtre de recherche de taille $2d \times 2d$ pixels, centrée sur position initiale du bloc $T_{extrait}(b_x,b_y)$ (*c.f.* Figure 4.12). L'espace de recherche limite par conséquent le déplacement maximal à $\pm d$ pixels autour de la position de ce bloc. Pour chaque bloc $M_{réf}(b_x+\delta_x,b_y+\delta_y)$ candidat on calcule donc son score de pénalisation $S_{bx,by}(\delta_x, \delta_y)$. Le bloc $M_{réf}(b_x+\delta_{Xbest},b_y+\delta_{Ybest})$ qui obtient le plus faible score de pénalisation est considéré comme le meilleur candidat et son contenu est recopié dans un tableau, de mêmes dimensions que le support $T_{extrait}$, à la position du bloc de coordonnées (b_x,b_y). La déformation locale de l'image est alors approximée par la translation $(-\delta_{Xbest}, -\delta_{Ybest})$.

Une fois l'opération répétée pour tous les blocs du support, on obtient un nouveau masque de référence M_{correct} adapté à l'image manipulée. A la fin de ce processus, l'image attaquée n'a certes pas été redressée, mais l'on dispose désormais d'un modèle, permettant d'extraire les valeurs des bits d'information au bon endroit dans le support du tatouage.

3.3.4. Détermination de la fonction de coût

Afin de déterminer le meilleur critère d'appariement entre les blocs T_{extrait} (b_x,b_y) et $M_{\text{réf}}$ $(b_x+\delta_x,b_y+\delta_y)$, nous avons calculé pour chaque déplacement (δ_x, δ_y) le nombre de bons appariements (*i.e.* bit de contrôle resynchronisé), le nombre de mauvais appariements (*i.e.* bit de contrôle mis en correspondance avec sa valeur opposée), le nombre d'appariements indéfinis (*i.e.* bit de contrôle mis en correspondance avec une valeur indéterminée) ainsi que le nombre de bits de data dans le bloc de référence. Les résultats reportés dans la figure 4.13a (pour des blocs 64×64 et une fenêtre de recherche 88×88) montrent que seuls les bons et les mauvais appariements au niveau des bits de contrôle donnent une information sur la qualité de la resynchronisation. Le nombre d'appariements indéterminés est statistiquement constant quelque soit la position testée et n'influence donc pas le choix du meilleur bloc. Le nombre de bits d'information est également stable, ce qui est normal compte tenu de la distribution spatiale des bits de contrôle et des bits d'information.

La fonction de coût retenue consiste donc à faire le rapport du nombre de mauvais appariements sur le nombre de bons appariements (4.2).

$$S_{b_x,b_y}(\delta_x,\delta_y)=\frac{nb^{mauvais}_{b_x,b_y}(\delta_x,\delta_y)}{nb^{bons}_{b_x,b_y}(\delta_x,\delta_y)} \tag{4.2}$$

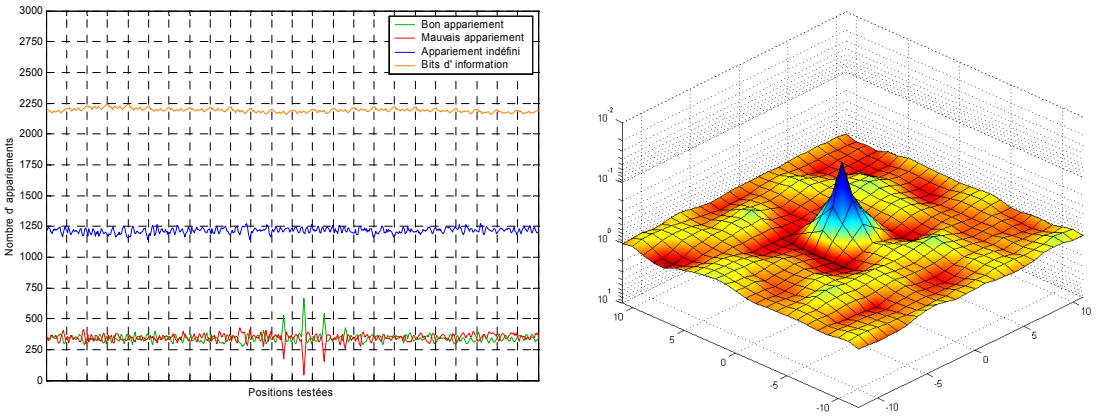
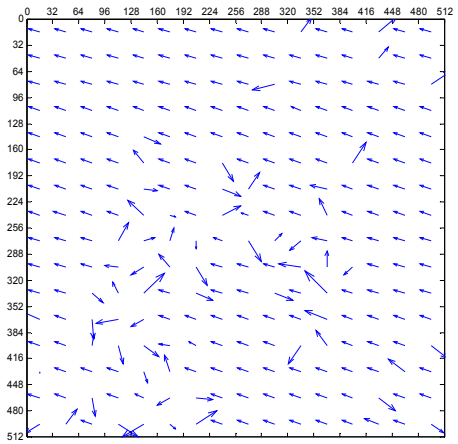


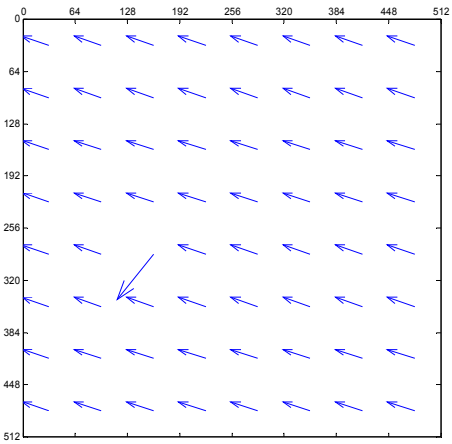
Figure 4.13 – *Appariements et score de pénalité en fonction des positions testées*

3.3.5. Efficacité de la resynchronisation en fonction de la taille des blocs

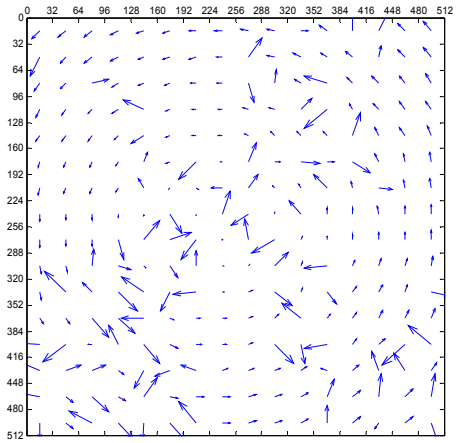
La taille des blocs et la dimension de la fenêtre de recherche doivent être choisies judicieusement car elles déterminent l'efficacité de l'algorithme de resynchronisation. Des blocs trop grands ne permettent pas de compenser les déformations géométriques locales de l'image, alors que des blocs trop petits ne contiennent pas assez de bits de contrôle pour estimer correctement ces déformations. La taille de la fenêtre de recherche, quant à elle, détermine l'amplitude maximale des déformations géométriques, ainsi que le coût en termes de temps de calcul de l'algorithme de recherche. La figure 4.14 illustre l'influence de la taille des blocs pour détecter des déformations géométriques ; dans le premier exemple l'image *Lena* a subi une translation, dans le deuxième une rotation de 1 degré et dans le dernier des déformations locales aléatoires (Stirmark). Nous avons déterminé expérimentalement que les meilleurs résultats étaient obtenus en prenant des blocs de taille 64×64.



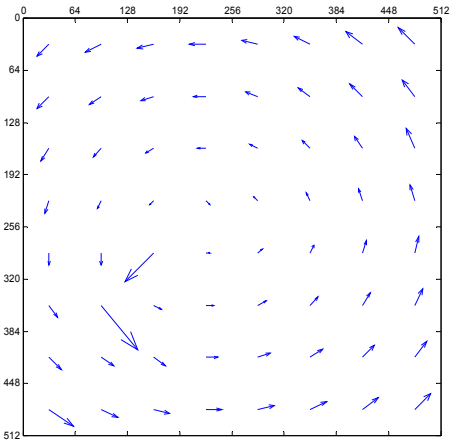
1.a) Translation $T(6,2)$ – blocs 32x32



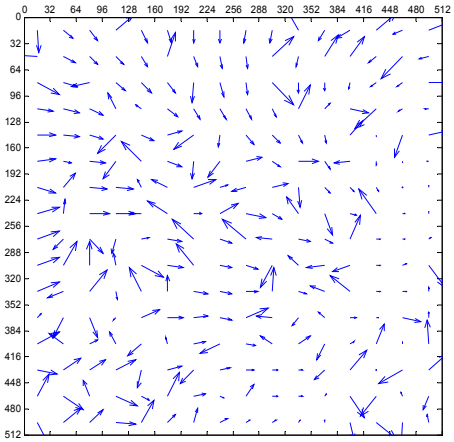
1.b) Translation $T(6,2)$ – blocs 64x64



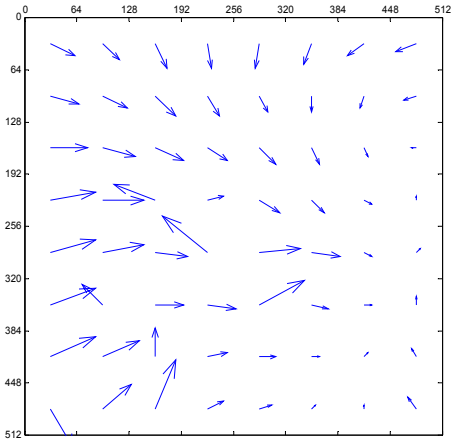
2.a) Rotation 1 degré – blocs 32x32



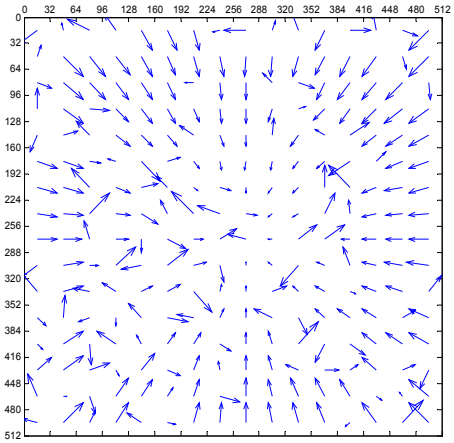
2.b) Rotation 1 degré – blocs 64x64



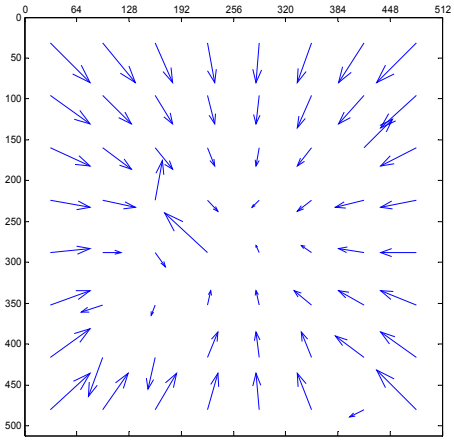
3.a) Stirmark – blocs 32x32



3.b) Stirmark – blocs 64x64



4.a) Zoom x1.05 – blocs 32x32

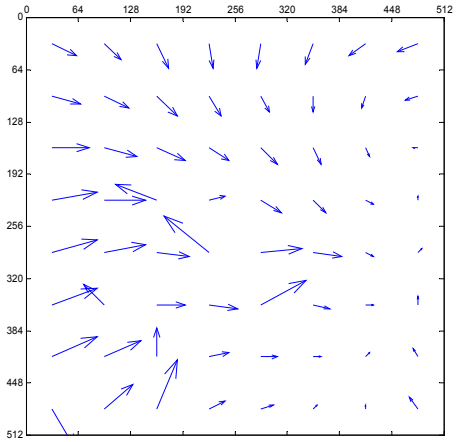


4.b) Zoom x1.05 – blocs 64x64

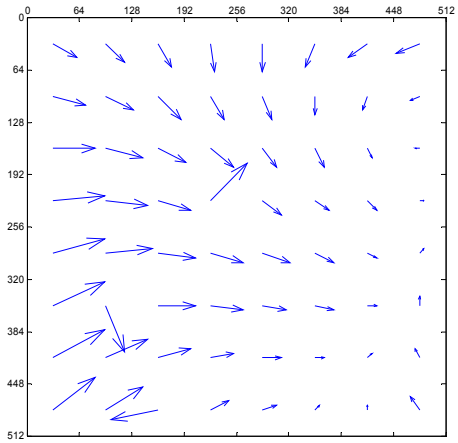
Figure 4.14 – *Flots optiques illustrant l'influence de la taille des blocs lors du block matching*

3.3.6. Affinement de la resynchronisation par un processus itératif

Face à certaines combinaisons d'attaques géométriques et photométriques, comme par exemple une légère rotation suivie d'une compression jpeg forte, l'algorithme de « block matching » a plus de difficultés pour apparier correctement les blocs du support avec ceux du masque. Le problème provient du fait que les attaques photométriques vont avoir tendance à atténuer en partie le signal de la marque cachée dans l'image. Ce qui se traduit au niveau du support du tatouage extrait par une augmentation significative du nombre de bits erronés ou indéterminés. En d'autres mots, le nombre de points d'accroche valides dans le support se trouve considérablement réduit, « faussant » ainsi le calcul du score de pénalisation. Afin de limiter ce phénomène et d'améliorer la resynchroni-



2.a) Stirmark – blocs 64x64 – 1 itération



2.b) Stirmark – blocs 64x64 – 2 itérations

Figure 4.15 – *Exemple d'affinement de la resynchronisation par le processus itératif*

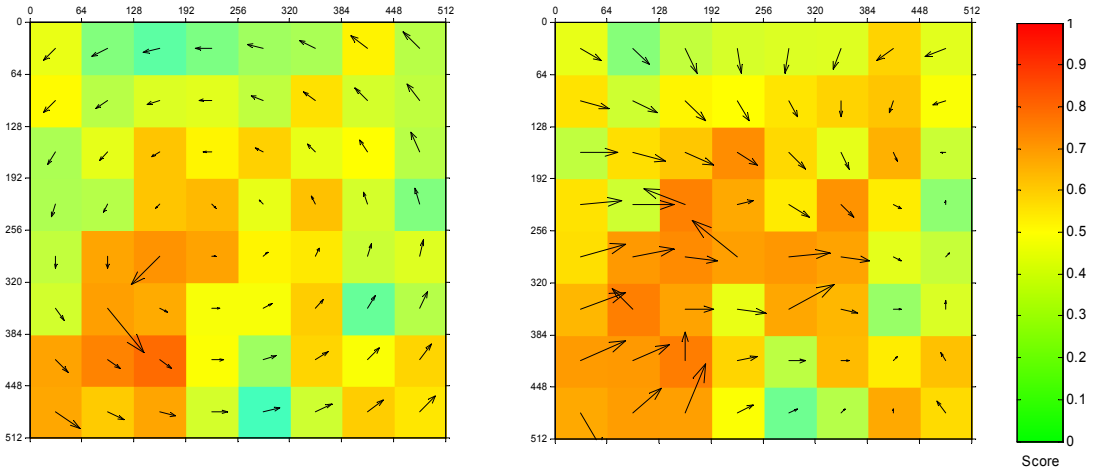
sation, nous avons mis en place un processus itératif. L'idée est de considérer, lors des itérations suivantes, les n meilleurs bits d'information comme des bits de contrôle pour la phase de recherche par « block matching ». De cette manière, on augmente le nombre d'éléments du support pris en compte dans le calcul du score d'appariement. Dans la pratique, lorsqu'il y a convergence, celle-ci est très rapide et en général une ou deux itérations sont suffisantes pour stabiliser le processus.

Processus de resynchronisation itératif

1. Block Matching
2. Reconstruction du message
3. Classement des bits d'information b_k en fonction du taux majoritaire S_k; (défini au chapitre 2, paragraphe 2.3.1)
4. Utilisation des n meilleurs bits d'information comme bits de contrôle supplémentaires ;
5. Itération du processus (critère d'arrêt : marque stable ou nombre maximum d'itérations atteint)

3.3.7. Détection des appariements de bloc incorrects et reconstruction du message

La reconstruction du message est réalisée à partir du support $T_{extrait}$ et du masque de référence $M_{correct}$ adapté et pondéré. Le principe de décodage des codes par répétition par vote majoritaire reste inchangé (*c.f.* chapitres 2 et 3). Néanmoins, il peut être judicieux de pondérer ou d'éliminer les éléments du support appartenant à un bloc $T_{extrait}(b_x,b_y)$ mal resynchronisé, de manière à dégager une plus forte majorité lors du décodage. Le score d'appariement $S_{bx,by}(\delta_{Xbest}, \delta_{Ybest})$ obtenu lors du processus de « block matching » donne une information qualitative sur la resynchronisation, mais n'est pas suffisant pour rejeter ou non le bloc considéré (*c.f.* Figure 4.16). Il est donc nécessaire,



a) Rotation 1 degré – blocs 64x64 b) Stirmark – blocs 64x64

Figure 4.16 – *Scores d'appariement par bloc*

dans le cas d'un « mauvais score », de vérifier en plus la cohérence locale du vecteur de mouvement $V_{bx,by}$ du bloc avec ceux des blocs voisins. Cela revient à comparer le vecteur $V_{bx,by}$ avec une estimation obtenue à partir d'un modèle géométrique affine local (4.3). Les paramètres du modèle sont déterminés en minimisant l'équation (4.4) par la méthode des moindres carrés.

$$\begin{aligned} d_x(x_i, y_i) &= a_1 + a_2.x_i + a_3.y_i \\ d_y(x_i, y_i) &= a_4 + a_5.x_i + a_6.y_i \end{aligned} \tag{4.3}$$

$$E_i^2 = \sum_i \left(\left(a_1 + a_2.x_i + a_3.y_i - d_x(x_i, y_i)\right)^2 + \left(a_4 + a_5.x_i + a_6.y_i - d_y(x_i, y_i)\right)^2 \right) \tag{4.4}$$

$$\begin{cases} \frac{1}{2}\frac{\partial E_i^2}{\partial a_1} = 0 \Rightarrow a_1 \sum_i 1 + a_2 \sum_i x_i + a_3 \sum_i y_i = \sum_i d_x(x_i, y_i) \\ \frac{1}{2}\frac{\partial E_i^2}{\partial a_1} = 0 \Rightarrow a_1 \sum_i x_i + a_2 \sum_i x_i^2 + a_3 \sum_i x_i.y_i = \sum_i x_i.d_x(x_i, y_i) \\ \frac{1}{2}\frac{\partial E_i^2}{\partial a_1} = 0 \Rightarrow a_1 \sum_i y_i + a_2 \sum_i x_i.y_i + a_3 \sum_i y_i^2 = \sum_i y_i.d_x(x_i, y_i) \\ \frac{1}{2}\frac{\partial E_i^2}{\partial a_1} = 0 \Rightarrow a_4 \sum_i 1 + a_5 \sum_i x_i + a_6 \sum_i y_i = \sum_i d_y(x_i, y_i) \\ \frac{1}{2}\frac{\partial E_i^2}{\partial a_1} = 0 \Rightarrow a_4 \sum_i x_i + a_5 \sum_i x_i^2 + a_6 \sum_i x_i.y_i = \sum_i x_i.d_y(x_i, y_i) \\ \frac{1}{2}\frac{\partial E_i^2}{\partial a_1} = 0 \Rightarrow a_4 \sum_i y_i + a_5 \sum_i x_i.y_i + a_6 \sum_i y_i^2 = \sum_i y_i.d_y(x_i, y_i) \end{cases} \tag{4.5}$$

$$\begin{cases} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} = \begin{pmatrix} \sum_i 1 & \sum_i x_i & \sum_i y_i \\ \sum_i x_i & \sum_i x_i^2 & \sum_i x_i.y_i \\ \sum_i y_i & \sum_i x_i.y_i & \sum_i y_i^2 \end{pmatrix}^{-1} \begin{pmatrix} \sum_i d_x(x_i, y_i) \\ \sum_i x_i.d_x(x_i, y_i) \\ \sum_i y_i.d_x(x_i, y_i) \end{pmatrix} \\ \begin{pmatrix} a_4 \\ a_5 \\ a_6 \end{pmatrix} = \begin{pmatrix} \sum_i 1 & \sum_i x_i & \sum_i y_i \\ \sum_i x_i & \sum_i x_i^2 & \sum_i x_i.y_i \\ \sum_i y_i & \sum_i x_i.y_i & \sum_i y_i^2 \end{pmatrix}^{-1} \begin{pmatrix} \sum_i d_y(x_i, y_i) \\ \sum_i x_i.d_y(x_i, y_i) \\ \sum_i y_i.d_y(x_i, y_i) \end{pmatrix} \end{cases} \tag{4.6}$$

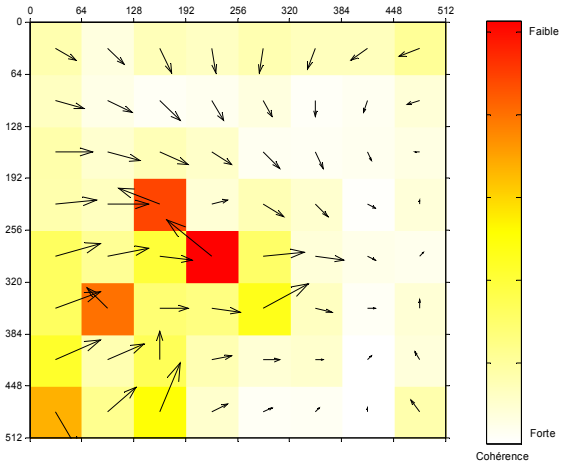
Dans la pratique, le modèle affine local est estimé à partir des vecteurs de mouvement du voisinage immédiat, ce qui correspond à seulement 8 observations. Cette restriction sur la dimension du voisinage est liée à la taille des blocs utilisés lors du processus de « block matching », ainsi qu'à l'hypothèse de travail qui stipule que les déformations géométriques aléatoires ne peuvent être approchées par des transformations affines que localement. Les paramètres (a_1, a_2, a_3, a_4, a_5, a_6) du modèle sont donc obtenus en résolvant le système (4.7) ci-dessous :

$$\begin{cases} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} = \begin{pmatrix} -1/6 & -1/6 & 11/24 \\ 0 & 1/6 & -1/6 \\ 1/6 & 0 & -1/6 \end{pmatrix} . \begin{pmatrix} \sum_{i=0}^{7} d_x(x_i, y_i) \\ \sum_{i=0}^{7} x_i.d_x(x_i, y_i) \\ \sum_{i=0}^{7} y_i.d_x(x_i, y_i) \end{pmatrix} \\ \begin{pmatrix} a_4 \\ a_5 \\ a_6 \end{pmatrix} = \begin{pmatrix} -1/6 & -1/6 & 11/24 \\ 0 & 1/6 & -1/6 \\ 1/6 & 0 & -1/6 \end{pmatrix} . \begin{pmatrix} \sum_{i=0}^{7} d_y(x_i, y_i) \\ \sum_{i=0}^{7} x_i.d_y(x_i, y_i) \\ \sum_{i=0}^{7} y_i.d_y(x_i, y_i) \end{pmatrix} \end{cases} \tag{4.7}$$

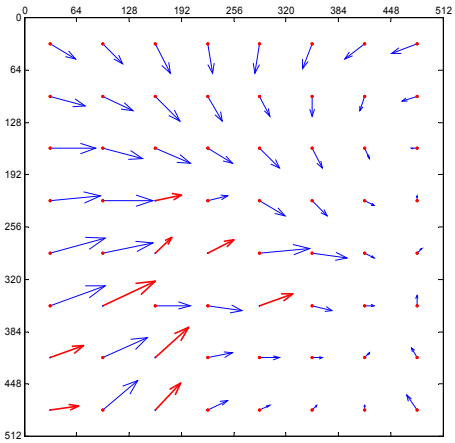
Un vecteur $V_{bx,by}$ est alors jugé incohérent si l'erreur quadratique (4.8) entre ce vecteur et son estimation à partir du flot optique est supérieure à un seuil $\delta_{cohérence}$.

$$Cohérence_{bx,by} = \left\| V_{bx,by} - \hat{V}_{bx,by} \right\|^2 \tag{4.8}$$

Les vecteurs incohérents peuvent être remplacés par leurs estimations respectives, ou tout simplement « éliminés » du flot optique. Cette modification se répercute bien évidement au niveau du masque $M_{correct}$ par une mise à jour des blocs concernés.



flot optique « brut » et cohérence locale des vecteurs de mouvement après BM



flot optique après détection et correction des « mauvais » appariements

Figure 4.17 – *Détection et correction des appariements de bloc incorrects*

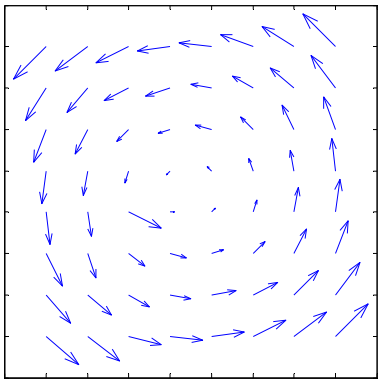
Dans l'exemple illustré par la figure 4.17, les vecteurs de mouvement incohérents avec le flot optique ont été remplacés par leurs estimations respectives (flèches rouges).

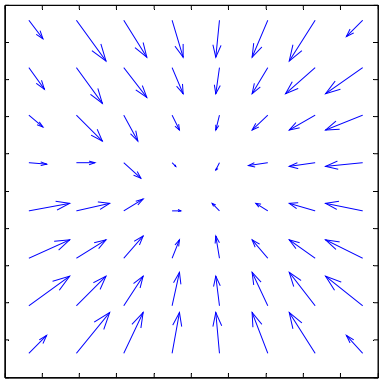
3.4. Résultats expérimentaux

3.4.1. Déformations géométriques compensées par la méthode

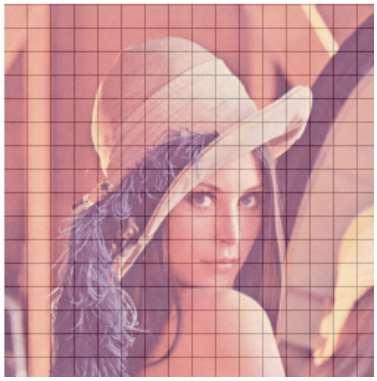
Nous avons évalué l'efficacité de notre technique de resynchronisation face à différents types de déformations géométriques de faible amplitude. La figure 4.18 illustre quelques unes des attaques géométriques appliquées à l'image (rotation, translation, zoom, stirmark, cisaillement, etc.), ainsi que la manière dont elles ont été compensées par le processus de resynchronisation.

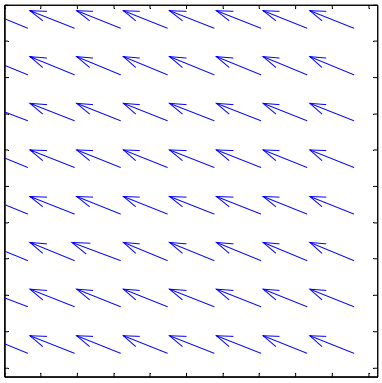
a) rotation de 2 degrés



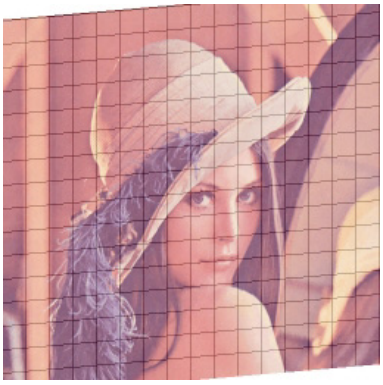


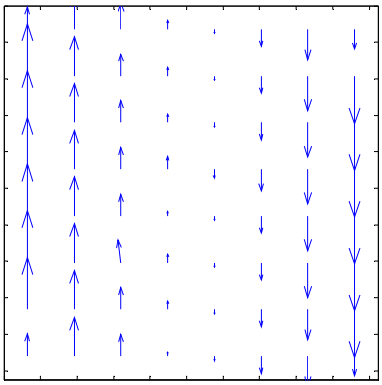
b) zoom x1.05



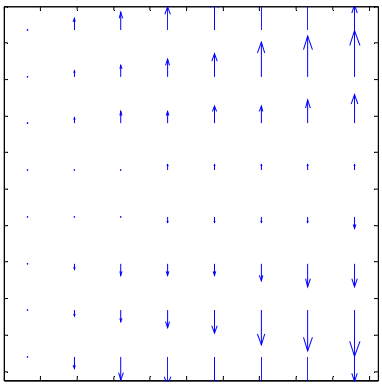


c) translation





d) cisaillement vertical de 5 degrés



e) effet de perspective

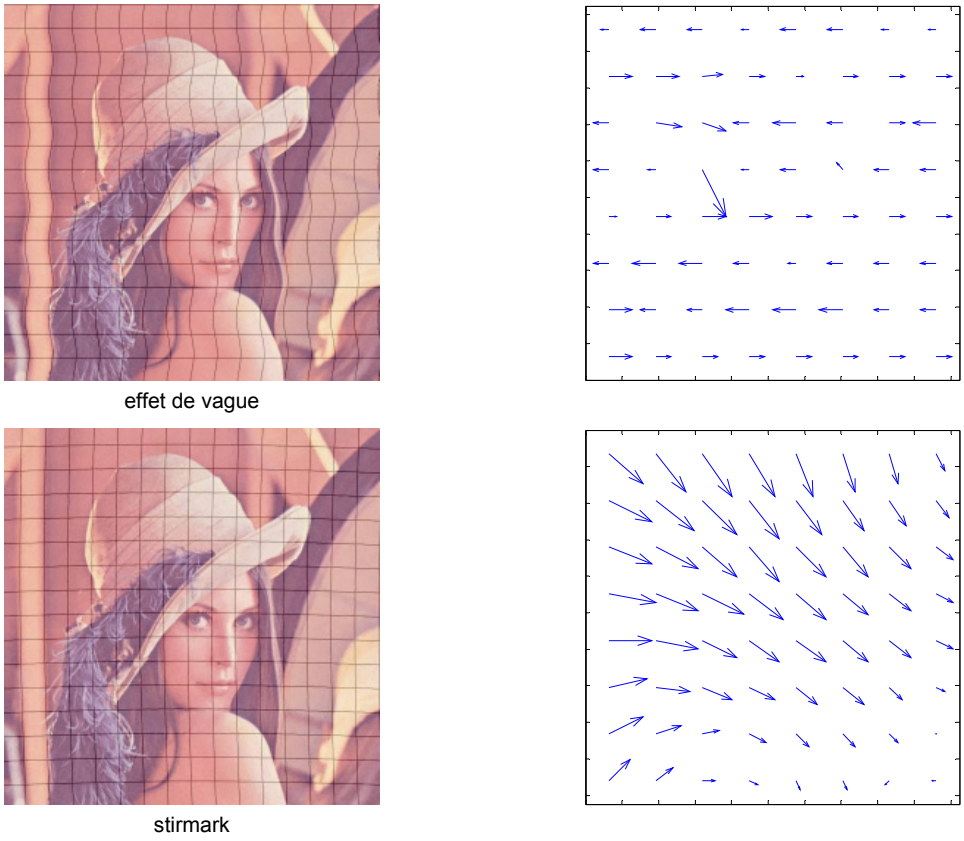


Figure 4.18 – *Exemples d'attaques géométriques compensées par le processus de resynchronisation*

3.4.2. Performances en termes de robustesse du tatouage

Bien que le processus de resynchronisation parvienne à estimer correctement les déformations géométriques appliquées à une image (*c.f.* figure 4.18), l'impact de ce type d'attaque sur le tatouage n'en demeure pas moins significatif. Cela s'explique d'une part par les approximations faites par l'algorithme de « block matching », les bits d'information d'un même bloc ne sont pas tous parfaitement resynchronisés, et d'autre part par les effets de bord liés à l'attaque géométrique elle-même (*e.g.* interpolation bilinéaire, calcul de l'attracteur biaisé, etc.). En effet, la géométrie de l'image étant différente, les appariements de blocs réalisés lors du calcul de l'attracteur ne seront plus les mêmes ce qui aura pour conséquence directe d'augmenter de manière significative le taux d'erreur binaire au niveau du support extrait.

La figure 4.19 illustre les performances réelles de notre algorithme de tatouage face à l'attaque Stirmark (*i.e.* déformations géométriques aléatoires + compression JPEG 75%) en fonction du nombre de bits d'information. A titre d'exemple, pour un message de 1000 bits, plus de 90% des bits d'information sont extraits sans erreur après resynchronisation.

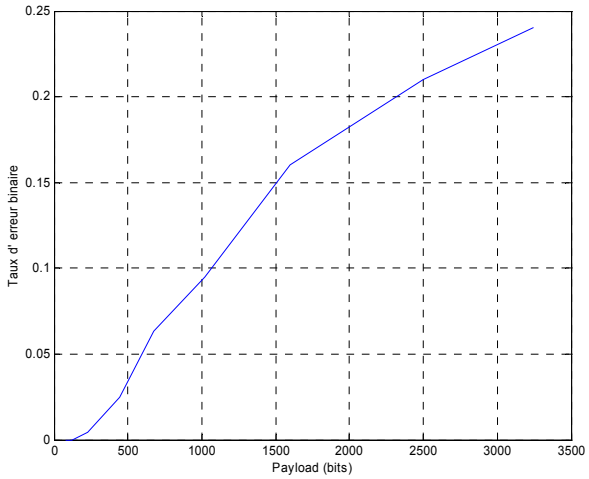


Figure 4.19 – *Taux d'erreur binaire en fonction du nombre de bits cachés vis-à-vis de l'attaque Stirmark*

3.5. Limites de la méthode

La méthode de resynchronisation proposée s'avère très efficace pour compenser des déformations géométriques locales de faible amplitude telles que celles qui peuvent être engendrées par l'attaque Stirmark ou par une rotation de quelques degrés. D'une manière générale, la méthode est capable de contrebalancer toutes les déformations qui peuvent être approchées localement par des translations de blocs. D'un point de vue pratique, la méthode atteint ses limites uniquement lorsque que l'image subie de grandes transformations affines (*e.g.* rotations supérieures à 3 degrés, translations plus grandes que la fenêtre de recherche, etc.), ou des déformations géométriques locales fortes. Cependant, dans la pratique, il faut également tenir compte du contenu de l'image. En effet, le tatouage n'est pas inséré de manière uniforme dans l'image, et la résistance de celui-ci ne va pas être la même suivant le contenu fréquentiel de la région considérée. Typiquement, lorsque qu'une région est très texturée, le tatouage est modulé avec un support composé essentiellement de hautes fréquences, ce qui le rend localement moins robuste aux filtrages passe-bas, ainsi qu'aux interpolations bilinéaires. Globalement le tatouage est extrait sans erreur grâce à une redondance importante, mais localement les bits d'information, comme les bits de resynchronisation, peuvent avoir un taux d'erreur proche de cinquante pour cent, faussant complètement le « block matching ». Dans de telles conditions le processus de resynchronisation est localement inefficace faute de repères fiables. Pour la majorité des images, ces problèmes locaux sont relativement peu nombreux et ils peuvent être détectés et éliminés grâce à des post-traitements. Cependant, dans le cas d'images très texturées ou lorsque l'on combine une déformation géométrique avec une forte attaque photométrique, il arrive que le support extrait contienne trop d'éléments erronés pour permettre une resynchronisation efficace.

4. Détection des transformations linéaires globales

4.1. Principe général

Nous présentons, dans la suite de ce chapitre, une méthode de resynchronisation, complémentaire de la première, permettant de détecter et de compenser les transformations affines globales de l'image. L'idée de base de la méthode consiste à insérer un motif répétitif carré au niveau du support, et d'analyser ensuite à l'extraction la position des pics correspondants au motif dans le domaine fréquentiel (*c.f.* Figure 4.20). Ces pics ont une topographie très particulière puisqu'ils correspondent aux points d'intersection d'une grille uniforme. Connaissant cette grille, il devient relativement aisé de déterminer les paramètres de la transformation affine duale dans le domaine fréquentiel.

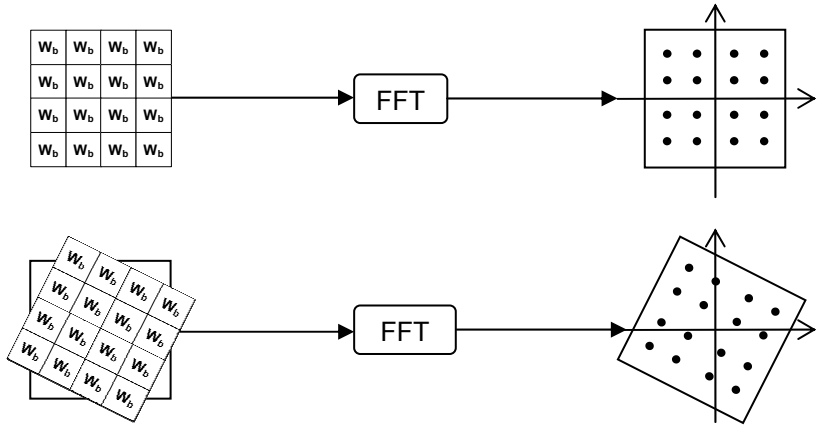


Figure 4.20 – *Principe général de la méthode de resynchronisation à l'aide d'un motif périodique*

Une transformation géométrique affine peut s'écrire d'une manière générale sous la forme matricielle suivante :

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = A.\begin{pmatrix} x \\ y \end{pmatrix} + \vec{T} = \begin{pmatrix} a & b \\ c & d \end{pmatrix}.\begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} e \\ f \end{pmatrix} \tag{4.9}$$

où la matrice A représente les paramètres de l'application linéaire et T le vecteur de translation. L'objectif de la méthode de resynchronisation est donc d'estimer les paramètres $\{a, b, c, d\}$ de la matrice A. Dans notre cas, nous considérons le repère orthonormé ayant pour origine le centre de l'image, et le vecteur de translation T égal au vecteur nul.

4.2. Application linéaire fréquentielle duale

L'estimation de l'application linéaire A_s appliquée à l'image est réalisée directement dans le domaine de Fourier. Plus exactement, on estime à partir des déformations des points de la grille la

transformation linéaire duale A_f dans le plan des fréquences. Nous montrons que la relation qui lie les applications A_s et A_f est la suivante :

$$A_f = \left(A_s^T\right)^{-1} \tag{4.10}$$

La transformée de Fourier $F(u,v)$ de $f(x,y)$ s'écrit

$$F(u,v) = \iint f(x,y) e^{-j(ux+vy)} \mathrm{d}x\mathrm{d}y \tag{4.11}$$

Soit $g(x,y)$ l'image de $f(x,y)$ par une application linéaire

$$g(x,y) = f(ax+by, cx+dy) \tag{4.12}$$

La transformée $G(u,v)$ de $g(x,y)$ s'écrit donc

$$\begin{aligned} G(u,v) &= \iint g(x,y) e^{-j(ux+vy)} \mathrm{d}x\mathrm{d}y \\ G(u,v) &= \iint f(ax+by, cx+dy) e^{-j(ux+vy)} \mathrm{d}x\mathrm{d}y \end{aligned} \tag{4.13}$$

En faisant le changement de variable

$$\begin{cases} X = ax+by \\ Y = cx+dy \end{cases} \Leftrightarrow \begin{cases} x = \dfrac{1}{\Delta}(dX - bY) \\ y = \dfrac{1}{\Delta}(-cX + aY) \end{cases} \qquad \text{avec} \quad \Delta = ad - bc \neq 0 \tag{4.14}$$

On obtient

$$\begin{aligned} G(u,v) &= \iint f(X,Y) e^{-\frac{j}{\Delta}((du-cv)X+(-bu+av)Y)} \left|\mathrm{J}(\mathrm{X},\mathrm{Y})\right| \mathrm{d}X\mathrm{d}Y \\ G(u,v) &= \iint \frac{1}{\Delta} f(X,Y) e^{-\frac{j}{\Delta}((du-cv)X+(-bu+av)Y)} \mathrm{d}X\mathrm{d}Y \\ G(u,v) &= \frac{1}{\Delta} F\left(\frac{du-cv}{\Delta}, \frac{-bu+av}{\Delta}\right) \end{aligned} \tag{4.15}$$

4.3. Insertion

Les modifications à apporter à l'étape d'insertion de l'algorithme de tatouage sont minimes, puisqu'il suffit simplement d'insérer, en plus de la marque, un motif périodique carré dans l'image. Il existe cependant différentes manières de créer un tel motif. Voici quelques unes des solutions envisageables :

Solution 1 : la façon la plus simple d'opérer consiste à utiliser le tatouage lui-même comme motif périodique. Pour cela, il est nécessaire de changer le modèle de cryptage en ne faisant plus un cryptage global du tatouage, mais simplement en appliquant le XOR avant la phase de duplication. Cette solution est très simple à mettre en œuvre et offre l'avantage de ne pas introduire de distorsion visuelle supplémentaire. Par contre, le tatouage n'étant plus crypté de manière globale, il n'y a plus de garantie sur une distribution homogène des « 0 » et des « 1 ». De ce fait, le support modulé ne sera plus à moyenne nulle ce qui entraîne une instabilité de l'attracteur à l'extraction, et par conséquent une diminution de la robustesse. D'autre part, cela ouvre également une faille pour des attaques cryptographiques telles que des collusions de type I appliquées sur les différentes duplications du tatouage.

Solution 2 : une variante de la première solution, consiste à ne crypter localement que les bits de resynchronisation, les bits d'information restant quant à eux cryptés de manière globale. En procédant ainsi, on élimine les principaux défauts de la solution précédente. Par contre, étant donné que seule une partie du motif (*i.e.* les bits de resynchronisation) est périodique, les pics correspondants dans le domaine fréquentiel auront une amplitude plus faible et seront par conséquent plus difficiles à détecter.

Solution 3 : une alternative à l'utilisation du tatouage comme motif de resynchronisation, consiste à insérer un « template » fixe. L'idée est de générer un bruit gaussien périodique à moyenne nulle et de l'ajouter directement au support. Cette solution à pour inconvénient majeur d'introduire une distorsion visuelle supplémentaire, plus ou moins importante suivant la variance du bruit utilisé.

4.4. Resynchronisation

Quelle que soit la solution adoptée à l'insertion sur le choix du motif périodique, le processus de resynchronisation est le même dans son principe et se décompose en 4 étapes :

- Détection des pics dans le domaine fréquentiel,
- Estimation des deux directions principales par la transformée de Hough,
- Estimation du facteur d'échelle suivant ces deux directions,
- Application de la transformation linéaire inverse.

4.4.1. Détection des pics

La première étape du processus de resynchronisation consiste à extraire, dans le domaine fréquentiel, les pics correspondants au motif périodique. Pour cela, on calcule la transformée de Fourier 2D du support I_{supp} duquel on a préalablement éliminé les éléments d'amplitude supérieure au seuil δ_h (dans le cas bien évidemment où l'on utilise le tatouage comme motif périodique). On ap-

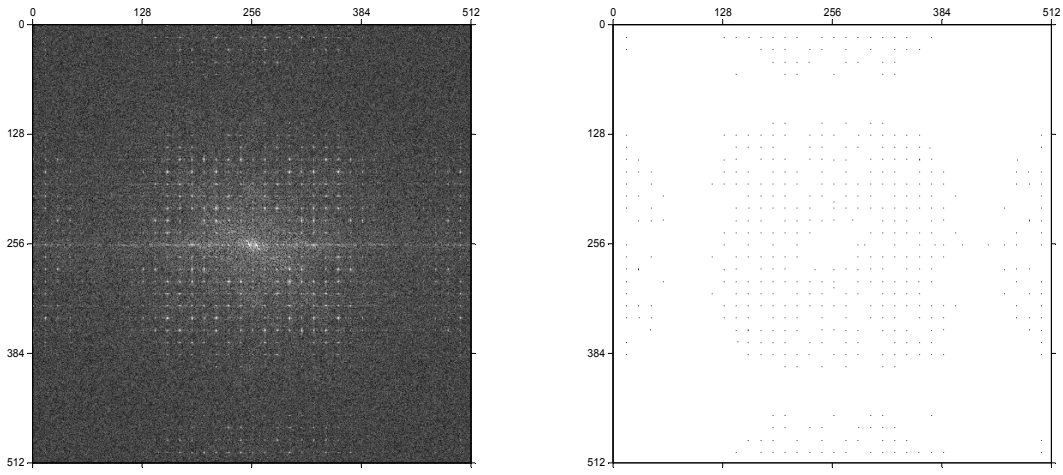


Figure 4.21 – *Extraction des pics dans le domaine fréquentiel*

plique ensuite un filtre fréquentiel passe-haut afin d'éliminer les très basses fréquences ainsi que la composante continue. On effectue enfin un seuillage de manière à extraire uniquement les pics correspondant à la grille (*c.f.* Figure 4.21). La qualité du seuillage est primordiale car elle conditionne directement l'efficacité du processus de resynchronisation. En effet, si lors du seuillage trop peu de pics sont détectés correctement ou si beaucoup de points parasites sont extraits en plus des pics, l'estimation de la grille déformée risque d'être biaisé, rendant alors la resynchronisation impossible.

La détection des pics est réalisée localement en recherchant les maxima locaux supérieurs à un seuil de détection δ_{pic}. Le seuil de détection est déterminé en fonction de la moyenne et de l'écart type local. La taille de la fenêtre de recherche doit être choisie de telle sorte qu'elle ne puisse pas contenir plusieurs pics potentiels.

$$F(u,w)\text{ est un pic} \Leftrightarrow \begin{cases} F(u,w) = Max(F_x(u,w)) \\ F(u,w) \geq \delta_{pic} \quad \text{avec } \delta_{pic} = \mu(u,w) + \alpha.\sigma_x(u,w) \end{cases} \tag{4.16}$$

4.4.2. Estimation des directions principales

L'estimation des directions principales formées par alignements des points de la grille est réalisée à l'aide de la transformée de Hough. Celle-ci permet de détecter des formes simples et facilement caractérisables, telles que des droites. Le principe de la détection est relativement simple. Pour caractériser de manière unique une droite dans l'espace de départ il suffit de 2 points (x_1,y_1) et (x_2,y_2). A partir de ces deux points, on déterminer le couple (ρ,θ) correspondant dans l'espace

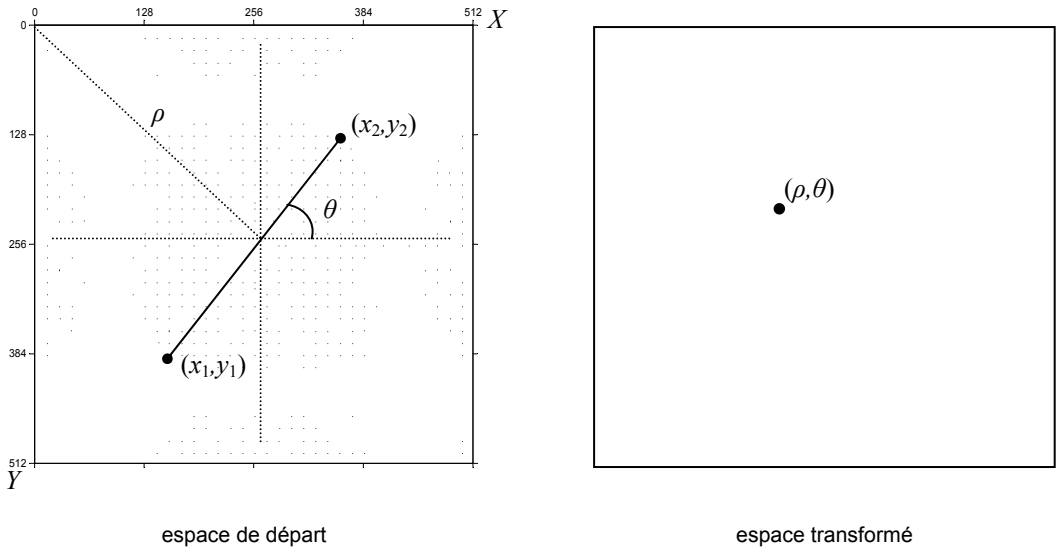


Figure 4.22 – *Relation entre les deux espaces*

transformé à partir des relations suivantes :

$$\theta = \arctan\left(\frac{y_1 - y_2}{x_1 - x_2}\right)$$

$$\rho = \frac{x_1.y_2 - x_2.y_1}{\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}} \tag{4.17}$$

Pour chaque couple de points (x_1,y_1) et (x_2,y_2) on incrémente alors le compteur correspondant au couple (ρ,θ) dans l'espace d'arrivée discrétisé (*c.f.* Figure 4.22), les paramètres ρ et θ prenant leurs valeurs respectives dans $[-max_\rho, max_\rho]$, avec $\max_\rho = \sqrt{X^2 + Y^2}$, et $]-\pi/2, \pi/2[$. Les deux directions principales sont alors déterminées en sommant pour chaque orientation θ le nombre de droites détectées, et en recherchant ensuite les angles associés aux deux plus grandes sommations (*c.f.* Figure 4.23). Dans l'exemple de la figure 4.23, l'image n'a subi aucune déformation géométrique ; les axes de la grille ont été correctement détectés avec pour orientation respective -90 et 0 degrés, on remarque également que les 3ème et 4ème orientations correspondent aux diagonales de la grille.

4.4.3. Estimation du facteur d'échelle suivant les deux directions principales

Une fois les axes principaux de la grille déterminés, il reste à estimer le facteur d'échelle suivant ces deux directions. Cette opération revient à estimer l'écartement entre les points le long de chaque axe. Une façon de modéliser ce problème est de rechercher pour chaque direction θ, la période t_θ

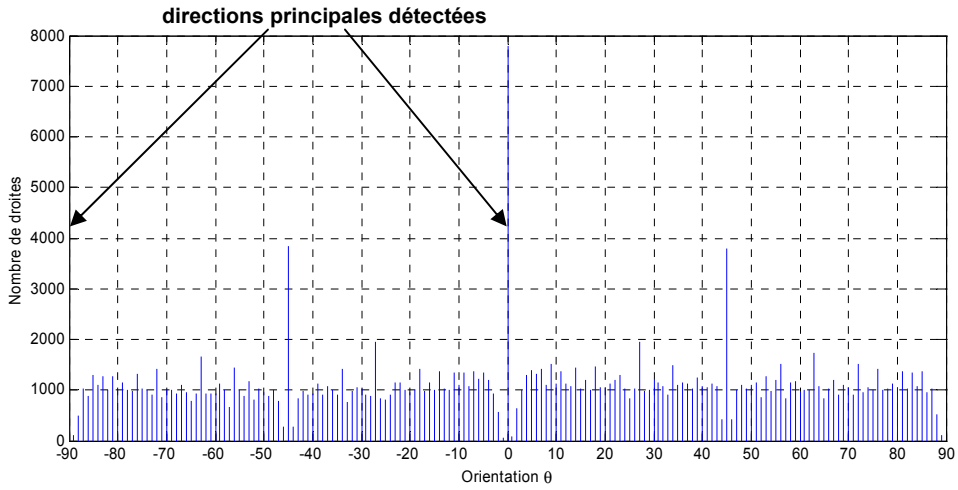


Figure 4.23 – *Nombre de droites détectées en fonction de la direction θ*

qui maximise l'équation (4.18), ou en d'autres termes trouver le peigne de Dirac qui correspond le mieux aux observations.

$$y(t_\theta) = \sum_{k=0}^{N} \left((1-a).f_\theta(\lfloor x \rfloor) + a.f_\theta(\lfloor x \rfloor + 1) \right) \tag{4.18}$$

avec : $x = k.t_\theta$ $\quad t_\theta \in [T_0 - \delta_t, T_0 + \delta_t]$ $\quad T_0$ = période de référence (*i.e.* sans déformation)

$a = x - \lfloor x \rfloor$

$f_\theta(\lfloor x \rfloor)$ = nombre de segments de direction θ dont la longueur est égale $\lfloor x \rfloor$, -1 si le nombre de segments est inférieur à une valeur $\delta_{\text{nb\_seg\_min}}$ (ce seuillage est nécessaire afin d'éviter de détecter une période qui soit une fraction de la période recherchée t_θ).

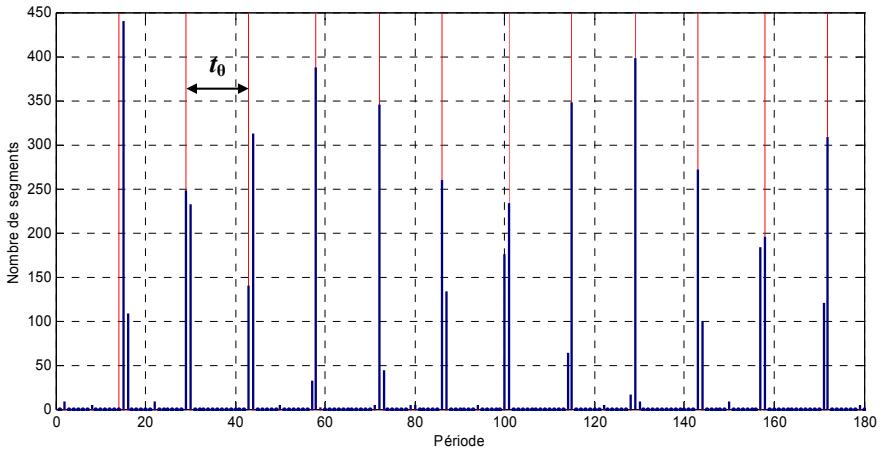


Figure 4.24 – *Estimation de la période suivant la direction θ*

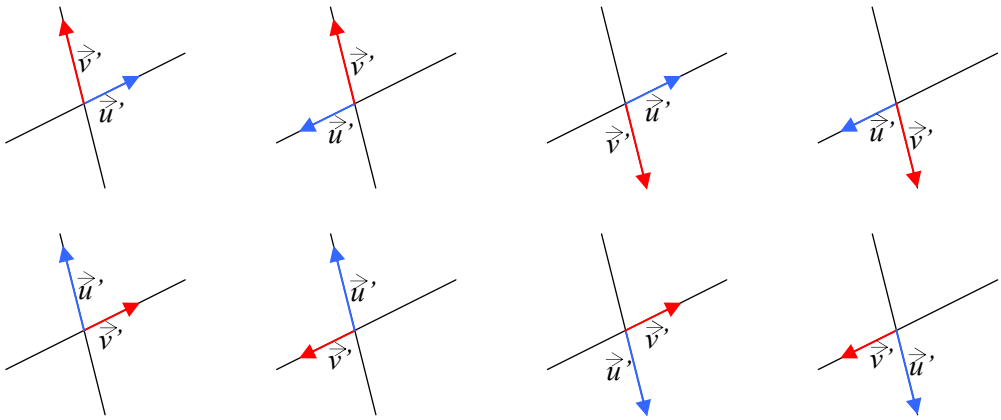


Figure 4.25 – *Illustration des ambiguïtés sur la direction et le sens des vecteurs u'et v'*

4.4.4. Détermination des paramètres de la matrice $\hat{A}_f$

Soit u et v deux vecteurs orthogonaux, définissant le repère de la grille dans le domaine fréquentiel :

$$\vec{u} = \begin{pmatrix} \|\vec{u}\| \\ 0 \end{pmatrix} \qquad \vec{v} = \begin{pmatrix} 0 \\ \|\vec{v}\| \end{pmatrix} \tag{4.19}$$

Leurs images respectives u', v' par l'application linéaire duale A_f sont données par les équations (4.20) et (4.21).

$$\vec{u}' = \begin{pmatrix} x_{u'} \\ y_{u'} \end{pmatrix} = A_f \cdot \begin{pmatrix} \|\vec{u}\| \\ 0 \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \cdot \begin{pmatrix} \|\vec{u}\| \\ 0 \end{pmatrix} = \begin{pmatrix} a.\|\vec{u}\| \\ c.\|\vec{u}\| \end{pmatrix} \tag{4.20}$$

$$\vec{v}' = \begin{pmatrix} x_{v'} \\ y_{v'} \end{pmatrix} = A_f \cdot \begin{pmatrix} 0 \\ \|\vec{v}\| \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \cdot \begin{pmatrix} 0 \\ \|\vec{v}\| \end{pmatrix} = \begin{pmatrix} b.\|\vec{v}\| \\ d.\|\vec{v}\| \end{pmatrix} \tag{4.21}$$

Des équations (4.20) et (4.21) on en déduit facilement les paramètres de la matrice A_f :

$$A_f = \begin{pmatrix} \frac{x_{u'}}{\|\vec{u}\|} & \frac{x_{v'}}{\|\vec{v}\|} \\ \frac{y_{u'}}{\|\vec{u}\|} & \frac{y_{v'}}{\|\vec{v}\|} \end{pmatrix} \tag{4.22}$$

A partir de la grille obtenue à l'extraction, il est possible d'estimer la direction et la norme des vecteurs des directions principales, mais la direction et le sens des vecteurs u', v' restent ambigus (*c.f.* Figure 4.25). Cette ambiguïté donne lieu à plusieurs applications affines $A_{f,i}$ possibles. Ces applications s'écrivent sous la forme générique $A_{f,i}=A_f.H_i$ où H_i peut être l'une des 8 isométries suivantes : l'identité H_1, la symétrie horizontale H_2, la symétrie verticale H_3, la symétrie centrale H_4, la rotation H_5 de $-\pi/2$, la rotation H_6 de $\pi/2$, et les réflexions H_7 et H_8 par rapport aux droites $y = x$ et $y = -x$.

$$H_1=\begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix} \qquad H2=\begin{pmatrix}1 & 0\\ 0 & -1\end{pmatrix} \qquad H_3=\begin{pmatrix}-1 & 0\\ 0 & 1\end{pmatrix} \qquad H_4=\begin{pmatrix}-1 & 0\\ 0 & -1\end{pmatrix}$$

$$H_5=\begin{pmatrix}0 & 1\\ -1 & 0\end{pmatrix} \qquad H_6=\begin{pmatrix}0 & -1\\ 1 & 0\end{pmatrix} \qquad H_7=\begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix} \qquad H_8=\begin{pmatrix}0 & -1\\ -1 & 0\end{pmatrix}$$

Nous attribuons donc arbitrairement la direction θ_1 au vecteur u' et la direction θ_2 au vecteur v', les normes respectives des vecteurs u', v' correspondent aux écartements des points détectés $t_{\theta 1}$ et $t_{\theta 2}$ suivant ces deux directions. Les normes des vecteurs u, v sont toutes deux égales à T_0. Les paramètres estimés de l'application linéaire $\hat{A}_f$ sont donnés par l'équation (4.23).

$$\hat{A}_f=\begin{pmatrix}\frac{\|\vec{u}'\|}{\|\vec{u}\|}\cos\theta_1 & \frac{\|\vec{v}'\|}{\|\vec{v}\|}\cos\theta_2\\ \frac{\|\vec{u}'\|}{\|\vec{u}\|}\sin\theta_1 & \frac{\|\vec{v}'\|}{\|\vec{v}\|}\sin\theta_2\end{pmatrix}=\begin{pmatrix}\frac{t_{\theta_1}}{T_0}\cos\theta_1 & \frac{t_{\theta_2}}{T_0}\cos\theta_2\\ \frac{t_{\theta_1}}{T_0}\sin\theta_1 & \frac{t_{\theta_2}}{T_0}\sin\theta_2\end{pmatrix} \tag{4.23}$$

4.4.5. Application de la transformation linéaire inverse

Dans la pratique, compte tenu du caractère discret des images numériques, on applique généralement la transformation linéaire inverse de l'image cible vers l'image source, plutôt que la transformation de l'image source vers l'image cible. Cela signifie que l'on recherche pour chaque pixel de l'image d'arrivée son antécédent dans l'image d'origine. En procédant ainsi, tous les pixels de l'image d'arrivée se trouvent renseignés, ce qui n'aurait pas été forcément le cas en faisant l'opération inverse. La dernière étape du processus de resynchronisation consiste donc à appliquer directement la transformation linéaire $\left(\left(\hat{A}_f.H_i\right)^T\right)^{-1}$ de l'image compensée vers l'image attaquée. De manière à améliorer la qualité de l'image compensée, nous avons appliqué la transformation en utilisant une interpolation spatiale en intensité de type bilinéaire (4.24). D'autres schémas d'interpolation d'ordre supérieur, telle que l'interpolation bicubique, peuvent également être utilisés.

$$O(x,y)=(1-a).(1-b).I(x',y')+(1-a).b.I(x',y'+1)+a.(1-b).I(x'+1,y)+a.b.I(x'+1,y'+1) \tag{4.24}$$

avec : x' et y' réels , $a = x'-[x]$ et $b = y'-[y']$.

4.4.6. Tests des transformations linéaires inverses possibles

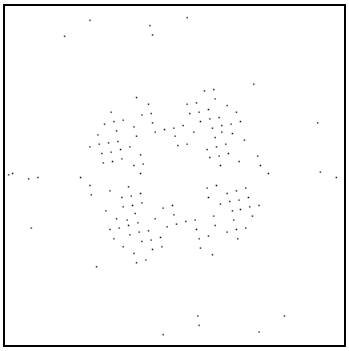
Comme nous l'avons montré au paragraphe 4.4.4, l'estimation de la transformation géométrique linéaire $\hat{A}_f$ est ambiguë, et par conséquent l'application $\hat{A}_s$ dans le domaine spatial l'est égale-

ment. Afin de lever cette ambiguïté, nous sommes donc obligés de tester séparément chacune des 8 transformations inverses possibles sur l'image attaquée, et tenter d'extraire ensuite le tatouage. La transformation géométrique retenue sera alors celle pour laquelle les scores de confiance S_{min} et S_{mean} (définis au chapitre 2) seront les plus élevés.

4.5. Résultats expérimentaux

Nous avons évalué l'efficacité de notre technique de resynchronisation face à différentes déformations géométriques linéaires. La figure ci-dessous illustre quelques unes des attaques géométriques appliquées à l'image (rotation, homothétie, cisaillement, etc.), ainsi que la manière dont elles ont été détectées, puis compensées par le processus de resynchronisation.

rotation de -13 degrés

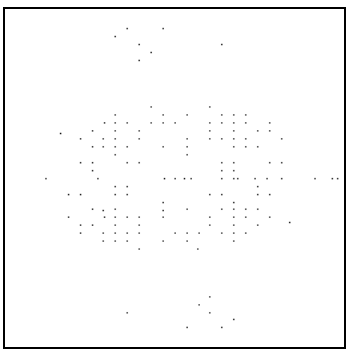


grille détectée

image redressée

$$A_S = \begin{pmatrix} 0{,}9748 & 0{,}2249 \\ -0{,}2249 & 0{,}9748 \end{pmatrix} \qquad \hat{A}_f = \begin{pmatrix} 0{,}2251 & 0{,}9739 \\ 0{,}9749 & -0{,}2248 \end{pmatrix} \qquad \hat{A}_S = \begin{pmatrix} 0{,}9749 & 0{,}2248 \\ -0{,}2251 & 0{,}9739 \end{pmatrix}$$

changement d'échelle (0.8*x, 1.2*y)



grille détectée

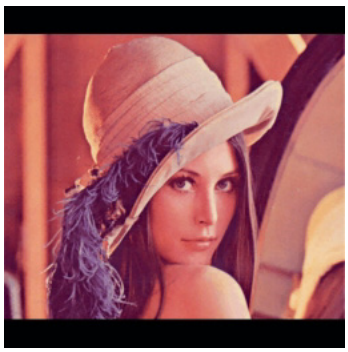
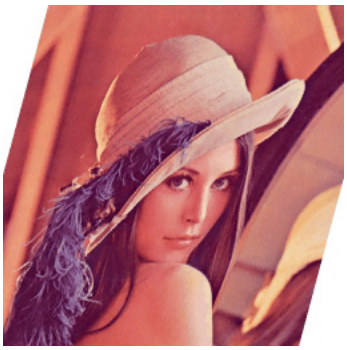
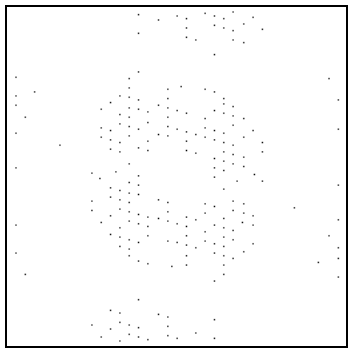


image redressée

$$A_S = \begin{pmatrix} 0{,}8 & 0 \\ 0 & 1{,}2 \end{pmatrix} \qquad \hat{A}_f = \begin{pmatrix} 0 & 1{,}248 \\ -0{,}8337 & 0 \end{pmatrix} \qquad \hat{A}_S = \begin{pmatrix} 0{,}8013 & 0 \\ 0 & 1{,}1995 \end{pmatrix}$$



cisaillement horizontal de 15 degrés

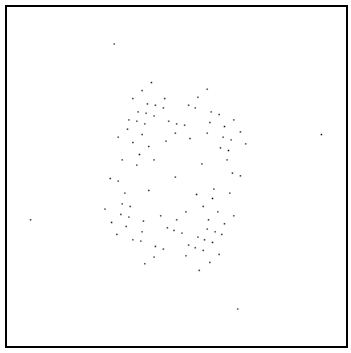


grille détectée

image redressée

$$A_S = \begin{pmatrix} 1 & -0{,}2679 \\ 0 & 1 \end{pmatrix} \qquad \hat{A}_f = \begin{pmatrix} 0 & 1{,}0002 \\ -1{,}0006 & 0{,}268 \end{pmatrix} \qquad \hat{A}_S = \begin{pmatrix} 0{,}9998 & -0{,}2678 \\ 0 & 0{,}9994 \end{pmatrix}$$

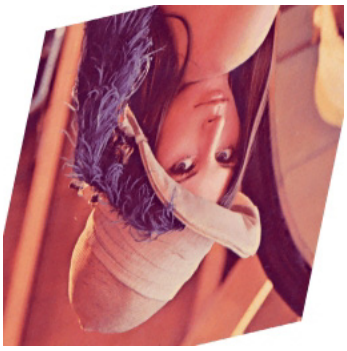
rotation -13 + zoom + miroir



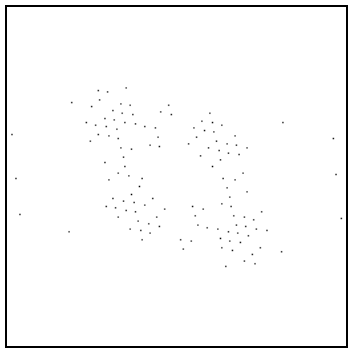
grille détectée

image redressée

$$A_S = \begin{pmatrix} -1{,}1692 & -0{,}2699 \\ -0{,}2024 & 0{,}8769 \end{pmatrix} \qquad \hat{A}_f = \begin{pmatrix} 0{,}8030 & 0{,}1907 \\ 0{,}2486 & -1{,}0812 \end{pmatrix} \qquad \hat{A}_S = \begin{pmatrix} -1{,}1671 & -0{,}2683 \\ -0{,}2058 & 0{,}8775 \end{pmatrix}$$



rotation -13 + cisaillement + flip



grille détectée

image redressée

$$A_S = \begin{pmatrix} 0{,}9744 & 0{,}2294 \\ -0{,}225 & -1{,}0694 \end{pmatrix} \qquad \hat{A}_f = \begin{pmatrix} 1{,}0798 & 0{,}2254 \\ 0{,}2295 & 0{,}9763 \end{pmatrix} \qquad \hat{A}_S = \begin{pmatrix} 0{,}9739 & 0{,}229 \\ -0{,}2248 & -1{,}0772 \end{pmatrix}$$

Figure 4.26 – *Exemples de transformations linéaires compensées par le processus de resynchronisation*

Les expériences réalisées montrent que le système de resynchronisation est capable d'estimer avec une bonne précision les déformations géométriques linéaires appliquées à une image tatouée. La qualité de l'estimation de l'application linéaire $\hat{A}_s$ dépend principalement du nombre et de la précision des pics détectés dans le domaine de Fourier, ainsi que de la manière dont l'espace de Hough a été discrétisé. D'autre part, il convient de remarquer que les éventuelles imperfections de l'image redressée peuvent être compensées en partie par le sur-échantillonnage appliqué lors de la mise en forme de la marque qui autorise un décalage de l'ordre du pixel, mais aussi par la méthode de resynchronysation des déformations locales décrites au paragraphe 3, à condition bien évidemment que la qualité de l'image redressée le permette. En effet, outre ces imprécisions d'ordre géométrique, l'image redressée subit également une distorsion photométrique qui peut être significative suivant les déformations géométriques appliquées à l'image (par exemple, dans le cas d'un sous-échantillonnage important). Cette distorsion est liée en grande partie aux interpolations réalisées lors des différentes déformations de l'image (*i.e.* attaques et compensation géométrique). Il se peut également que l'image subisse également en plus des déformations géométriques, une attaque de type photométrique comme une compression avec perte ou un filtrage passe-bas.

4.6. Limites de la méthode

Notre méthode de resynchronisation aux déformations géométriques globales de l'image se limite uniquement aux transformations linéaires. Les translations et les recadrages (non centrés) ne peuvent donc pas être compensés, de même que les transformations globales qui ne sont pas des applications affines (composition d'une rotation et de l'attaque « stirmark » par exemple). La méthode montre également ses limites lorsque l'on combine une application linéaire avec une attaque photométrique forte (ajout de bruit, filtrage passe-bas, etc.). Dans ce cas de figure, l'algorithme est généralement pris en défaut par le fait que les pics caractéristiques du motif périodique ne sont plus détectables dans le plan des fréquences, rendant alors impossible l'estimation des paramètres de l'application linéaire.

En dehors des manipulations décrites précédemment, on peut toujours concevoir des attaques malveillantes visant délibérément à tromper le système de resynchronisation ou à le rendre totalement inopérant. Parmi ces attaques, on peut en imaginer une qui consisterait à ajouter *a posteriori* un autre motif périodique à l'image, mais de nature (*i.e.* orientation et période) complètement différente. En procédant ainsi, l'alignement et l'écartement des pics dans la FFT seraient perturbés par des pics parasites (correspondant au deuxième motif), ce qui pourrait fausser l'estimation des paramètres d'une éventuelle transformation linéaire. Une autre attaque possible contre notre système est la « template attack » évoquée au paragraphe 2.2.2. En effet, cette attaque vise directement à supprimer les pics caractéristiques dans le domaine fréquentiel. Dans notre cas, cette attaque n'est efficace que si elle est réalisée au niveau du support car les pics introduits ne sont pas détectables de-

puis la FFT de l'image. Il est alors possible d'envisager une parade en rendant le support de tatouage dépendant de la clé secrète, en définissant par exemple un dictionnaire de transformées et un domaine de recherche pseudo aléatoire. La question qui reste alors en suspens, est de savoir si deux supports générés à l'aide de deux clés différentes, sont suffisamment éloignés (*i.e.* différents).

5. Conclusion

Nous avons présenté dans ce chapitre, deux méthodes de resynchronisation complémentaires opérant toutes deux en mode aveugle. La première méthode, qui a fait l'objet d'un brevet, est dédiée aux déformations géométriques locales de faible amplitude comme celles engendrées par l'attaque « Stirmark », tandis que la seconde permet de compenser spécifiquement les applications linéaires globales (*e.g.* rotations centrées, changements d'échelle, cisaillements, etc.). Ces deux techniques de resynchronisation donnent de très bons résultats lorsque l'image subit simplement des attaques géométriques de même type. En effet, si l'on applique successivement à l'image une transformation linéaire globale, suivie d'une attaque « Stirmark », la déformation résultante ne sera pas forcément compensable ; bien que ces deux attaques le soient séparément. Outre ce cas particulier, la resynchronisation peut être aussi prise en défaut lorsque l'on combine par exemple une attaque géométrique avec une attaque photométrique forte (*e.g.* compression, filtrage, etc.), ou si les interpolations liées aux déformations géométriques entraînent une perte d'information importante, ce qui est le cas lors d'un sous-échantillonnage important de l'image. Dans ces cas, l'échec de la resynchronisation doit cependant être relativisé par la qualité visuelle, souvent médiocre, de l'image attaquée. Enfin, les translations et les recadrages de l'image peuvent être également compensés par la première méthode de resynchronisation, mais au prix, soit d'une combinatoire très importante (*i.e.* agrandissement de la taille de la fenêtre de recherche du BM), soit en réduisant la robustesse aux attaques cryptographiques (*e.g.* attaques par collusion) en ne bruitant le tatouage que localement. Ces résultats montrent qu'un choix doit être réalisé au niveau des attaques face auxquelles le tatouage doit résister. Un algorithme de tatouage ne pourra jamais être robuste à l'ensemble des manipulations d'images. Ce compromis dépendra essentiellement de l'application visée et des besoins de l'utilisateur final.

Chapitre 5

Protection de l'intégrité des images

1. Introduction

Le problème de l'intégrité est encore peu abordé par la communauté « watermarking » et de nombreuses questions restent ouvertes. On peut par exemple se demander s'il est préférable d'avoir recours à un tatouage fragile plutôt qu'à un tatouage robuste, ou bien encore opter pour une toute autre solution (*i.e.* cryptage classique) ? D'autre part, un service d'intégrité remet partiellement en cause certains paramétrages communément établis en tatouage d'image pour assurer une fonction plus classique de sécurité de type « droits d'auteur », notamment en termes de quantité et nature des informations cachées (pour le copyright, la marque est indépendante de l'image et est usuellement un identifiant codé sur 64 bits), ainsi qu'en termes de robustesse.

1.1. Notion d'intégrité

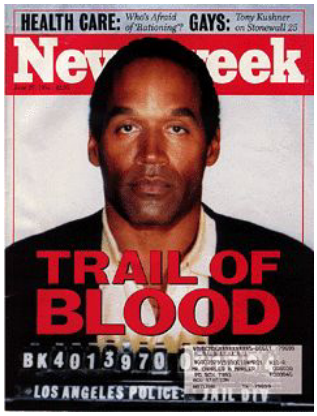
La notion d'intégrité visuelle est un concept bien connu en sécurité. Sa définition repose sur une décision binaire qui garantit que les données reçues sont rigoureusement identiques à celles émises. Cette définition est applicable à tout type de documents numériques, néanmoins, dans la pratique, elle s'avère être beaucoup trop stricte et inadaptée pour les documents multimédia. En effet, l'interprétation que l'on a d'une image dépend principalement des éléments la constituant plutôt que des valeurs numériques exactes des pixels ou de sa résolution. En d'autres termes, le problème de l'intégrité des images se pose principalement en termes de contenu sémantique ; c'est-à-dire la détection des modifications du document pouvant engendrer une gêne dans sa visualisation et/ou une erreur dans son interprétation (modification de la légende, disparition d'un visage, etc.). Dans le but d'assurer un service d'intégrité approprié aux images, il est donc primordial de distin-

guer les manipulations malveillantes consistant à détourner le contenu initial de l'image, des manipulations liées à son utilisation ou son stockage sous une forme numérique (conversion de format, compression, ré-échantillonage, filtrage, etc.) réalisés par des fournisseurs de contenu ou les utilisateurs eux-mêmes. Malencontreusement cette distinction n'est pas toujours aisée d'un point de vue informatique et dépend en partie du type d'image et de son utilisation. Par exemple, dans le cas particulier de l'imagerie médicale, des manipulations anodines, comme une simple compression, voire le processus de tatouage lui-même, peuvent causer la disparition de certains signes visibles d'une pathologie faussant alors le diagnostic du médecin. Dans ce contexte, l'utilisation de méthodes dites classiques sera plus appropriée pour garantir une intégrité stricte du document.

1.2. Exemples classiques de manipulations malveillantes

Dans notre société, les messages véhiculés par les images ont un impact considérable. En effet, le réalisme d'une photographie est tel que nous avons tendance à prendre pour réelles des scènes qui ne le sont pas (toutes les images, y compris celles réalisées en toute innocence, ont la capacité d'être détournées de leur sens). Les manipulations, qui avant, nécessitaient des moyens coûteux sont désormais à la portée de tout le monde et les progrès de la technique et du tout numérique les rendent quasi indécelables. Dans ce contexte, un service d'intégrité d'image n'a bien évidemment pas la prétention de vérifier la véracité des événements, mais de déceler des manipulations qui auraient pu y être apportées *a posteriori* (*i.e.* entre la prise de la photographie et sa diffusion) dans le but de détourner le contenu de l'image ou de rendre impossible toute interprétation. Nous donnons ci-après quelques exemples célèbres de manipulations intentionnelles d'images (sources : Ça m'intéresse – septembre 2000 [Cam00]).La couverture du magazine « Time » du 27 juin 1994 est un bel exemple de falsification d'image (*c.f.* Figure 5.1), ainsi qu'un véritable scandale journalistique qui a fait couler beaucoup d'encre à l'époque. Les éléments ajoutés à la photo originale d'O.J. Simpson (flou, noircissement du visage, halo obscur), créés de toutes pièces au moyen d'un logiciel de retouche à des fins de manipulation, exposèrent au grand jour le problème de la numérisation des images. Un autre exemple qui a fait le tour du monde est celui du recadrage de la place Tien An Men. En sélectionnant le personnage et le premier char, le recadrage de cette photographie n'en modifie pas la signification générale, mais en dramatise et en mystifie l'action. Une autre affaire de photographie truquée célèbre est celle diffusée en 1995, dans l'émission de France 3, « La marche du siècle », où de jeunes « beurs » avaient été transformés à leur insu en redoutables intégristes (source : Le Monde Diplomatique [Ros95]). Plus récemment, une photographie publiée en une du quotidien autrichien « Neue Kronen Zeitung », prétendait illustrer l'agressivité des manifestants opposés à l'entrée du parti de Haider dans le gouvernement autrichien. Par un truquage numérique on a recadré la photographie et raccourci la distance entre un manifestant et un policier apparemment directement frappé. En réalité, comme l'atteste l'image originale diffusée par l'agence Reuters [Reut], une distance de près de deux mètres séparait les deux protagonistes. Aussi l'utilisation comme élément à charge par

couverture du magazine « Time »
(photo manipulée)



couverture du magazine « Newsweek »
(photo originale)

Figure 5.1 – *Exemple de falsification d'image (l'affaire O.J. Simpson)*

l'image, l'audio ou la vidéo devient plus que douteuse et critiquable à l'heure où les caméras de surveillance envahissent les villes, les stades et les routes.

1.3. Schéma générique d'un système d'authentification d'image

On se propose de définir un schéma générique d'un système d'authentification d'image (dont différentes formulations ont été initialement proposées par Wu et Liu [WL98] et Lin et Chang [LC00]). Pour être efficace, ce dernier doit satisfaire les critères suivants :

- *Sensibilité* - le système doit être capable de déceler des manipulations pouvant modifier l'interprétation que l'on a d'une image, telles que des recadrages ou des retouches locales (exemple Figure 1) ;

- *Tolérance* - le système doit être tolérant vis-à-vis des algorithmes de compression avec pertes tels que Jpeg, et plus généralement vis-à-vis des manipulations bienveillantes (générées, par exemple, par les fournisseurs de contenu multimédia ou un utilisateur de bonne foi) ;

- *Localisation des régions altérées* - le système doit être en mesure de donner à l'utilisateur une information visuelle permettant d'identifier rapidement les régions qui ont été manipulées ;

- *Reconstruction des régions altérées* - le système doit éventuellement permettre une restauration partielle des zones de l'image qui ont été manipulées ou détruites, afin de donner à l'utilisateur la possibilité de se faire une idée sur le contenu original de ces régions.

En plus des critères précédents, d'autres contraintes techniques (classiques du tatouage d'image) sont également à prendre en considération :

- *Mode de stockage* - il est préférable de cacher les données d'authentification dans l'image elle-même, sous la forme d'un tatouage, plutôt que dans un fichier séparé comme dans le cas d'une signature externe ;

- *Mode d'extraction* - suivant que les données d'authentification sont dépendantes ou non de l'image, on optera pour un mode d'extraction du tatouage aveugle ou semi-aveugle. En mode d'extraction aveugle, la marque représentant les données d'authentification est récupérée à partir de l'image marquée seule (éventuellement manipulée), alors qu'en semi-aveugle il s'agit principalement de vérifier la présence de telle marque dans une image (via un score de corrélation). Il est bien évident qu'un mode d'extraction non aveugle est dénué de sens pour un service d'intégrité dans la mesure où il fait appel à l'image originale ;

- *Algorithme asymétrique* - Contrairement aux services de sécurité plus classiques comme le « copyright » où l'on peut se contenter d'une même clé (privée) pour l'insertion et l'extraction de la marque, un service d'intégrité nécessite de préférence l'utilisation d'un algorithme de tatouage asymétrique (ou de chiffrement, selon le cas) dans la mesure où tout un chacun doit pouvoir s'assurer de l'intégrité d'une image ;

- *Visibilité* - les données d'authentification doivent être invisibles (dans les conditions normales de visualisation). Il s'agit de faire en sorte que l'impact visuel du marquage (*i.e.* distorsion) soit le plus faible possible afin que le document marqué reste fidèle à l'original ;

- *Robustesse et sécurité* - les données d'authentification doivent être protégées par des méthodes de chiffrement de manière à éviter qu'elles soient falsifiées ou manipulées ;

- *Protocoles* - enfin, les protocoles tiennent également une place prépondérante dans tout système d'authentification d'images. En effet, l'algorithme ne permet pas à lui seul de garantir l'authenticité d'une image. Il est nécessaire de définir en plus un ensemble de spécifications décrivant les conventions et les règles du système, comme par exemple la gestion des clés ou bien encore éviter qu'une image déjà protégée, puisse l'être à nouveau, *a fortiori* si elle a été manipulée.

2. Etat de l'art

2.1. Description

Cette section n'a pas pour objectif de dresser un panorama complet et exhaustif des différentes techniques permettant d'assurer un service d'intégrité pour les images. Néanmoins, elle vise à présenter dans les grandes lignes plusieurs méthodes significatives du domaine afin d'introduire pro-

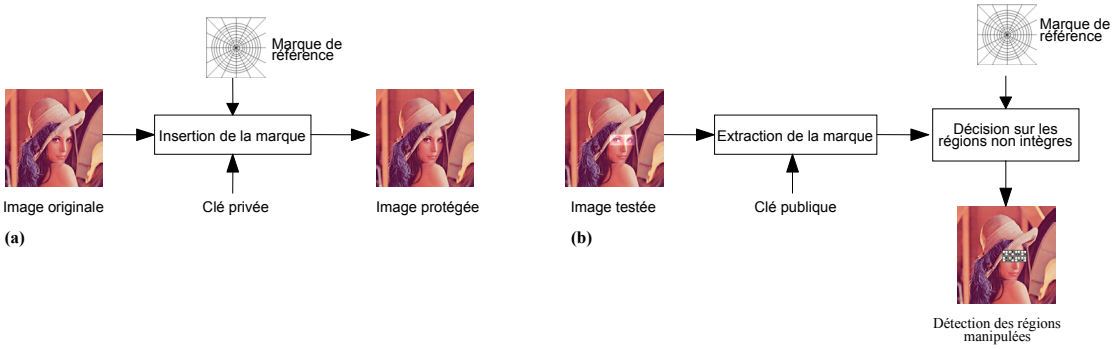


Figure 5.2 – *Schéma général d'un système d'intégrité basé sur un tatouage fragile*

gressivement les notions clés associées à ce type de service. Les systèmes d'authentification des images peuvent être regroupés de plusieurs manières suivant qu'ils assurent un service d'intégrité stricte ou bien une intégrité en termes de contenu, suivant le mode de stockage des données d'authentification (*i.e.* tatouage ou signature externe), ou bien encore selon la nature des informations qu'ils enfouissent dans le document à protéger.

2.2. Tatouages fragiles

2.2.1. Principe

Les premières méthodes proposées pour assurer un service d'intégrité étaient basées sur l'utilisation d'un tatouage fragile, par opposition au tatouage robuste classiquement utilisé pour la protection des droits d'auteur. Le principe de ces approches est d'insérer une marque ou un logo binaire (généralement prédéfini et indépendant des données à protéger [YM97]) dans l'image d'origine de telle manière que les moindres modifications apportées à l'image se répercutent également sur la marque insérée (*c.f.* Figure 5.2.a). Pour vérifier l'intégrité d'une image, il suffit alors de vérifier localement la présence de cette marque (*c.f.* Figure 5.2.b).

2.2.2. Insertion de « checksums » dans les LSB

Une des premières techniques utilisées pour vérifier l'intégrité d'une image visait à insérer des valeurs de « checksums » dans les bits les moins significatifs (LSB) des pixels de l'image. L'algorithme proposé par Walton [Wal95] en 1995 consiste à sélectionner, de manière pseudo-aléatoire (en fonction d'une clé), des groupes de pixels et de calculer, pour chacun d'eux, une valeur de « checksum ». Ces valeurs sont obtenues à partir des nombres formés par les 7 bits les plus significatifs (MSB) des pixels sélectionnés, et sont ensuite insérées sous forme binaire au niveau des bits de poids faible. Ci-après, de manière plus détaillée l'algorithme tel qu'il était proposé à l'origine :

de poids faible. Ci-après, de manière plus détaillée l'algorithme tel qu'il était proposé à l'origine :

Algorithme 1 - Etape d'insertion

6. Soit une valeur de N suffisamment grande ;
7. Diviser l'image en blocs de taille 8×8 pixels ;
8. Pour chaque bloc B_i :
   - définir un ordre de parcours pseudo-aléatoire (selon par exemple une clé secrète et l'indice du bloc B_i) des 64 pixels $(p_1, p_2, \ldots, p_{64})$;
   - générer une séquence pseudo-aléatoire de 64 entiers $(a_1, a_2, \ldots, a_{64})$ du même ordre de grandeur que N ;
   - la valeur de checksum S est alors calculée de la manière suivante :

$$S = \sum_{j=1}^{64} \left(a_j \cdot g\left(p_j\right) \right) \bmod N$$

     avec $g(p_j)$ le niveau de gris du pixel p_j en ne tenant compte que des 7 MSB.

   - coder et crypter S en binaire ;
   - insérer la séquence binaire obtenue au niveau des LSB des pixels du bloc

L'algorithme de vérification est dual de celui d'insertion. Il consiste à vérifier pour chaque bloc, la valeur de « checksum » recalculée à partir des MSB des pixels de l'image testée, avec celle de l'image originale codée au niveau des LSB.

Cette méthode garantit d'une part, en insérant les données d'authentification directement au niveau des LSB de l'image, une distorsion visuelle minime, quasi imperceptible par l'œil humain. D'autre part, elle a l'avantage d'être simple, rapide et sensible à la moindre modification de l'image (i.e. réponse binaire équivalente à une intégrité stricte). Si on échange, par exemple, les MSB de deux pixels quelconques d'un même bloc, la valeur de S s'en trouvera automatiquement modifiée car chaque pixel p_j est multiplié par un coefficient a_j différent. De plus, l'ordre de parcours des pixels p_j ainsi que les valeurs des coefficients a_j sont dépendants du bloc, ce qui rend impossible un éventuel « copier/coller » entre deux blocs différents d'une même image.

Le lecteur remarquera cependant, que dans la version décrite ci-dessus, il est possible d'intervertir deux blocs homologues (*i.e.* de même position) de deux images protégées avec la même clé, sans que le système ne décèle une perte d'intégrité (depuis différentes améliorations ont été proposées [Fri99] pour pallier à ce type d'attaque). Par contre si l'image est légèrement recadrée ou compressée, le système détecte une perte d'intégrité alors que le contenu sémantique de l'image reste inchangé.

2.2.3. « Self-embedding »

Fridrich et Goljan [FG99b] ont, quant à eux, développé une technique utilisant également les LSB comme support, mais dans le but, cette fois-ci, de cacher suffisamment d'informations afin de pouvoir non seulement déceler d'éventuelles manipulations, mais surtout de permettre une reconstruction partielle des régions détériorées. L'idée de base consiste à découper l'image en blocs 8×8, à en calculer les coefficients DCT (Transformée en Cosinus Discrète) en ne tenant compte bien évidemment que des MSB. Ces coefficients DCT sont ensuite quantifiés à l'aide de la table de quantification correspondant à une compression JPEG d'une qualité de l'ordre de 50%. La matrice quantifiée résultante est alors encodée sur 64 bits et insérée dans les LSB des pixels d'un autre bloc. Le bloc servant de support au tatouage doit être suffisamment éloigné du bloc à protéger afin d'éviter qu'une manipulation locale de l'image ne détériore à la fois l'image et les données de reconstruction.

Comme pour toutes les méthodes de tatouage utilisant les LSB comme support, l'impact visuel est très faible. Par contre, la qualité des régions restaurées est nettement inférieure à celle d'une compression JPEG 50%, mais largement suffisante pour informer l'utilisateur sur le contenu original de ces régions. Les auteurs ont également proposé une variante afin d'améliorer légèrement la qualité de la reconstruction en utilisant cette fois-ci les deux bits de poids faible comme support (la matrice quantifiée étant alors codée sur 128 bits). La reconstruction est certes meilleure, mais l'image tatouée perd sensiblement en qualité.

Le principal inconvénient de cette méthode est lié à la nature très fragile du tatouage qui ne garantit pas, dès lors que plusieurs régions de l'image ont été manipulées, une restauration correcte. En effet, les données de reconstruction correspondant à un bloc erroné peuvent elles aussi être altérées si les LSB les supportant ont eux aussi été modifiés. Ce problème est d'autant plus vrai lorsque l'image subit des manipulations globales, même « faibles », comme un filtrage passe-bas ou une compression JPEG. D'une manière générale, on peut donc légitimement se poser la question de l'intérêt des méthodes de tatouages fragiles vis-à-vis des techniques cryptographiques classiques, dans la mesure où elles ne garantissent qu'une intégrité stricte finalement.

2.3. Tatouages semi-fragiles

Face à ce semi-constat d'échec, les recherches s'orientent actuellement vers des approches dites semi-fragiles. Les méthodes ayant recours à un tatouage semi-fragile se distinguent des méthodes fragiles dans la mesure où elles offrent une robustesse accrue face à certaines manipulations d'image. L'objectif recherché est de pouvoir discriminer des opérations malveillantes, comme par exemple l'ajout ou la suppression d'un élément important de l'image, des transformations globales « raisonnables » ne portant pas atteinte au contenu sémantique de l'image. L'utilisation de telles méthodes est principalement motivée par le fait que les images sont généralement transmises et stoc-

kées sous une forme compressée et que pour la majorité des applications, les pertes liées au processus de compression n'affectent pas l'intégrité de l'image au sens de son interprétation.

2.3.1. Méthode transparente à la compression Jpeg

Lin et Chang proposent un algorithme d'authentification [LC00] robuste à la compression Jpeg. Les auteurs ont mis en évidence et démontré deux propriétés d'invariance des coefficients DCT vis-à-vis de la compression JPEG.

La première propriété énonce que si on donne à un coefficient DCT, quel qu'il soit, une valeur entière multiple d'un pas de quantification prédéfini Q'_m supérieur à tous les pas de quantification possibles d'une compression JPEG acceptable (*i.e.* facteur qualité de 50% environ), alors cette valeur peut être recalculée exactement après une compression JPEG acceptable. La deuxième propriété définit une règle d'invariance de la relation d'ordre entre les coefficients homologues de deux blocs DCT vis-à-vis de la compression JPEG. En effet, lors de la compression, les différents blocs DCT d'une image sont tous divisés par la même table de quantification, de ce fait la relation qui lie les coefficients de mêmes coordonnées de deux blocs reste inchangée après le processus de quantification. La seule exception est que dans certains cas, des inégalités strictes peuvent devenir de simples égalités, par le biais de la quantification.

Le système d'authentification proposé par Lin et Chang repose donc sur ces deux propriétés. La première est utilisée pour définir un support de tatouage robuste à la compression JPEG, tandis que la seconde sert à générer les données d'authentification proprement dites. Les étapes d'insertion et d'authentification peuvent se résumer ainsi :

Algorithme 2.a Génération des bits d'authentification

1. Découper l'image originale en blocs 8×8
2. Apparier les blocs deux par deux en fonction d'une clé secrète
3. Pour chaque paire de blocs (p, q) :
   - sélectionner un ensemble B de n coefficients DCT (autres que la composante continue) ;
   - générer la signature binaire ϕ de la paire de bloc à l'aide de la règle suivante :

$$\phi(\upsilon) = \begin{cases} 1, & F_p(\upsilon) - F_q(\upsilon) \geq 0 \\ 0, & F_p(\upsilon) - F_q(\upsilon) < 0 \end{cases}$$

avec $\upsilon \in B$ et $F(\upsilon)$ la valeur du coefficient υ

   - insérer les bits d'authentification suivant l'algorithme 2.b.

La signature binaire obtenue est ensuite cachée en partie dans chacun des deux blocs de la paire. L'algorithme de tatouage utilisé est relativement simple puisqu'il s'agit de définir une relation d'égalité entre les LSB des coefficients DCT prédéfinis avec les bits de la signature.

Algorithme 2.b Insertion du tatouage

1. Sélectionner un ensemble E, de $\frac{n}{2}$ coefficients DCT, avec $E \cap B = \varnothing$;
2. Pour cacher un bit d'authentification $\phi(\upsilon)$ dans un coefficient DCT ω :

   Soit $f'_p(\omega) = \left[\frac{F_p(\omega)}{Q'_m(\omega)} \right]$

$$\widetilde{F}_p(\omega) = \begin{cases} f'_p(\omega) \cdot Q'_m(\omega), & \text{si } LSB\left(f'_p(\omega)\right) = \phi(\upsilon) \\ \left(f'_p(\omega) + signe\left(\frac{F_p(\omega)}{Q'_m(\omega)} - f'_p(\omega) \right) \right) \cdot Q'_m(\omega), & \text{sinon.} \end{cases}$$

   avec $signe(x) = 0$ si $x < 0$, 1 sinon.

La vérification de l'intégrité d'une image est réalisée simplement en extrayant les bits d'authentification des coefficients DCT recevant le tatouage et en comparant la signature extraite avec celle obtenue à partir des blocs de l'image testée. Si les deux signatures correspondent parfaitement, la paire de blocs est alors jugée intègre, dans le cas contraire cela signifiera que l'un des deux blocs, voire les deux, ont été manipulés.

Les auteurs ont proposé de nombreuses améliorations à cette méthode, notamment l'ajout de bits de reconstruction. L'intérêt de ces bits supplémentaires est double. Ils permettent d'une part, comme leur nom l'indique, de reconstruire les blocs erronés, et d'autre part d'aider à localiser précisément les zones de l'image qui ont réellement été altérées (*i.e.* lever l'ambiguïté sur l'identification des blocs erronés). Les bits de reconstruction sont obtenus à partir d'une version sous-échantillonnée et compressée de l'image originale, et sont ensuite insérés de la même manière que les bits d'authentification dans quatre blocs de l'image originale.

2.3.2. Tatouage par région

Le tatouage par région consiste à découper l'image que l'on souhaite protéger en blocs relativement grands (de l'ordre de 64×64 pixels) et à insérer, dans chacun d'eux, une marque « relativement

robuste ». Lorsque l'on souhaite vérifier l'intégrité de l'image, on teste la présence de la marque dans les différents blocs. Dans le cas où la marque est présente avec une probabilité élevée dans chacun des blocs, on peut affirmer que l'image testée est intègre.

La technique « Variable-Watermark Two-Dimensional » (VW2D) décrite par Wolfgang et Delp [WD96], [WD99] reprend le principe décrit précédemment ; à savoir de cacher une marque binaire différente $W(b)$ dans chaque bloc b d'une image X. Ils préconisent de générer une marque binaire pseudo-aléatoire à partir de « m-sequences » [Pro95] à la manière des travaux initiés par Shyndel *et al* [STO94]. L'utilisation de « m-sequences » est en effet motivée par le fait qu'elles ont d'excellentes propriétés d'auto-corrélation, ainsi qu'une très bonne robustesse à l'ajout de bruit. Dans le système d'authentification proposé, la séquence binaire {0, 1} est transformée en une séquence de {–1, +1}, puis arrangée de manière à former un bloc de même taille que le bloc de l'image auquel elle va être modulée. La modulation de la marque et de l'image est réalisée très simplement en ajoutant ou en supprimant un niveau de gris au pixel correspondant (*c.f.* équation 5.1) :

$$Y(b) = X(b) + W(b) \tag{5.1}$$

avec X l'image originale, et Y l'image tatouée.

Ensuite, pour déterminer si la marque recherchée est bel et bien présente dans un bloc, on calcule un score statistique δ (*c.f.* équation 5.3) basé sur un calcul de corrélation (équation 5.2) entre l'image (marquée et attaquée) et la marque :

$$A(b) \cdot B(b) = \sum_i \sum_j A(i,j)B(i,j) \tag{5.2}$$

$$\delta(b) = Y(b) \cdot W(b) - Z(b) \cdot W(b) \tag{5.3}$$

avec Z l'image à tester, la marque $W(b)$ est supposée connue à l'extraction.

Si $\delta(b) < T$, avec T un seuil fixé par l'utilisateur, le bloc b est alors jugé intègre. En jouant sur la valeur de T, on tolère des changements plus ou moins importants dans l'image. De ce fait il est possible d'affiner la détection en définissant plusieurs seuils correspondant à plusieurs niveaux de dégradation pour les blocs (par exemple : T_1 - intègre, T_2 - légèrement altéré, T_3 - très dégradé, T_4 - complètement modifié, avec $T_1 > T_2 > T_3 > T_4$).

Fridrich [Fri98a], [Fri98b] propose une technique similaire, mais préconise, pour des raisons de sécurité, de rendre le tatouage dépendant de la région de l'image dans laquelle il est inséré. La marque binaire utilisée correspond à un signal pseudo-aléatoire généré à partir d'une clé secrète, du numéro du bloc et d'un M-tuplet de bits représentatifs de la portion d'image considérée. Chaque bloc est ensuite tatoué en utilisant une technique d'étalement de spectre, similaire à celle proposée

par Ó Ruanaidh [RP97]. D'après l'auteur, la marque offre une bonne robustesse aux opérations classiques de traitement d'image telles que de petits ajustements de contraste ou de luminosité, l'ajout de bruit, l'application de filtres passe-bas ou passe-haut, l'égalisation d'histogramme, ou bien encore une compression JPEG de l'ordre de 50%, permettant ainsi de distinguer les changements liés à l'utilisation d'une image, des manipulations malveillantes.

2.3.3. Autres approches

D'autres techniques sont étudiées ou en cours d'investigation. Parmi celles-ci on peut citer celle de Kundur et Hatzinakos [KH98], et celle de Lin et Chang [LC98b] qui utilisent les ondelettes. Le principe de la méthode proposée par Lin et Chang est de choisir, tout d'abord, un bruit pseudo-aléatoire et une ondelette de base, qui constituent le secret du système d'authentification. Puis de décomposer l'image en 4 sous-bandes (LL, LH, HL et HH) en fonction de l'ondelette de base choisie au départ. L'étape suivante revient à substituer la sous-bande HH par le bruit pseudo-aléatoire et à effectuer ensuite la transformation en ondelettes inverse afin d'obtenir l'image tatouée. Il est intéressant de noter que le fait de modifier uniquement la sous-bande HH (i.e. hautes fréquences) n'entraîne pas de dégradations visibles.

Le processus d'authentification consiste alors à effectuer la même décomposition que celle utilisée lors de la phase d'insertion, puis à corréler la sous-bande HH obtenue avec le bruit pseudo-aléatoire. Si l'image n'a subi aucune manipulation, le résultat du test ressemblera à une matrice de points uniformément répartis. Dans le cas contraire, la distribution perdra son caractère uniforme dans les régions où l'image a été manipulée. Les auteurs font remarquer que cette méthode est « perméable » à certaines manipulations telles qu'un flou ou un rehaussement des contours, dans la mesure où les changements ne sont pas trop importants. Expérimentalement, cette méthode permet également de laisser passer une légère compression JPEG. Par contre, les auteurs ne démontrent pas la robustesse de leur méthode face à des attaques spécifiques visant par exemple à substituer la sous-bande HH ou au contraire à la préserver (*i.e.* modifier l'image, puis réinsérer la sous-bande HH de l'image originale protégée). En d'autres termes, est-ce que le choix de l'ondelette de base comme secret est suffisant pour éviter ce type d'attaque ?

2.4. Signatures externes

Les signatures externes offrent une alternative aux techniques de tatouage classiques dans le cadre d'un service de contrôle d'intégrité dans les images. Contrairement aux techniques de tatouage d'image, la marque n'est pas insérée dans l'image elle-même, mais transmise avec celle-ci sous une forme chiffrée. On peut établir un parallèle entre l'utilisation de signatures numériques pour assurer un service d'authentification et le domaine de l'indexation d'image [Nas97], où de

nombreuses techniques ont recours à ce type d'empreintes externes (ou signatures condensées) pour retrouver des images en fonction de leur contenu. Ces signatures sont générées le plus souvent à partir d'attributs significatifs qui traduisent le contenu sémantique, tels que la couleur, la forme ou la texture.

2.4.1. Fonctions de hachage

Le rôle principal des fonctions de hachage est de permettre de vérifier l'intégrité d'un document numérique sans avoir recours à l'original. Une fonction de hachage opère généralement sur un message M de longueur arbitraire pour fournir une valeur de hachage h de taille fixe. Pour qu'une telle fonction soit considérée comme sûre elle doit vérifier les propriétés suivantes :

- Il est « facile » de calculer h connaissant M ;
- Il est « difficile » de retrouver M connaissant h ;
- Il est « difficile » de trouver un message M' (différent de M) ayant comme valeur de hachage $h' = h$.

En d'autres termes, une fonction de hachage sert à produire un condensé (ou une empreinte) unique, représentatif du document original. Il existe de nombreuses fonctions de hachage parmi lesquelles on peut citer : MD-4, MD-5 (Message Digest), CRC-32 (32 bits Cyclic Redundancy Check), SHA-1 (Secure Hash Algorithm), etc.

Row-column hash function

La technique des « row-column hash function » [WD96] consiste à calculer une valeur de hachage pour chaque ligne et chaque colonne de l'image originale. Lorsque l'on souhaite vérifier l'intégrité d'une image, on recalcule les valeurs de hachage des lignes et des colonnes de l'image à tester et on les compare avec celles de l'image originale. Pour localiser les éventuelles disparités, il suffit d'identifier les lignes et les colonnes qui sont différentes. Cependant, dans le cas où plusieurs zones de l'image ont été modifiées, on n'est plus capable de les localiser sans ambiguïté, c'est-à-dire que des régions intègres seront considérées comme altérées (*c.f.* Figure 5.3), ce qui réduit considérablement l'intérêt de cette technique.

Block based hash function

Un autre algorithme utilise également des fonctions de hachage, il s'agit du Block-Based Hash function (BBH) [WD96]. Le principe est similaire à celui décrit précédemment, à la différence près

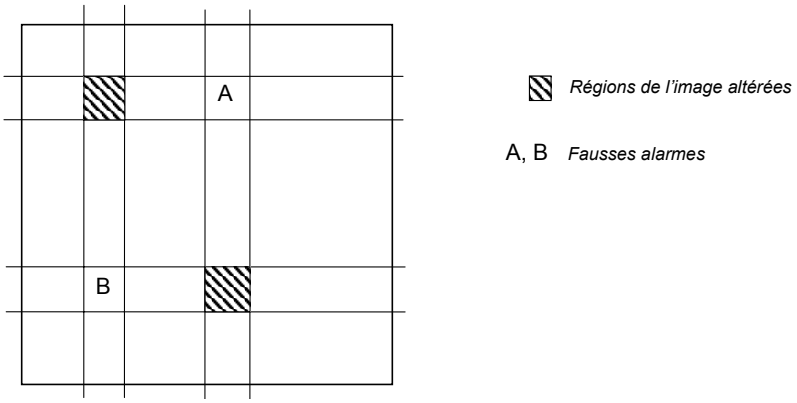


Figure 5.3 – *Ambiguïté dans la localisation des régions altérées de l'image*

qu'il n'opère plus sur les lignes ou les colonnes de l'image, mais sur des blocs. Ainsi lorsque l'on constate des différences dans les valeurs de hachage, il suffit de se reporter aux blocs concernés pour localiser les zones de l'image qui ont été manipulées.

Les fonctions de hachage ont la particularité d'être extrêmement sensibles à la moindre variation ; en effet il suffit de modifier la valeur d'un pixel d'un seul bit pour changer radicalement la valeur de hachage du bloc associé. Elles ne permettent donc pas de distinguer les manipulations malveillantes des manipulations bienveillantes (*i.e.* utilisateurs ou fournisseurs de contenus).

2.4.2. Signature basée sur des caractéristiques de l'image

Contrairement aux techniques ayant recours à des fonctions de hachage pour générer une empreinte de l'image, certains auteurs, comme Queluz [Que98] ou Lin et Chang [LC98a], [LC98c], proposent d'extraire des caractéristiques intrinsèques de l'image, telles que les contours, et de les crypter à l'aide d'un algorithme de chiffrement asymétrique afin de les transmettre en même temps que l'image (*c.f.* Figure 5.4). Dans le cas d'un système d'authentification basé sur l'utilisation d'une signature externe, la distinction entre des manipulations innocentes et malveillantes repose principalement sur le choix des caractéristiques de l'image pour générer la signature. Ce problème est exactement le même que celui rencontré par certaines méthodes semi-fragiles, à la différence près qu'ici la contrainte de quantité d'informations, liée à la capacité de l'algorithme de tatouage, ne se pose plus (puisque les informations sont stockées dans un fichier séparé). Certains auteurs, comme Lin et Chang, suggèrent de coder la relation d'ordre entre les coefficients DCT homologues de deux blocs distincts (la méthode est la même que celle présentée au paragraphe 2.3.1). Queluz, quant à elle, opte pour des caractéristiques plus visuelles (principalement les contours), mais également moins stables. Pour compenser ce manque de stabilité, elle a mis en place des post-traitements complexes afin de réduire les fausses alarmes liées à une compression JPEG.

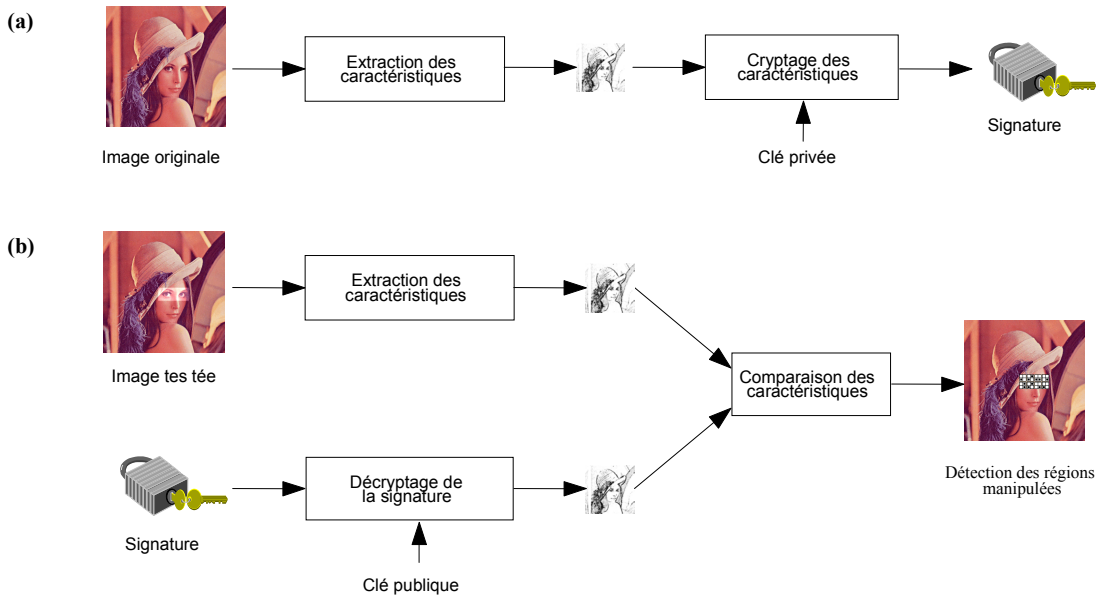


Figure 5.4 – *Principe général d'un système d'authentification utilisant une signature externe*

Bhattacharjee et Kutter [BK98] proposent également une technique ayant recours à une signature externe ; mais plutôt que d'extraire des caractéristiques par bloc d'image, ils suggèrent de rechercher des points d'intérêts (basée sur les travaux de Manjunath *et al.* [MSC96]) et de coder leurs coordonnées. La signature ainsi obtenue est ensuite chiffrée à l'aide d'un algorithme à clé privée / publique tel que RSA (méthode de chiffrement asymétrique inventée par Ronald **R**ivest, Adi **S**hamir et Leonard **A**dleman [RSA77]). L'inconvénient majeur de ces techniques qui font appel à une signature externe pour assurer un service d'intégrité d'image est que l'image ne s'auto-suffit plus. On perd par conséquent en grande partie, l'intérêt du tatouage d'image. De plus, ces méthodes soulèvent de nouveaux problèmes comme par exemple l'authenticité de la signature, ainsi que celle du couple image / signature.

2.5. Attaques malveillantes contre les algorithmes d'intégrité

Avant de conclure ce tour d'horizon des méthodes permettant d'assurer un service d'intégrité pour les images, il convient d'aborder le problème des attaques malveillantes de pirates (ou crackers). L'objectif commun de ces attaques, n'est pas de détourner le contenu d'une image (comme les manipulations présentées au paragraphe 1.2), mais d'utiliser les failles ou les faiblesses d'un système d'authentification afin de le tromper, autrement dit de faire croire au système qu'une image est intègre alors que son contenu a été modifié (ou l'inverse dans certains cas). Ce paragraphe n'a pas pour but de faire l'inventaire de toutes ces attaques, mais d'en présenter quelques unes parmi les

plus fréquentes. Même si certaines de ces attaques paraissent triviales et simples à prévenir, il est néanmoins très important d'en tenir compte lors de l'élaboration d'un algorithme d'authentification.

Une des attaques les plus courantes contre les systèmes à base de tatouage fragile, consiste à tenter de modifier une image protégée sans affecter le tatouage qu'elle contient, ou bien encore à tenter de créer une nouvelle marque que le détecteur considérera comme authentique. Prenons par exemple le cas volontairement simplifié où l'intégrité d'une image est assurée par une marque fragile, indépendante du contenu, insérée dans les LSB des pixels. Il est clair que si on modifie l'image sans se préoccuper de savoir quels sont les bits affectés par la manipulation, on a toutes les chances que la marque soit dégradée et l'attaque détectée. Par contre, si on prend soin de modifier l'image sans toucher aux LSB, la marque restera intacte et le système ne décèlera aucune falsification.

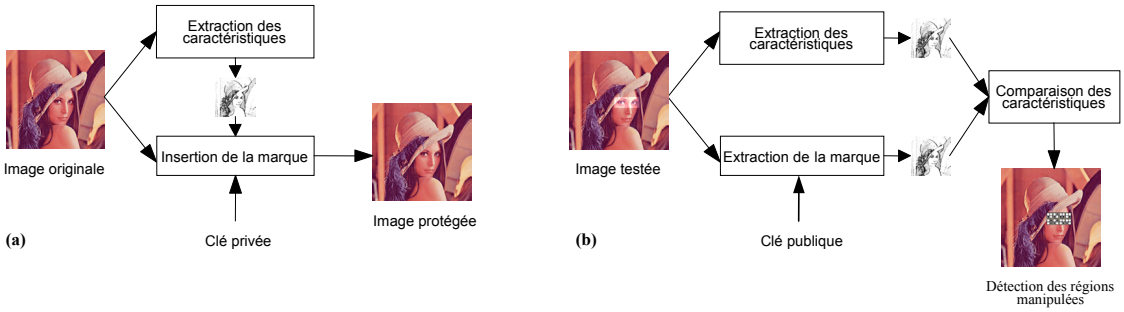
D'un point de vue plus général, dès lors que l'intégrité est assurée par une marque indépendante du contenu de l'image à protéger il est possible d'imaginer une attaque qui recopie une marque valide d'une image vers une autre (*e.g.* la « Copy Attack » de Kutter). De cette manière la deuxième image se retrouve alors protégée. Cette attaque peut très bien être appliquée sur la même image ; dans ce cas, la marque est dans un premier temps retirée de l'image, l'image est ensuite manipulée, et enfin la marque est réinsérée dans l'image manipulée, trompant ainsi le système d'authentification.

Dans le même esprit, la « Collage-Attack » [JGM00] proposée par Fridrich *et al.* consiste à créer une image contrefaite de toutes pièces à partir d'une banque d'images protégées par la même marque et la même clé. Cette attaque ne présuppose aucune connaissance *a priori* sur la marque binaire cachée, ni sur la clé secrète utilisée. Son principe est relativement simple puisqu'il consiste à remplacer chaque pixel de l'image à manipuler par le pixel qui lui est le plus similaire parmi les pixels de même position des images de la base. La difficulté de cette méthode est de disposer d'une banque d'images suffisamment variées pour obtenir une image falsifiée de bonne qualité visuelle.

Une autre attaque classique consiste à essayer de trouver la clé secrète utilisée pour générer la marque. Ce type d'attaque, également appelé « Brute Force Attack », est très connu par la communauté « sécurité ». Une fois la clé trouvée, il devient alors très facile pour un pirate de falsifier la marque d'une image protégée avec cette clé. La seule parade efficace est d'utiliser des clés de grande taille de manière à rendre cette attaque très dissuasive en termes de temps de calcul.

3. Méthodes proposées pour protéger le contenu des images

Dans ce chapitre, nous allons présenter deux méthodes permettant d'assurer un service d'intégrité pour des images. La première méthode est basée sur l'utilisation d'un tatouage robuste (nous utiliserons l'algorithme décrit au chapitre 2). L'idée est de cacher dans l'image à protéger des



(a) Protection de l'image, (b) Vérification de l'intégrité d'une image

Figure 5.5 –*Schéma général d'un système d'authentification basé sur un tatouage robuste*

informations sur son contenu. Lors de la phase de vérification, ces informations sont extraites et comparées avec les caractéristiques de l'image testée. Les différences constatées indiquent alors les régions qui ont été manipulées. La deuxième méthode est une alternative de la première. Elle consiste à combiner un tatouage robuste et une empreinte externe. Le tatouage est du même type que celui utilisé dans le cadre d'un service classique de droits d'auteur. La marque contenue dans l'image permet d'accéder à une empreinte externe stockée dans une base de données. Cette empreinte est un condensé des caractéristiques de l'image originale.

4. Protection de l'intégrité à l'aide un tatouage robuste

Le schéma de la méthode que nous proposons pour protéger le contenu des images est basée sur un tatouage robuste et a l'avantage d'être indépendante de l'algorithme de tatouage. Cependant, le tatoueur doit néanmoins disposer d'une forte capacité d'insertion et doit être capable d'extraire la marque en mode aveugle.

4.1. Principe général

Contrairement aux techniques classiques basées sur un tatouage fragile ou semi-fragile, la marque que nous utilisons n'est pas fixe, mais dépend de l'image elle-même. L'idée de base consiste à extraire certaines caractéristiques de l'image originale et à les cacher ensuite dans l'image sous la forme d'un tatouage robuste et invisible. Lorsque l'on souhaite vérifier l'intégrité d'une image, on compare simplement les caractéristiques de cette image avec celles de l'image originale contenues dans le tatouage (*c.f.* Figure 5.5). Si les caractéristiques sont identiques, cela signifiera que l'image n'a pas été manipulée, sinon les différences indiqueront les régions qui ont été altérées. Cette technique impose de nouvelles contraintes, principalement en termes de robustesse et de capacité d'insertion. En effet, il est impératif, d'une part, d'extraire la marque sans erreur sous peine d'avoir un taux éle-

vé de fausses alarmes. D'autre part, la précision de la détection des régions de l'image qui ont été manipulées est directement liée à la quantité d'information cachée dans l'image. Il est donc nécessaire de trouver un bon compromis pour la taille de la marque afin de satisfaire simultanément aux deux contraintes : robustesse et sensibilité de la détection.

4.2. Protection de l'image

4.2.1. Choix des caractéristiques

Le choix des caractéristiques de l'image est primordial dans la mesure où il va conditionner les manipulations que l'on pourra détecter et celles qu'on laissera passer. De plus, ce choix dépend également du type d'image considéré (peinture, image satellite, image médicale, photo, etc.), ainsi que de l'application visée. D'une manière générale, on sélectionne les caractéristiques de l'image en fonction de leur stabilité face aux différentes attaques. Typiquement, on recherchera des caractéristiques qui sont invariantes face à une compression JPEG, mais sensibles à des retouches locales de l'image.

Les traits les plus couramment utilisés pour caractériser les images sont les contours, la texture, le gradient, la luminance et la couleur. Les premiers tests que nous avons effectués ont porté sur les contours binarisés de l'image. Ce choix était motivé par le fait que les contours sont un des attributs les plus représentatifs d'une image. Malheureusement, pour des raisons de capacité, il était nécessaire d'extraire les contours à partir d'une version sous échantillonnée de l'image. Les contours alors obtenus étaient beaucoup trop imprécis pour permettre de détecter avec précision et fiabilité de petites variations de l'image.

Les expériences suivantes ont principalement porté sur la luminance moyenne par bloc. Cette caractéristique de l'image est très intéressante dans la mesure où elle est invariante (à un niveau de gris près) face à une compression JPEG, mais elle est susceptible d'être modifiée par des manipulations locales de l'image, telle que la suppression d'un personnage dans une scène.

4.2.2. Extraction des caractéristiques et mise en forme de la marque

L'extraction des caractéristiques est réalisée de la manière suivante. La première étape du procédé consiste à découper l'image en blocs. Typiquement on choisit des blocs 16×16 pixels ou 32×32 pixels. Au delà d'une taille de 32×32 pixels, les blocs sont trop gros pour permettre de détecter de petites manipulations comme l'ajout ou la manipulation d'une légende. Si l'on choisit au contraire des blocs plus petits, 8×8 pixels par exemple, on sera bien évidemment capable de détecter plus finement les manipulations, mais en contrepartie la taille du message à cacher dans l'image

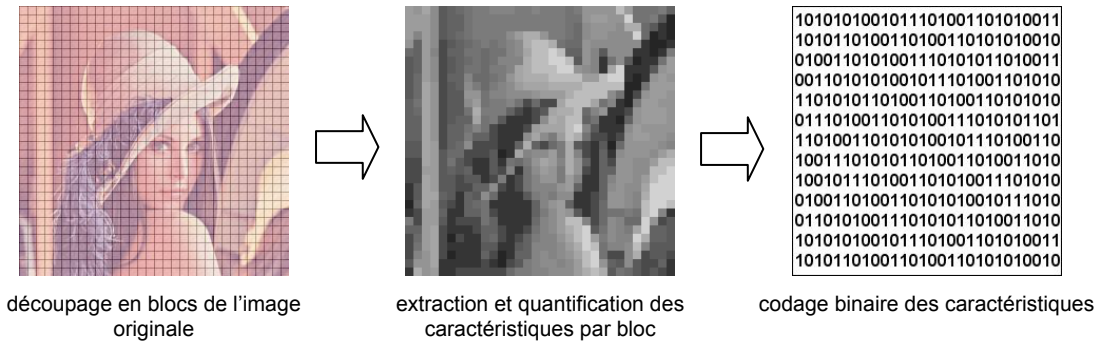


Figure 5.6 – *Extraction et codage des caractéristiques d'une image*

sera très importante et le tatoueur n'aura pas forcément une capacité d'insertion suffisante.

Une fois l'image découpée, on extrait les caractéristiques de chacun des blocs que l'on quantifie ensuite de manière à réduire la quantité d'information à cacher. Dans le cas de la luminance moyenne, on utilise généralement 16 niveaux de quantification. La signature ainsi obtenue est alors codée sous la forme d'une marque binaire, qui sera mise en forme, puis insérée dans l'image originale via l'algorithme de tatouage décrit au chapitre 2.

4.2.3. Tatouage itératif

Un des inconvénients de dissimuler les attributs caractéristiques de l'image sous la forme d'un tatouage réside dans le fait que l'image tatouée est légèrement modifiée par l'insertion de la marque. Même si ces variations sont imperceptibles à l'œil, elles affectent légèrement les caractéristiques intrinsèques de l'image. De ce fait, les caractéristiques de l'image originale et celles de l'image tatouée ne sont plus exactement les mêmes, et on risque alors de détecter des régions altérées alors que l'image n'a pas été manipulée. Ce risque de fausses alarmes est plus ou moins important en fonction de la nature des caractéristiques choisies et de l'algorithme d'insertion utilisé.

Principe

Pour résoudre ce problème, nous avons mis au point un processus de tatouage itératif. Le processus est initialisé en tatouant une première fois l'image originale avec ses propres caractéristiques. Puis, de manière itérative, on extrait les nouvelles caractéristiques de l'image tatouée que l'on insère à nouveau dans l'image originale sous la forme d'un tatouage. Seule l'image originale est marquée pour éviter d'accumuler des distorsions liées au processus de tatouage. De cette manière, grâce à ce processus itératif, les caractéristiques contenues dans le tatouage coïncident parfaitement avec celles de l'image une fois tatouée. Le processus itératif s'arrête lorsque les caractéristiques de l'image tatouée sont stables ou lorsqu'un nombre maximal d'itérations a été atteint.

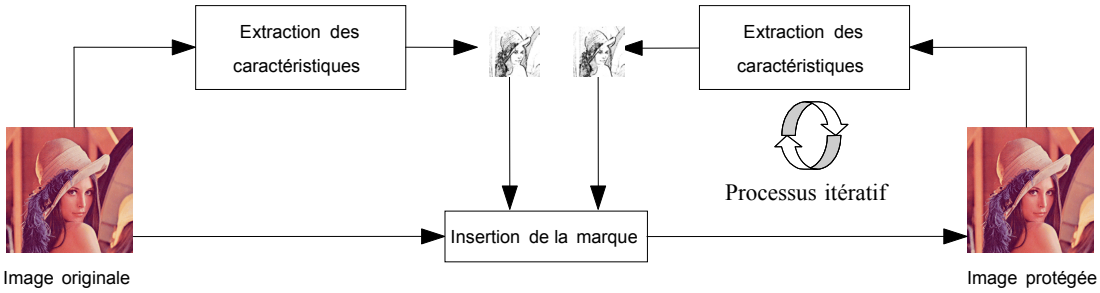


Figure 5.7 – *Processus de tatouage itératif*

Résultats

Dans la pratique, trois itérations sont suffisantes pour stabiliser les caractéristiques de l'image tatouée. L'exemple illustré par la figure ci-dessous montre la diminution rapide des fausses alarmes (symbolisées par des carrés blanc et noir) au fil des itérations.

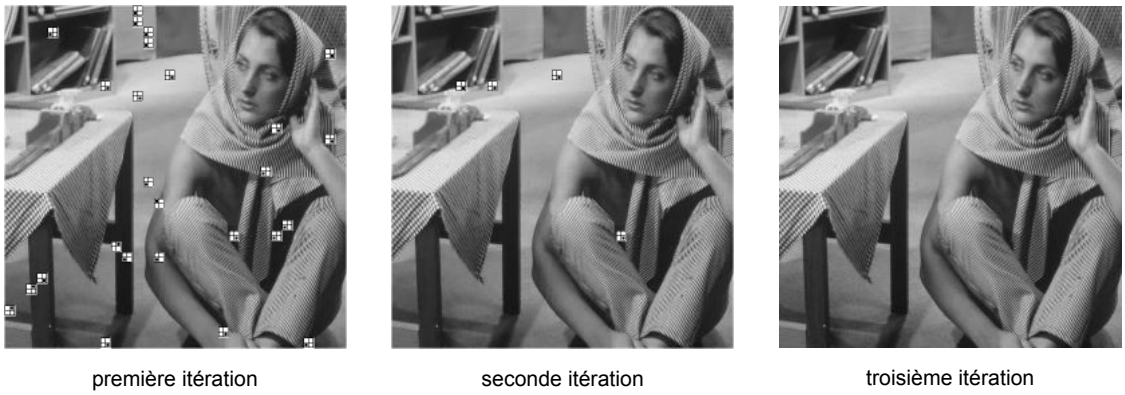


Figure 5.8 – *Illustration de l'efficacité du processus de tatouage itératif pour réduire le nombre de fausses alarmes*

4.3. Vérification de l'intégrité d'une image

4.3.1. Détection des erreurs et restauration des régions altérées

Le processus de vérification de l'intégrité d'une image se décompose en trois étapes :

- Extractions des caractéristiques de l'image originales contenues dans le tatouage ;
- Extraction des caractéristiques de l'image testée ;
- Comparaison des caractéristiques.

Extraction des caractéristiques de l'image originale contenues dans le tatouage

C'est l'étape critique du processus de vérification. En effet, la marque doit être extraite sans erreur sous peine de biaiser la vérification de l'intégrité de l'image testée. Cette opération est rendue difficile par la quantité très importante d'information enfouie, qui est de très loin supérieure à la taille d'un message communément utilisé pour la protection des droits d'auteur. D'autre part, les dimensions du tatouage sont directement liées à celles de l'image originale. De ce fait, si l'image subit des manipulations géométriques comme un changement d'échelle ou un recadrage, ses dimensions ne seront plus les mêmes, et l'information sur la taille du tatouage sera faussée. Une fois le tatouage extrait, on procède au décodage de la marque binaire afin de retrouver les valeurs numériques des caractéristiques quantifiées de l'image originale.

Extraction des caractéristiques de l'image testée

Les caractéristiques de l'image testée sont extraites et quantifiées suivant le même procédé que celui utilisé pour celles de l'image originale lors de l'étape d'insertion (le processus itératif en moins). De cette façon, les caractéristiques obtenues sont directement comparables à celles de l'image originale qui ont été reconstituées à partir du tatouage.

Comparaison des caractéristiques

La comparaison des caractéristiques est réalisée bloc à bloc. Si elles sont identiques, cela signifie que le bloc testé n'a pas été manipulé (ou qu'une éventuelle manipulation n'a pas altéré cette caractéristique). En cas de non correspondance, le bloc concerné est identifié par un symbole visuel de façon à alerter l'utilisateur. Suivant la nature des caractéristiques utilisées, il est possible de remplacer le bloc douteux par une estimation de son contenu original à partir des caractéristiques extraites. Dans le cas où la caractéristique utilisée est la luminance moyenne, les régions qui ont été manipulées seront remplacées par une mosaïque de blocs représentant les niveaux de gris moyen de cette partie de l'image originale.

4.3.2. Fausses alarmes et mauvaises détections

La précision de la détection implique que l'on détecte correctement les régions de l'image qui ont été modifiées tout en minimisant les fausses alarmes. Le choix de caractéristiques stables et représentatives du contenu sémantique de l'image, ainsi que le processus de tatouage itératif permettent de réduire considérablement le risque de fausses alarmes. Néanmoins, il peut arriver que des régions non attaquées soient détectées comme étant non conformes. Ces fausses alarmes sont le

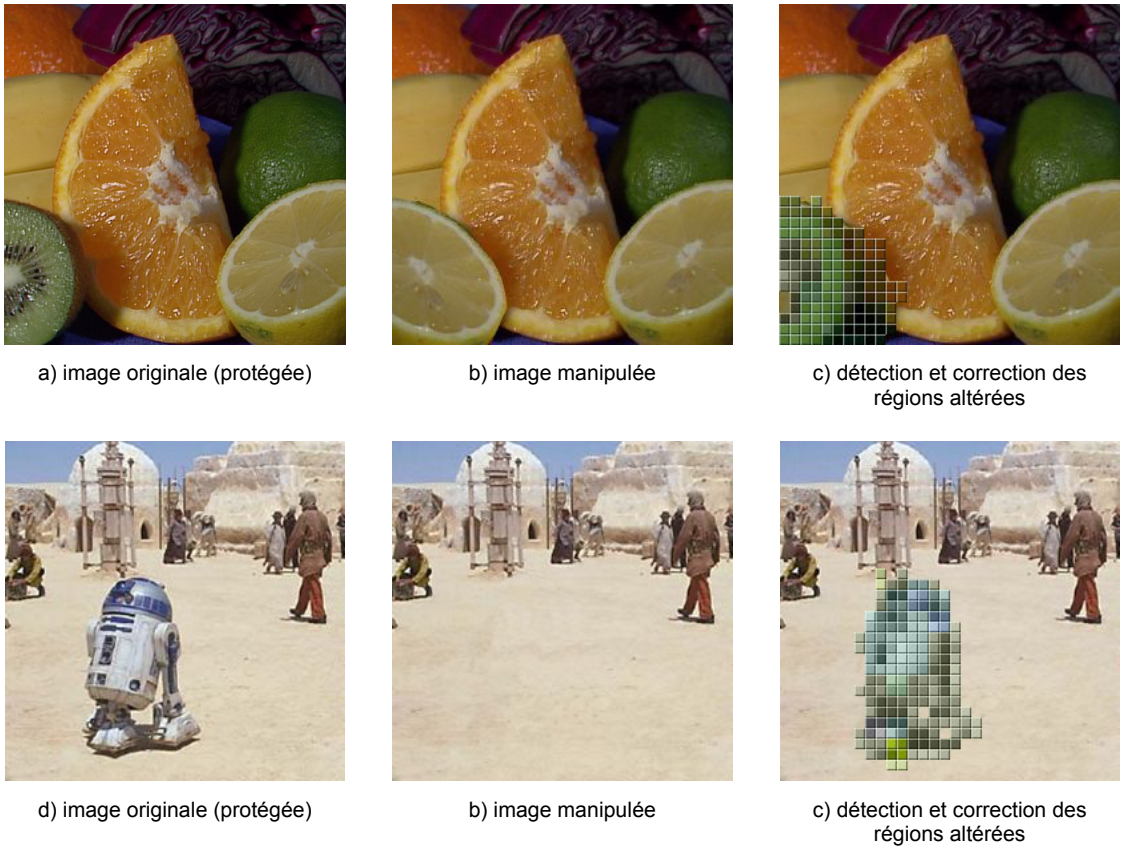


Figure 5.9 – *Exemples de vérification de l'intégrité d'une image à l'aide d'un tatouage robuste*

plus souvent dues à une mauvaise extraction du tatouage. En effet, les bits erronés au niveau de la marque vont entraîner des erreurs lors de la reconstruction des caractéristiques de l'image originale, biaisant ainsi le processus de comparaison.

Une autre source d'erreur provient de la quantification importante des caractéristiques utilisées. Dans le cas de la luminance moyenne par bloc, une variation de un niveau de gris moyen d'un bloc, suite à une compression JPEG par exemple, peut entraîner un changement de niveau de quantification et être interprétée comme une manipulation locale de l'image. Le faible nombre de niveaux de quantification ne nous permet pas d'introduire un seuil de tolérance (un changement de un niveau de quantification correspond à une variation de 1 à 16 niveaux de gris). Il est cependant possible de réduire le nombre de fausses alarmes à l'aide de post-traitements. En effet, dans la plupart des cas, une retouche locale de l'image se répercute sur plusieurs blocs. Une solution consiste alors à étudier *a posteriori* le voisinage des blocs erronés. Un bloc erroné isolé devra alors être considéré comme peu fiable.

4.4. Résultats expérimentaux

Les meilleurs résultats ont été obtenus en utilisant la luminance moyenne par bloc comme caractéristique du contenu de l'image. Les autres caractéristiques qui ont été essayées sont les contours et le gradient. La figure 5.9 illustre deux exemples classiques de retouche d'image. Les images de tests ont été protégées en cachant des informations de luminance et de chrominance. La luminance moyenne a été calculée pour des blocs de taille 16×16 et quantifiée sur 16 niveaux. Quant aux deux composantes de chrominance, elles ont été calculées pour des blocs de taille 32×32 et également sur 16 niveaux. Pour donner un ordre de grandeur, la quantité d'information cachée dans une image de taille 512×512 est de 6144 bits. Nous avons volontairement donné moins d'importance aux informations de chrominance dans la mesure où la chrominance apporte en règle générale moins d'information sur le contenu sémantique de l'image que la luminance. De plus ces caractéristiques sont beaucoup moins stables vis-à-vis de la compression. Lors de la vérification de l'intégrité, nous comparons seulement les caractéristiques de luminance, les informations de chrominance servant uniquement à donner un aperçu en couleur des régions manipulées. Dans le premier exemple, nous avons remplacé à l'aide d'un logiciel de retouche d'image le demi-kiwi par un citron. Dans la deuxième image, extraite du film « La guerre des étoiles », le robot R2D2 au premier plan a été tout simplement effacé. Dans les deux cas, les régions altérées ont correctement été identifiées et les informations contenues dans le tatouage ont permis de les restaurer partiellement.

4.5. Remarques concluantes

L'utilisation d'un tatouage robuste pour garantir l'intégrité des images est intéressante à plus d'un titre. D'une part, aucune information autre que la clé et image elle-même n'est nécessaire pour vérifier l'intégrité du contenu d'une image. D'autre part, le tatouage peut être inséré au plus tôt dans le cycle de vie d'une image, c'est-à-dire dès sa création (*e.g.* au niveau de l'appareil photo numérique). Tous les tests ont été réalisés à l'aide du même algorithme de tatouage que celui utilisé pour la protection des droits d'auteurs, avec cependant un paramétrage légèrement différent. Néanmoins, la méthode proposée est relativement indépendante de l'algorithme de tatouage, et d'autres techniques de tatouage peuvent également être utilisées. Ces algorithmes doivent cependant disposer d'une forte capacité d'insertion. La méthode proposée donne dans l'ensemble de bons résultats. Les manipulations visant par exemple à modifier, supprimer ou ajouter des éléments dans une image sont détectées relativement précisément en fonction de la taille des blocs et du nombre de niveaux de quantification choisi. Cette méthode a également l'avantage de permettre une reconstruction partielle des régions qui ont été manipulées. Par contre lorsque l'image subit une attaque photométrique globale ne modifiant pas ou peu son contenu (*e.g.* compression JPEG, filtrage passe-bas, etc.), l'algorithme détecte des erreurs plus ou moins nombreuses en fonction de la sévérité de l'attaque. Ces fausses alarmes sont dues à une mauvaise extraction du tatouage. En effet, bien que disposant d'une bonne capacité d'insertion, l'algorithme de tatouage n'a pas été conçu dans cette optique. Ce-

pendant, l'idée de protéger l'intégrité des images en cachant certaines de leurs caractéristiques dans elles-mêmes ne doit pas être abandonnée pour autant. Cette solution peut être améliorée en utilisant par exemple un algorithme de tatouage principalement robuste aux manipulations photométriques et en particulier à la compression JPEG tout en ayant un « payload » élevé.

5. Signature externe

La deuxième solution que nous proposons pour protéger l'intégrité des images est basée sur l'utilisation d'une signature externe, et plus particulièrement sur une coopération entre un tatouage robuste et une signature stockée en dehors de l'image.

5.1. Principe général

L'idée de base de cette approche (*c.f.* Figure 5.10) est d'extraire des caractéristiques représentatives du contenu de l'image, de la même manière que pour la méthode précédente, mais au lieu de les insérer dans l'image sous la forme d'un tatouage, elles sont stockées séparément sous la forme d'une signature externe. Notre technique se distingue néanmoins des autres approches existantes utilisant une signature externe dans la mesure où la signature n'est pas jointe à l'image, mais enregistrée auprès d'une tierce personne de confiance (*i.e.* entité d'authentification). Seul un identifiant unique permettant de retrouver la signature est caché dans l'image sous la forme d'un tatouage robuste de type « copyright ».

5.2. Protocole d'une application type

Afin d'illustrer le principe de notre méthode, nous avons imaginé une application type (volontairement simplifiée, mais néanmoins réaliste) mettant en scène différents acteurs : l'auteur d'une image (photographe), une entité d'authentification digne de confiance, un utilisateur lambda et un utilisateur malveillant. L'objectif de notre application est de permettre à une agence de reportage de diffuser via Internet des photographies d'actualité prises à l'aide d'appareils numériques « agréés » et de permettre à leurs clients de vérifier l'intégrité des images achetées. Nous nous intéressons ici qu'à l'aspect intégrité, les problèmes commerciaux et la gestion des doits ne sont pas abordés.

5.2.1. Enregistrement d'une image

Le processus d'enregistrement ou de dépôt d'une image par une signature externe se décompose en trois grandes étapes :

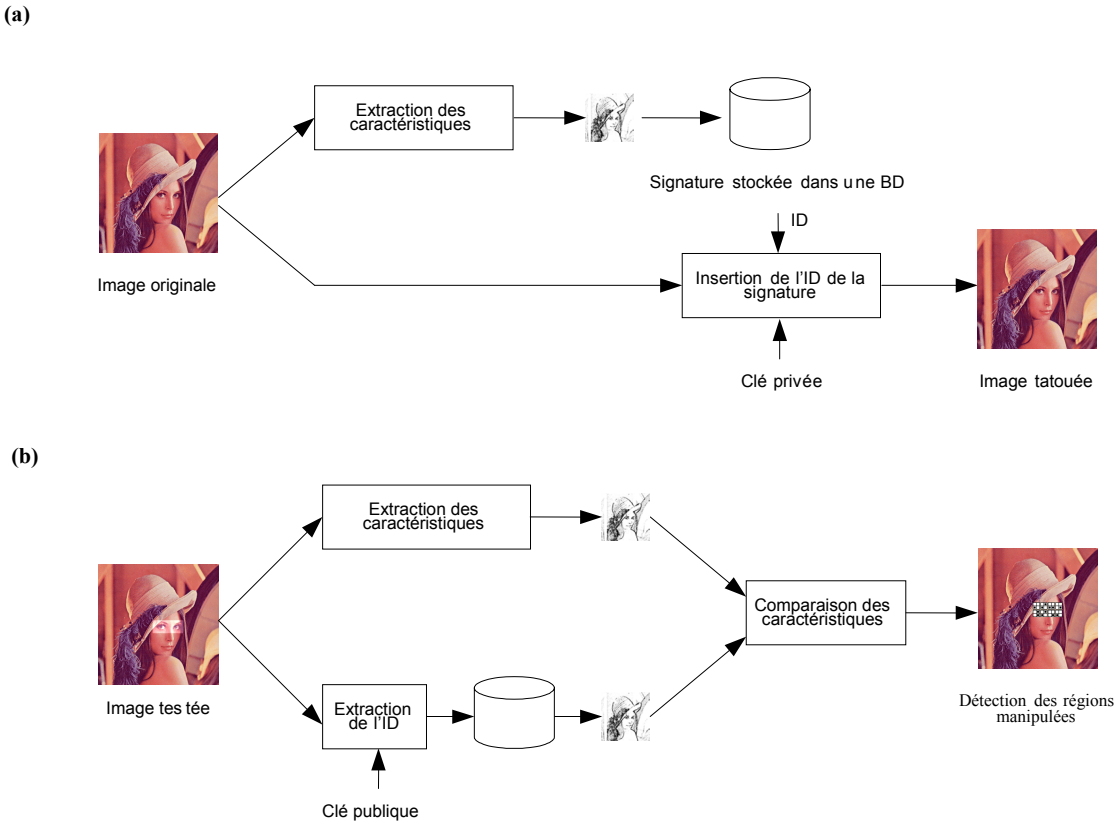


Figure 5.10 – *Principe d'un système d'authentification combinant une signature externe et un tatouage robuste*

- Acquisition de l'image et insertion d'un tatouage fragile (temporaire) au niveau de l'appareil numérique (phases 1 et 2) ;
- Connexion et identification auprès de l'entité d'authentification (phases 3 à 6) ;
- Vérification et protection de l'image déposée (phases 7 à 13).

Acquisition de l'image et insertion d'un tatouage fragile

Pour qu'un service d'intégrité d'images de journalisme soit crédible, les images doivent être protégées au moment même de leur acquisition. En effet, si cela n'est pas le cas, il est alors possible de manipuler des images sans que les manipulations puissent être mises en évidence et de protéger ensuite l'intégrité des images manipulées. Malheureusement, dans le contexte de la protection du contenu d'une photographie numérique par une signature externe, il existe un laps de temps pendant lequel l'image n'est pas protégée. Il est donc nécessaire de trouver une solution afin de garantir que l'image ne subit aucune malversation entre le moment de son acquisition et celui de son dépôt auprès de l'entité d'authentification. Une solution simple consiste par exemple à calculer une fonc-

tion de hachage (*e.g.* CRC, MD5, etc.) à partir des MSB de l'image ; le résultat de cette fonction est ensuite crypté à l'aide d'un algorithme de chiffrement asymétrique de type RSA dont la clé secrète est dépendante de l'appareil numérique, puis inséré dans les LSB de l'image de manière à minimiser la distorsion visuelle. De cette façon, si une personne malintentionnée modifie l'image, de quelque manière que ce soit, le résultat de la fonction de hachage calculé à partir de l'image attaquée ne correspondra plus avec celui contenu dans les LSB et l'image ne pourra pas être enregistrée par l'entité d'authentification (*n.b.* cela implique également que le photographe ne peut ni recadrer, ni compresser la photographie).

Connexion à l'entité d'authentification

Il s'agit d'une phase d'initialisation au cours de laquelle la personne (photographe ou agence photos) désirant enregistrer une image s'identifie auprès du serveur de l'entité d'authentification. L'établissement de la communication est établi classiquement en envoyant le login et le mot de passe. Cette étape permet, d'une part d'identifier la provenance des images, et d'autre par de gérer l'aspect commercial du service (*i.e.* facturation).

Vérification et protection de l'image

Il s'agit de l'étape proprement dite de protection de l'intégrité de l'image. Une fois identifié, le photographe transmet à l'entité d'authentification l'image qu'il souhaite protéger, ainsi que le numéro de série de l'appareil photo numérique qui a servi à prendre la photographie (permettant à l'entité d'authentification de retrouver la clé publique permettant de déchiffrer le tatouage). L'entité d'authentification commence alors par vérifier que l'image reçue n'a subi aucune manipulation depuis sa création à l'aide du tatouage fragile. Si l'image n'est pas intègre, le serveur retourne un message d'alerte à l'utilisateur et met fin au processus d'enregistrement de l'image concernée. Dans le cas contraire, le serveur génère une signature à partir des caractéristiques de l'image. La signature et les dimensions de l'image d'origine sont ensuite enregistrées dans une base de données protégée (la signature n'est jamais rendue publique). L'identifiant (ID de 64 bits) permettant de retrouver la signature, est ensuite inséré dans l'image sous la forme d'un tatouage robuste à l'aide d'une clé unique rendue publique (nous aborderons plus en détail cet aspect au paragraphe 5.2.3). L'image tatouée est finalement retournée à l'utilisateur, terminant le processus de protection de l'image.

5.2.2. Vérification de l'intégrité d'une image

Contrairement à une application de type « copyright », où seul l'auteur ou le propriétaire d'une image est en mesure d'extraire la marque prouvant ses droits, dans le cadre d'un service d'intégrité n'importe quel utilisateur doit pouvoir vérifier que le contenu d'une image n'a pas été modifié.

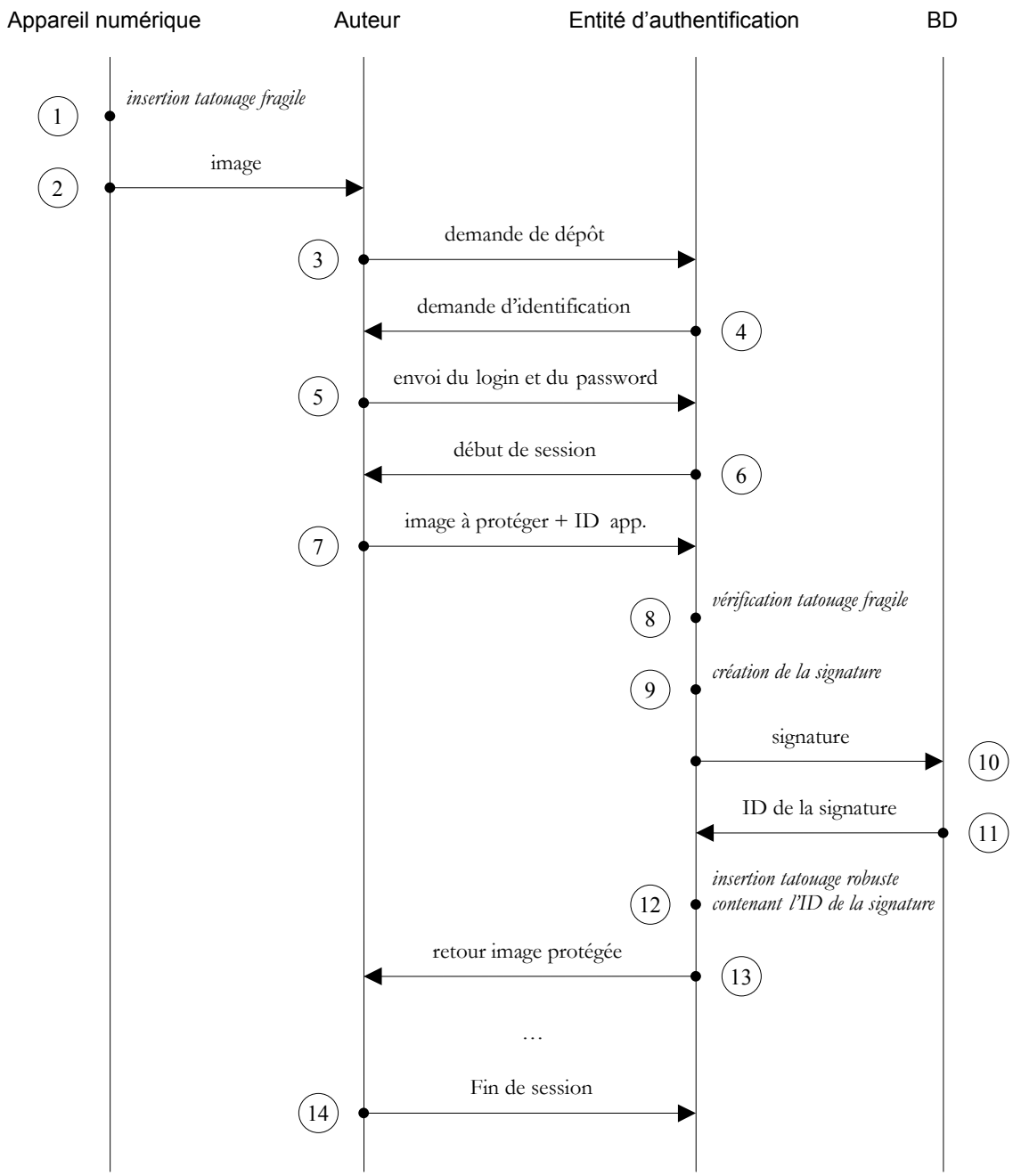


Figure 5.11 – *Protocole de protection d'une image par une signature externe*

Pour cela, la personne qui souhaite vérifier l'intégrité d'une image (*c.f.* Figure 5.12), envoie l'image à tester à l'entité d'authentification (phase 1). Celle-ci extrait dans un premier temps l'ID contenu dans l'image (phase 2), puis recherche dans sa base de données la signature correspondant à l'ID extrait (phase 3). Si aucune signature n'est associée à cet ID, l'image sera alors considérée comme non intègre. Sinon, on procède ensuite à l'extraction des caractéristiques de l'image testée, puis à la

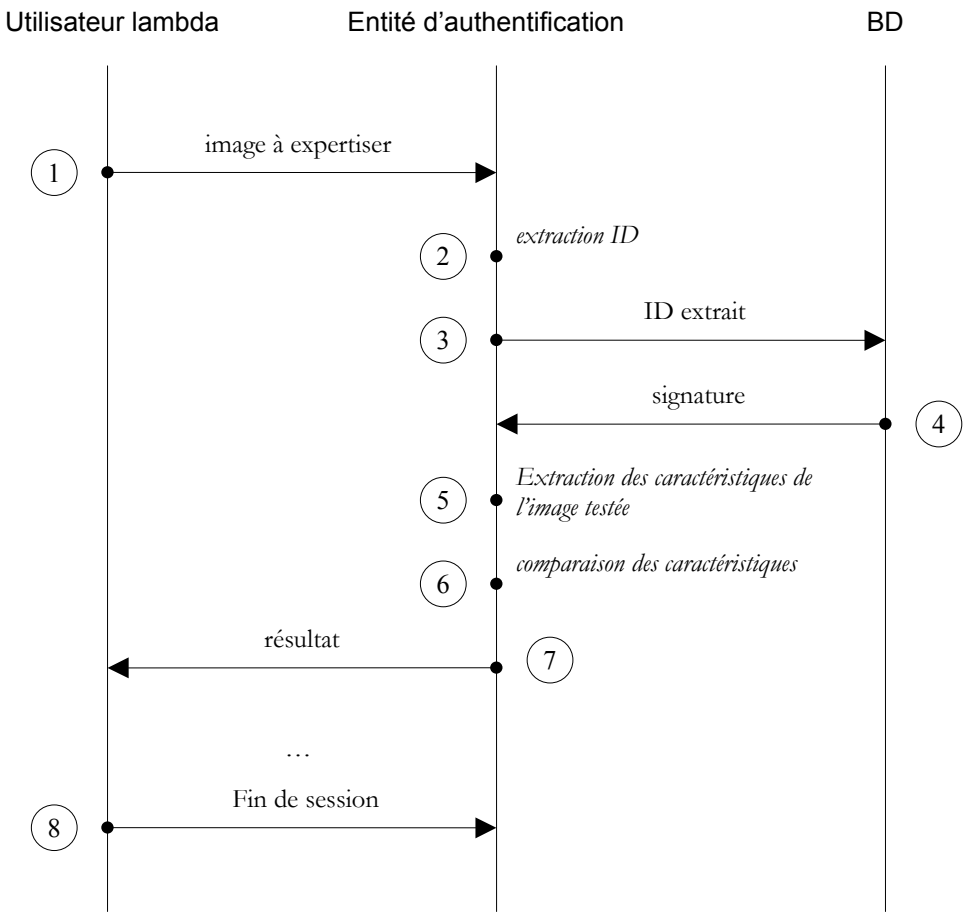


Figure 5.12 – *Protocole de vérification de l'intégrité d'une image à l'aide d'une signature externe*

comparaison de ces caractéristiques avec celles de l'image originale contenues dans la signature (phases 4 à 6). Dans le cas où les caractéristiques sont « identiques », l'image est jugée intègre, sinon on retourne l'image à l'utilisateur en indiquant les régions altérées ou manquantes (phase 7).

5.2.3. Scénarii d'attaque

Il existe deux grands types de schémas d'attaque contre un système d'intégrité d'image. Le premier vise à manipuler le contenu d'une image sans que le système ne décèle la supercherie. Le second consiste à piéger l'algorithme pour lui faire croire que l'image n'est pas intègre, dans le but par exemple de discréditer le fournisseur de contenu. Dans ce paragraphe, nous présentons quelques uns des scénarii d'attaque contre notre système d'intégrité, ainsi que les éventuelles parades à mettre en œuvre. Pour qu'une image falsifiée ne puisse être détectée comme telle par notre méthode, la personne malveillante doit au préalable la faire enregistrer par l'entité d'authentification. Plusieurs scénarii sont envisageables :

Scénario 1 : la personne malveillante manipule l'image à l'aide d'un logiciel de retouche, puis demande simplement à l'entité d'authentification (E.A.) d'enregistrer l'image manipulée. Ce scénario se solde immédiatement par un échec car l'image présentée ne contient évidemment pas de tatouage fragile valide, indispensable pour son enregistrement par l'E.A.

Scénario 2 : ce scénario est similaire au précédent, à la différence près que la personne malveillante tente d'insérer un tatouage fragile pour tromper l'entité d'authentification. Pour que ce scénario puisse marcher, notre pirate doit impérativement connaître l'identifiant et la clé codée au niveau du hardware d'un appareil photo numérique agréé par l'E.A. (selon le principe de Kerckoffs, on fait l'hypothèse que l'algorithme de tatouage fragile ne constitue pas un secret). La viabilité de ce scénario repose donc sur la possibilité de découvrir la clé privée de l'appareil, tâche rendue difficile par l'utilisation d'une puce pour coder la clé. D'autre part, l'E.A peut également vérifier lors de l'enregistrement d'une image la présence d'un tatouage robuste. Si le tatouage d'un ID est détecté, l'image ne sera pas enregistrée, malgré la validité du tatouage fragile. Le pirate doit donc également trouver un moyen de supprimer le tatouage robuste (*c.f.* scénario 5).

Scénario 3 : ce scénario est une variante du scénario 2. Le pirate contourne la difficulté qui consiste à pirater la clé de l'appareil photo, en prenant une photographie de l'image manipulée (préalablement imprimée). L'image ainsi obtenue contient alors un tatouage fragile parfaitement valide. L'inconvénient de cette méthode est qu'elle entraîne une perte de qualité de l'image finale. De plus, le problème lié à la présence du tatouage robuste persiste (*c.f.* scénario 5). Dans l'hypothèse, où le pirate parvient à contourner cette ultime difficulté, l'E.A. peut encore trouver une parade en comparant par exemple l'empreinte de l'image à enregistrer avec les empreintes déjà présentes dans la base de données. S'il existe une empreinte similaire, l'image ne sera pas enregistrée. Cependant, cet ultime contrôle a un coût non négligeable en termes de temps de calcul.

Scénario 4 : pour parvenir à ses fins, le pirate peut toujours tenter de pirater la base de données de l'E.A. Pour parer à cette éventualité, la base de données de l'E.A. doit être installée sur un serveur protégé. Les systèmes de protection de réseaux dépassant le cadre de cette thèse, nous ne présenterons pas les solutions techniques à ces problèmes.

Voici d'autres scénarii dont l'objectif est de supprimer ou de substituer le tatouage robuste de l'ID, dans le but par exemple de discréditer le fournisseur de contenu :

Scénario 5 : ce scénario consiste à supprimer le tatouage contenu dans une image protégée. Pour cela le pirate peut, soit tenter de lessiver l'image à l'aide de techniques de traitement d'image classiques (qui dégradent fortement la qualité), soit essayer d'estimer le tatouage contenu dans l'image pour le soustraire ensuite. L'estimation du tatouage est cependant rendue difficile par le fait que le pirate n'a accès qu'aux images tatouées, contenant de surcroît un ID différent.

Scénario 6 : dans ce dernier cas de figure, le pirate essaie de sur-tatouer une image, dans le but d'insérer un faux ID. Pour que cela fonctionne, il doit connaître la clé secrète utilisée par l'E.A. Comme pour le scénario précédent, il ne peut estimer cette clé à partir des images tatouées. Le seul moyen pour lui de se procurer cette information est d'avoir recours à des méthodes de cryptanalyse du type « brute force attaque ». Cette recherche exhaustive de la clé secrète est matériellement irréalisable pour les systèmes de chiffrement actuels.

5.3. Tests expérimentaux

5.3.1. Caractéristiques utilisées

Contrairement à l'approche précédente où les caractéristiques étaient directement cachées dans l'image elle-même sous la forme d'un tatouage robuste, cette méthode a l'avantage d'être beaucoup moins contraignante en terme de capacité, puisque seul un identifiant de 64 bits est inséré dans l'image, la signature étant stockée séparément de l'image. Cette contrainte relâchée, il est alors possible de mémoriser un plus grand nombre de caractéristiques offrant une meilleure précision et une meilleure sensibilité lors de l'étape de vérification de l'intégrité. Dans ce contexte, la solution idéale serait de stocker directement l'image originale dans la base de données de l'entité d'authentification. Malheureusement cette solution n'est pas envisageable car beaucoup trop coûteuse en termes de capacité d'archivage et de traitements. Pour ces raisons, nous avons choisi de ne stocker qu'une version réduite de l'image (*e.g.* une vignette). La vignette est simplement obtenue par un sous-échantillonnage de l'image originale d'un rapport de 1/16ème.

5.3.2. Comparaison des caractéristiques

La méthode de comparaison des caractéristiques est relativement simple. Cependant, plusieurs cas sont à considérer :

- *L'image n'a subi aucune déformation géométrique* – dans ce cas les deux signatures (originale et testée) ont les mêmes dimensions et peuvent être comparées directement. La comparaison est réalisée en faisant la différence pixel à pixel des deux vignettes. On applique ensuite un seuillage à l'image de différence pour éliminer les petites variations liées par exemple à une compression JPEG ou un filtrage passe-bas, et éviter ainsi des fausses alarmes. Les régions pour lesquelles les pixels de l'image d'erreur sont non nuls, seront considérées comme altérées.

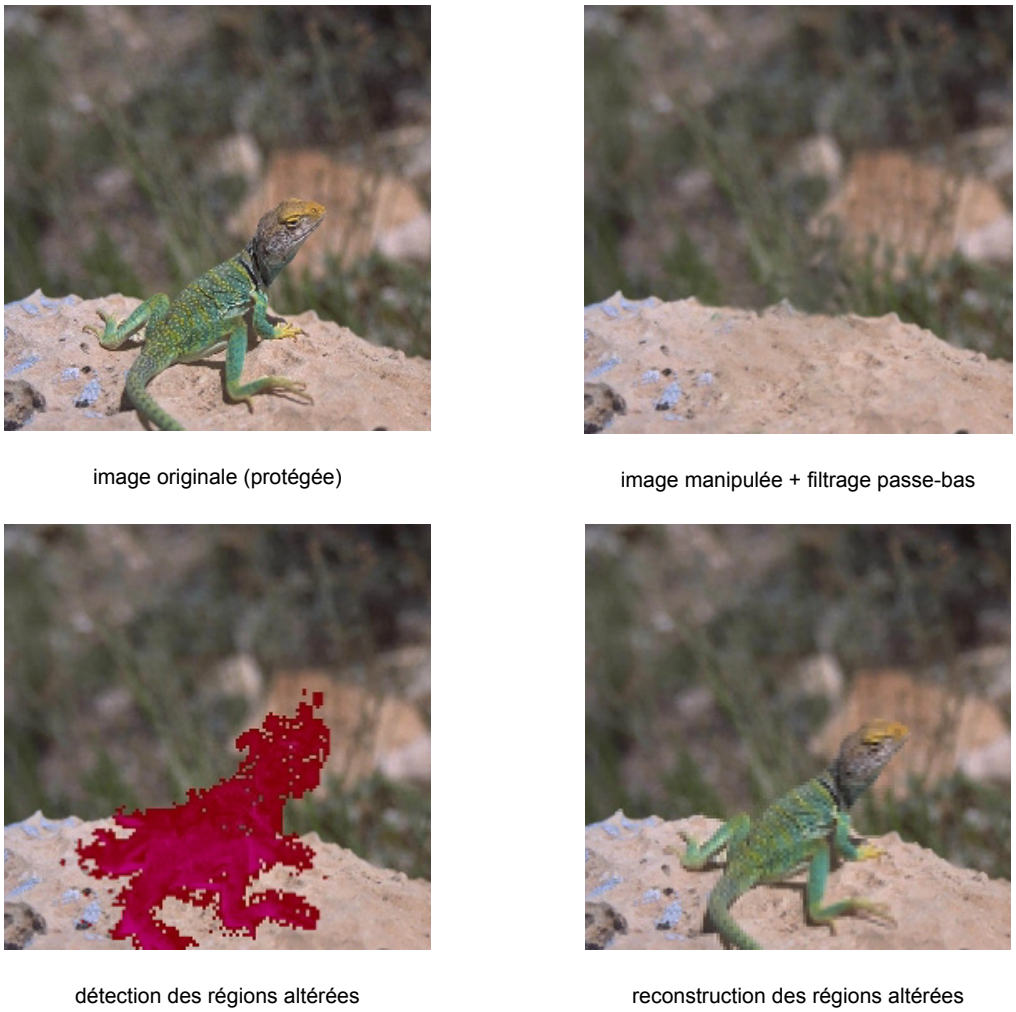


Figure 5.13 – *Exemple de vérification d'intégrité d'une image à l'aide d'une signature externe*

- *L'image a subie une déformation géométrique linéaire* – l'image étant redressée par le processus de resynchronisation, les vignettes peuvent être comparées sans difficulté de la même manière que précédemment.

- *L'image subie un recadrage* – dans ce cas la vignette de l'image testée n'a pas la même taille que celle de l'image l'originale. Pour identifier les régions manquantes de l'image, nous réalisons une recherche par « block matching » entre la vignette testée et l'originale. Une fois l'appariement réalisé, les parties communes aux deux vignettes sont comparées, et les zones manquantes identifiées.

- *L'image subie des déformations géométriques non linéaires* – deux cas sont à considérer. Dans le premier cas le tatouage n'a pas pu être extrait sans erreur, l'intégrité de l'image ne peut pas être vérifier faute de pouvoir retrouver la signature, l'image est donc considérée comme non intè-

Figure 5.14 – *Exemple de détection de la perte d'intégrité par recadrage*

gre. Dans le deuxième cas, le tatouage est extrait correctement, mais les déformations géométriques de l'image n'ont pas été compensées de manière précise ce qui entraîne des fausses alarmes locales (généralement près des contours).

5.3.3. Résultats

Les résultats obtenus par cette méthode sont très bons. Le système parvient à détecter et localiser précisément les zones de l'image qui ont été manipulées (*c.f.* Figure 5.13). La quantité importante d'information contenue dans la signature permet de donner une bonne approximation des régions dont le contenu a été modifié. Contrairement à l'approche utilisant uniquement un tatouage « robuste », cette méthode est très peu sensible à la compression JPEG ; le tatouage contenant l'identifiant de la signature résiste très bien jusqu'à une qualité de compression de 25% et les caractéristiques (*i.e.* luminance moyenne) de l'image compressée sont quasi identiques (à un epsilon près) à celles de l'image originale. En fixant un seuil de tolérance lors de la comparaison des vignettes, on parvient à réduire considérablement le taux de fausses alarmes. De plus, le problème de fausses alarmes liées à une mauvaise extraction du tatouage ne se pose plus ici du fait que les données d'intégrité sont externes à l'image. Par contre, la perte d'un bit de l'ID entraîne une impossibilité de vérification de l'intégrité. Dans l'exemple de la figure 5.14, l'image a été recadrée sur le robot R2D2. Lors de la vérification, le système est parvenu à extraire le tatouage depuis l'image recadrée, puis à identifier et reconstruire les zones manquantes.

6. Conclusion

Dans ce chapitre, nous avons présenté deux solutions recourant à des techniques de tatouage pour la protection du contenu des images. La première méthode utilise un tatouage robuste contenant les caractéristiques de l'image originale. Elle opère en mode d'extraction aveugle (*i.e.* l'image s'auto-suffit) et a l'avantage d'être relativement indépendante de l'algorithme de tatouage. La méthode est efficace face à des manipulations locales de l'image. Les régions manipulées sont détectées avec une bonne précision. Cependant un certain nombre de contraintes techniques restent à résoudre pour pouvoir pleinement tirer partie du potentiel de cette approche. En effet, la principale limitation de notre technique vient de la difficulté de garantir la robustesse de la marque face à des attaques globales de l'image comme une compression JPEG, même en ayant recours à des codes correcteurs d'erreurs. Une solution possible consisterait à recourir à un tatouage « semi-robuste ». L'idée est d'utiliser une technique de marquage principalement robuste aux attaques globales ne modifiant pas le contenu sémantique de l'image. En restreignant le panel des attaques auxquelles l'algorithme doit résister, on peut espérer accroître la capacité du canal de tatouage.

La deuxième méthode que nous avons présentée se différencie de la première dans la mesure où les caractéristiques de l'image originale ne sont plus cachées dans l'image elle-même, mais enregistrées sous la forme d'une signature externe auprès d'une entité d'authentification. Seul un identifiant permettant de retrouver la signature est inséré dans l'image. Cette solution donne de biens meilleurs résultats que la précédente, que ce soit en termes de détections des erreurs et des fausses alarmes, qu'en termes de robustesse face aux attaques de protocole. En contrepartie, elle est bien évidemment moins souple pour l'utilisateur final puisqu'il est nécessaire d'avoir recours à une tierce personne pour vérifier ou protéger le contenu d'une image. Mais peut-on réellement se passer d'une tierce personne pour assurer un service de sécurité d'image ?

Conclusions et Perspectives

1. Conclusions

Dérivées des méthodes classiques de stéganographie, les techniques de tatouage d'image se sont d'abord focalisées sur les problèmes de dégradation de la qualité visuelle de l'image liés à l'ajout d'une information supplémentaire aux pixels. Les progrès se sont ensuite portés dans l'intégration progressive de critères de robustesse. De ce point de vue, la multitude des manipulations possibles, ne serait-ce qu'en termes de traitement d'image, nous incite à penser que cette quête est loin d'être terminée. En effet, au fur et à mesure des progrès accomplis en robustesse par les systèmes de tatouage, de nouvelles attaques plus évoluées et plus efficaces ont fait leur apparition. Nous avons également pris part à cette course-poursuite entre « tatoueurs » et « crakers » en proposant, dans cette thèse, différents moyens à mettre en œuvre pour accroître la résistance globale du tatouage. Nos travaux ont principalement porté sur l'aspect formatage du tatouage, c'est-à-dire sur la manière dont la marque binaire est mise en forme avant d'être insérée dans l'image (*i.e.* ajout de bits de redondance et de bits de resynchronisation). Les choix opérés lors de cette étape ont une influence directe sur la résistance du tatouage face aux diverses manipulations de l'image. Dans ce sens, nous avons proposé d'avoir recours à des codes correcteurs d'erreurs évolués plutôt qu'une simple duplication des bits d'information. Expérimentalement, nous avons montré que la concaténation d'un code produit et d'un code par répétitions permettait d'accroître de manière significative la robustesse globale du tatouage.

Nous avons également proposé deux méthodes pour améliorer la robustesse du tatouage face à des manipulations géométriques de l'image. Nos efforts ont principalement porté sur les déformations géométriques aléatoires générées par l'attaque Stirmark. Ce choix été motivé par le fait qu'il y a trois ans très peu d'algorithmes aveugles (voire aucun) pouvaient prétendre être robustes à cette attaque, constituant ainsi un véritable challenge. La première méthode de resynchronisation présentée dans cette thèse, et qui a fait l'objet d'un brevet, opère directement dans le domaine spatial et ne nécessite aucune information sur l'image originale, ni sur le message caché. Les simulations réalisées

ont montré l'efficacité de notre approche face à des déformations locales ou globales de faibles amplitudes, et en particulier face aux déformations de l'attaque Stirmark. Ce gain important de robustesse a été obtenu sans dégrader la qualité de l'image tatouée (*i.e.* la distorsion reste de l'ordre de 38 dB en moyenne), ni sacrifier la quantité d'information cachée. La deuxième méthode de resynchronisation proposée se veut complémentaire de la première dans la mesure où elle permet uniquement de détecter et compenser les déformations linéaires de l'image qui sont parmi les déformations géométriques les plus couramment usitées.

Enfin, la dernière partie de cette thèse (historiquement le début) a été consacrée à l'étude d'un service de sécurité bien particulier : celui de l'intégrité des images. Ce service est beaucoup moins étudié par la communauté « watermarking » et de nombreuses voies restent à explorer. En effet, les techniques classiquement utilisées en cryptographie bien qu'éprouvées se révèlent inadaptées à la manière que l'on a d'appréhender le contenu sémantique d'une image. Dans ce contexte, nous avons étudié deux approches différentes permettant de protéger et de vérifier l'intégrité des images. La première méthode que nous avons proposée consiste à cacher dans l'image elle-même, sous la forme d'un tatouage robuste, des informations sur son propre contenu. Ces informations sont utilisées *a posteriori* pour vérifier que l'image n'a subi aucune manipulation. Cette approche est la plus intéressante car l'image s'auto-suffit, malheureusement les résultats obtenus ne sont pas entièrement satisfaisants et de nombreux problèmes restent irrésolus (*e.g.* algorithme de tatouage asymétrique notamment). Nous parvenons à détecter et localiser correctement des manipulations locales de l'image, en revanche la quantité très importante d'information cachée dans l'image (de l'ordre de plusieurs milliers de bits) ne permet pas de garantir la robustesse de la marque face à des manipulations globales, comme une compression Jpeg, entraînant ainsi un nombre relativement important de fausses détections. Une solution pour améliorer la robustesse de la marque consiste à réduire sa taille en considérant par exemple des blocs plus grands (au risque de perdre en sensibilité de détection) ou bien encore en ne protégeant que les zones d'intérêt de l'image.

Ces problèmes importants de capacité nous ont amenés à explorer une autre voie. La deuxième méthode que nous proposons combine l'utilisation d'un tatouage robuste et d'une signature ou empreinte externe. Les caractéristiques sont stockées séparément de l'image et sont accessibles uniquement grâce à un identifiant caché dans l'image. La création de la signature, ainsi que le processus d'authentification sont réalisés par une tierce personne de confiance afin de limiter au maximum les risques de malversation tel que l'enregistrement d'une image manipulée. Bien que moins souple d'utilisation pour l'utilisateur lambda (*i.e.* l'image ne s'auto-suffit plus), cette méthode bénéficie d'un fort potentiel et offre de meilleures garanties quant à l'authenticité des caractéristiques de références utilisées lors de l'étape de vérification de l'intégrité de l'image. Au travers de cette étude, nous avons également abordé les problèmes de protocoles liés aux applications assurant des services d'intégrité des images à des fins d'expertise principalement (*e.g.* police, justice, expert automobile, assurance, photo journalisme, etc.). Cet aspect du problème est souvent négligé ou totalement mis de coté, par

de nombreux auteurs alors qu'il constitue la clé de voûte de tout le système. Notre analyse nous porte à croire qu'il est indispensable d'avoirs recours à une tierce personne afin de garantir l'intégrité et l'unicité des signatures.

2. Perspectives

Nos travaux sur l'utilisation de codes correcteurs sont relativement récents. Les résultats obtenus sont très encouragent, néanmoins il pourrait être intéressant de parvenir à estimer, dans le cas de notre algorithme, la capacité théorique du canal de tatouage afin de pouvoir entre autre situer nos résultats par rapport à la limite théorique de Shannon et connaître ainsi la marge de progression restante. La difficulté de cette étude réside principalement dans la modélisation du canal de tatouage. Comme nous l'avons montré au chapitre 3, la distribution du support de tatouage suit une gaussienne généralisée dont les paramètres varient fortement en fonction de l'image considérée.

Toujours dans une optique d'optimisation des performances, nous envisageons de changer la règle de modulation du tatouage avec le support afin d'augmenter d'une part la probabilité d'insertion, et d'autre part de permettre une meilleure maîtrise de la force de tatouage (*e.g.* masque psychovisuel, prise en compte de la couleur, etc.).

Enfin, nous envisageons d'étendre notre algorithme de tatouage à la vidéo. Cette extension paraît naturelle dans la mesure où l'on peut représenter une vidéo comme une succession d'images fixes. Cependant, il faut prendre garde à ne pas tomber dans les différents pièges liés à cette représentation simpliste. La dimension temporelle remet partiellement en cause certains paramétrages de l'algorithme de tatouage, en particulier au niveau des masques psychovisuels en marquant de préférences les zones ou les objets en mouvement plutôt que les plans fixes. Elle permet également d'ajouter un niveau de redondance supplémentaire non négligeable en répétant le tatouage dans les différentes trames de la vidéo. Ce gain en terme d'espace de tatouage est à double tranchant car il ouvre la porte à de nouvelles attaques, en particulier à la collusion intra-vidéo (*i.e.* en considérant les trames tatouées comme des images différentes contenant le même tatouage). D'un point de vue plus pratique, il faudra également prendre en compte les problèmes de complexité et de temps de calcul si on souhaite tendre vers le temps réel, indispensable pour ce type de support.

3. Quel futur pour le tatouage numérique ?

On a assisté en quelques années à une multiplication des algorithmes de tatouage d'image, ainsi qu'à une déclinaison des techniques de tatouage à d'autres média (*e.g.* audio, vidéo, texte, texte formaté, byte-code java, etc.). Le tatouage d'image, malgré un manque de maturité, fait déjà partie des

fonctionnalités proposées par certains logiciels commerciaux. Les nouveaux matériels, comme les lecteurs DVD et certains appareils photo numériques, sont en passe d'intégrer eux aussi des techniques de tatouage, que ce soit pour contrôler le nombre de copies, que pour protéger les droits d'auteur ou le contenu des images. Dans ce sens, la pression de la part des fournisseurs de contenu pour trouver un système permettant de protéger efficacement les droits d'auteur a été un moteur formidable qui a permis au tatouage numérique de devenir en une dizaine d'années un thème de recherche majeur en traitement d'image et de se voir également inscrit au cahier des charges des futures normes de compression. Cet engouement général a surtout permis aux algorithmes de tatouages de faire d'énormes progrès. Malgré cela, les problèmes de normalisation rencontrés et les nombreuses difficultés techniques et juridiques persistantes, ne permettent pas encore la mise en oeuvre à grande échelle de tels systèmes. L'état actuel des choses amène industriels et chercheurs à trouver un second élan en se tournant notamment vers de nouvelles applications où la sécurité ne constitue pas l'enjeu principal. Cela ne signifie pas pour autant que l'aspect robustesse du tatouage n'a plus d'importance, bien au contraire beaucoup de manipulations dites bienveillantes constituent de véritables attaques pour les informations cachées. Même si pour ces futures applications, les données n'ont plus besoin d'être sécurisées d'un point de vue cryptographique, le fait qu'elles puissent rester accessibles constitue une véritable valeur ajoutée. Dans ce sens, les techniques de tatouage restent plus avantageuses que les méthodes de stéganographie classiques ou l'ajout d'informations dans l'en-tête des fichiers.

Annexe – A : Base d’images utilisée

Bibliographie

[BBCP98] F. Bartolini, M. Barni, V. Cappellini and A. Piva. Mask building for perceptually hiding frequency embedded watermarks. *In Proceedings of the International Conference on Image Processing (ICIP'98)*, vol. 1, pp. 450-454, Chicago, Illinois, US, Oct. 1998.

[BBPR98] M. Barni, F. Bartolini, A. Piva and F. Rigacci. Statistical Modeling of full Frame DCT coefficients. *In Proceedings of EUSIPCO'98*, pp. 1513-1516, Rodos, Greece, Sep. 1998.

[BBRP99] M. Barni, F. Bartolini, A. De Rosa and A. Piva. Capacity of the watermarking-channel: how many bits can be hidden within a digital image? *In Proceedings of SPIE, Security and Watermarking of Multimedia Contents, Electronic Imaging'99*, vol. 3657, San Jose, USA, Jan. 1999.

[BCM00] P. Bas, J.-M. Chassery and B. Macq. Robust watermarking based on the warping of pre-defined triangular patterns. *In Proceedings of SPIE Electronic Imaging 2000, Security and Watermarking of Multimedia Contents II*, pp. 99-109, San Jose, CA, USA, Jan. 2000.

[BDS+01] S. Baudry, J.-F. Delaigle, B. Sankur, B. Macq and H. Maître. Analyses of error correction strategies for typical communication channels in watermarking. *IEEE Transactions on Signal Processing*, vol. 81, n° 6, pp. 1239-1250, Jun. 2001.

[BF96] C.J. van den Branden Lambrecht and J.E. Farrell. Perceptual quality metric for digitally coded color images. *In Proceedings of EUSIPCO*, pp. 1175-1178, Trieste, Italy, Sep. 1996.

[BGM95] W. Bender, D. Gruhl and N. Morimoto. Techniques for data hiding. *In Proceedings of the SPIE*, 1995.

[BJM+88] M.F. Barnsley, A. Jacquin, F. Malassenet, L. Reuter and A.D. Sloan. Hernessing chaos for image synthesis. *Computer Graphics*, 22(4):131-140, Aug. 1988.

[BK98] S. Bhattacharjee and M. Kutter. Compression Tolerant Image Authentication. *IEEE International Conference on Image Processing (ICIP'98)*, Chicago, USA, Oct. 1998.

[BMPB01] F. Bartolini, A. Manetti, A. Piva and M. Barni. A Data Hiding Approach for Correcting Errors in H.263 Video Transmitted over a Noisy Channel. *In Proceedings of MMSP 2001*, pp. 59-64, Cannes, France, Oct. 2001.

[BPS01] F. Balado, F. Pérez-González and S. Scalise. Turbo Coding for Sample-level Watermarking in the DCT domain. *In Proceedings of IEEE Conference on Image Processing*, pp. 1003-1006, Thessaloniki, Greece, Oct. 2001.

[BQM95] O. Bruyndonckx, J.-J. Quisquater and B. Macq. Spatial Method for Copyright Labeling of Digital Images. *IEEE Workshop on Nonlinear Signal and Image Processing – NSIP'95*, pp. 456-459, 1995.

[BQR01] T. Brandão, M.P. Queluz and A. Rodrigues. Improving Spread Spectrum Based Image Watermarking Through Non-binary Channel Coding. *In Proceedings of 3rd Conference on Telecommunications*, Figueira da Fez, Apr. 2001.

[BR60] R.C. Bose and D. K. Ray Chaudhuri. On a class of error correcting binary group fields. *Inform. Contr.*, vol. 3, pp. 68-79, Mar. 1960.

[BS98] D. Boneh and J. Shaw. Collusion-Secure Fingerprinting for Digital Data. *IEEE Transactions on Information Theory*, vol. 44, No 5, 1998.

[Cam00] Ça m'intéresse. Pourquoi faut-il se méfier des images ? No 235, Sept. 2000.

[CC98] G.C. Clark and J.B. Cain. Error-Correction Coding for Digital Communication. *Plenum Press*, 1998.

[Cert] CERTIMARK: a benchmark suite for watermarking of visual content and a certification process for watermarking algorithms. http://www.certimark.org

[Cha72] D. Chase. A class of algorithms for decoding block codes with channel measurement information. *IEEE Transactions on Information Theory*, vol. IT-18, Jan. 1972.

[Chec] Checkmark - http://watermarking.unige.ch/Checkmark/index.html

[CKLS97] I.J. Cox, J. Kilian, T. Leighton and T. Shamoon. Secure Spread Spectrum Watermarking for Multimedia. *IEEE Transactions on Image Processing*, vol. 6, No. 12, pp. 1673-1687, 1997.

[CMB01] I.J. Cox, M.L. Miller and J.A. Bloom. Digital watermarking. *Morgan Kaufmann Publishers*, 2001.

[CMYY98] S. Craver, N. Memon, B.-L. Yeo and M. Yeung. Resolving rightful ownerships with invisible watermarking techniques: limitations, attacks, and implementations. *IEEE Journal on Selected Areas in Communications (Special issue on Copyright and Privacy Protection)*, 16(4):473-586, May 1998.

[DBHC99] F. Davoine, P. Bas, P.-A. Hebert and J.M. Chassery. Watermarking et résistance aux déformations géométriques. *CORESA'99*, Sophia-Antipolis, France, Jun. 1999.

[DBQM96] J.-F. Delaigle, J.-M. Boucqueau, J.-J. Quisquarter and B. Macq. Digital images protection techniques in a broadcast framework: overview. *In Proceedings of European Conference on Multimedia Applications, Services and Techniques (ECMAST'96),* pp. 711-728, Louvain-la-Neuve, Belgium, May 1996.

[DDNM98] V. Darmstaedter, J.-F. Delaigle, D. Nicholson and B. Macq. A block based watermarking technique for MPEG-2 signals: Optimization and Validation on real digital TV distribution links. *In Proceedings of European Conference on Multimedia Applications, Services and Techniques (ECMAST'98),* May 1998.

[Digi] DIGIMARC Corp. Identify, Manage and Track your images. http://www.digimarc.com

[DR99a] J.-L. Dugelay and S. Roche. Method for hiding binary data in a digital image. Pending patent PCT/FR99/004485(EURECOM 09-PCT), March 1999.

[DR99b] J.-L. Dugelay and S. Roche. Process for marking a multimedia document, such an image, by generating a mark. Pending patent EP 99480075.3 (EURECOM 11/12 EP), July 1999.

[DSM00] J. Darbon, B. Sankur and H. Maître. Error Correcting Code Performance for Watermark Protection. *In Proceedings of SPIE – Security and Watermarking of Multimedia Contents III*, San Jose, USA, Jan. 2000.

[Dvd97] Watermarking for DVD – CfP – http://www.dvcc.com/dhsg/, 1997.

[DVM97] J.-F. Delaigle, C. De Vleeschouwer and B. Macq. Watermarking using a matching model based on the human visual system. *Ecole thématique CNRS GDR-PRC ISIS : Information Signal Images Marly le Roi*, Apr. 1997.

[DVM98a] J.-F. Delaigle, C. De Vleeschouwer and B. Macq. Psychovisual approach for digital picture watermarking. *Journal of Electronic Imaging*, 7(3): 319-336, May 1998.

[DVM98b] J.-F. Delaigle, C. De Vleeschouwer and B. Macq. Watermarking algorithm based on a human visual model. *Signal Processing*, 66(3) : 319-336, May 1998.

[Eiko] EikonMark from Alphatec ltd. http://www.alphatecltd.com

[Eli54] P. Elias. Error-free Coding. I.R.E. *IEEE Transactions on Information Theory*, vol. PGIT-4, pp.29-37, Sep. 1954.

[Eli55] P. Elias. Coding for noisy channels. *I.R.E. Conv. Rec.*, vol. 3, pp. 37-46, 1955.

[Fan63] R. M. Fano. A heuristic discussion of probabilistic decoding. *IEEE Transactions on Information Theory*, IT-9:64–73, 1963.

[FDM00] J.R. Fernández, J.-F. Delaigle and B. Macq. Improving Data Hiding by Using Convolutional Codes and Soft Decision Decoding. *In Proceedings of SPIE – Security and Watermarking of Multimedia Contents III*, San Jose, USA, Jan. 2000.

[FG99a] J. Fridrich and M. Goljan. Comparing robustness of watermarking techniques. *In Proceedings of SPIE, Electronic Imaging'99, Security and Watermarking of Multimedia Contents*, vol. 3657, Jan. 1999.

[FG99b] J. Fridrich and M. Goljan. Protection of Digital Images using Self Embedding. *The Symposium on Content Security and Data Hiding in Digital Media*, New Jersey Institute of Technology, Mar. 1999.

[Fis94] Y. Fisher. Fractal image compression – Theory and application. Springer-Verlag, N-Y, 1994.

[Fis95] Y. Fisher, editor. Fractal image compression: theory and application. Springer-Verlag, New-York, 1995.

[For66] G.D. Forney. Concatenated Codes. *M.I.T. Press*, 1966.

[Fri98a] J. Fridrich. Image Watermarking for Tamper Detection. *In Proceedings of IEEE International Conference on Image Processing (ICIP'98)*, Chicago, USA, Oct. 1998.

[Fri98b] J. Fridrich. Methods for detecting changes in digital images. *Proceedings IEEE International Conference on Image Processing (ICIP'98)*, Chicago, USA, Oct. 1998.

[Fri99] J. Fridrich. Robust Bit Extraction From Images. *ICMCS'99*, Florence, Italy, june 1999.

[Giov] Giovanni from BlueSpike. http://www.bluespike.com/giovanni/gdigmark.html

[GJ96] A. Glavieux and M. Joindot. Communications numériques – Introduction. *Collection pédagogique de télécommunication, ed. Masson*, 1996.

[HPRN98] J.R. Hernández, F. Pérez-González, J.M. Rodríguez and G. Nieto. The impact of channel coding on the performance of spatial watermarking for copyright protection. *In Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'98)*, vol. 5, pp. 2973-2976, 1998.

[Hut81] J.E. Hutchinson. Fractals and self similarity. *Indiana University Mathematics Journal*, vol. 30 (number 3), 1981.

[HVR01] A. Herrigel, S. Voloshynovskiy and Y. Rystar. The watermark template attack. *In Proceedings of SPIE Electronic Imaging 2001, Security and Watermarking of Multimedia Contents III*, No. paper 4314-46, San Jose, CA, USA, Jan. 2001.

[HW99] C. Heegard and S.B. Wicker. Turbo coding. *Ed. Kluwer Academic Publishers*, 1999.

[Jac90] A. Jacquin. A novel fractal block-coding technique for digital images. *In proceedings of ICASSP*, volume 4, pp. 2225-2228, 1990.

[Jac92] A. Jacquin. Image coding based on fractal theory of iterated contractive image transformations. *IEEE Transactions on Image Processing*, 1(1):18-30, Jan. 1992.

[JGM00] J. Fridrich, M. Goljan and N. Memon. Further Attacks on Yeung-Minzer Fragile Watermarking Scheme. *SPIE International Conference on Security and Watermarking of Multimedia Contents II*, vol. 3971, No 13, San Jose, USA, Jan. 2000.

[JJ98a] N.F. Johnson and S. Jajodia. Exploring steganography: seeing the unseen. *Computer*, 31(2) : 26-34, Feb. 1998.

[JJ98b] N.F. Johnson and S. Jajodia. Steganalysis of images created using current steganography software. *In Proceedings Second International Workshop on Information Hiding*, pp 273-289, Apr. 1998.

[Jpeg00] JPEG 2000. http://www.jpeg.org/JPEG2000.html

[Ker83] A. Kerckoffs. La cryptographie militaire. *Journal des Sciences militaires*, vol. 9, pp. 5-38, 1883.

[KH98] D. Kundur and D. Hatzinakos. Towards a Telltale Watermarking Technique for Tamper-Proofing. *IEEE International Conference on Image Processing (ICIP'98)*, Chicago, USA, Oct. 1998.

[KH98a] D. Kundur and D. Hatzinakos. Digital watermarking using multi-resolution wavelet decomposition. *In Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 6, pp. 2969-2972, 1998.

[KJB97] M. Kutter, F. Jordan and F. Bossen. Digital signature of color images using amplitude modulation. *In Proceedings of SPIE storage and retrieval for image and video databases*, pp 518-526, San Jose, USA, Feb. 1997.

[KL98] T. Kalker and J.-P. Linnartz. Improved watermark detection reliability using filtering before correlation. *In Proceedings IEEE International Conference on Image Processing*, Chicago, USA, Oct. 1998.

[KMKM00] M. Kesal, M.K. Mihçak, R. Koetter and P. Moulin. Iteratively Decodable Codes for Watermarking Applications. *In Proceedings of 2nd International Symposium on Turbo Codes and Related Topics*, Brest, France, Sep. 2000.

[KP99] M. Kutter and F.A. Petitcolas. Fair benchmark for image watermarking systems. *In Proceedings of SPIE, Electronic Imaging'99, Security and Watermarking of Multimedia Contents*, vol. 3657, Jan. 1999.

[Kut98] M. Kutter. Watermerking resisting to translation, rotation and scaling. *In Proceedings of SPIE*, vol. 3528, pp. 423-431, Boston USA, Nov. 1998.

[KVH00] M. Kutter, S. Voloshynocskiy and A. Herrigel. The Watermark Copy Attack. *In Proceedings of SPIE, Security and Watermarking of Multimedia Content II, Vol. 3971*, San Jose, USA, Jan. 2000.

[KZ95] E. Koch and J. Zhao. Towards robust and hidden image copyright labeling. *IEEE Workshop on Nonlinear Signal and Image Processing*, Thessaloniki, Greece, 1995.

[LBC+00] C.-Y. Lin, J.A. Bloom, I.J. Cox, M.-L. Miller and Y.M. Lui. Rotation-scale-, and translation-resilient public watermarking for images. *In proceedings of SPIE International Conference on Security and Watermarking of Multimedia Contents II*, pp. 90-98, San Jose, USA, Jan 2000.

[LC00] C.-Y. Lin and S.-F. Chang. Semi-Fragile Watermarking for Authenticating JPEG Visual Content. *SPIE International Conference. on Security and Watermarking of Multimedia Contents II*, vol. 3971, No 13, San Jose, USA, Jan 2000.

[LC98a] C.-Y. Lin and S.-F. Chang. A Robust Image Authentication Method Surviving JPEG Lossy Compression. *SPIE Storage and Retrieval of Image/Video Database*, San Jose, USA, Jan. 1998.

[LC98b] C.-Y. Lin and S.-F. Chang. A Watermark-Based Robust Image Authentication Using Wavelets. *ADVENT Project Report*, Columbia University, Apr. 1998.

[LC98c] C.-Y. Lin and S.-F. Chang. Generating Robust Digital Signature for Image/Video Authentication. *Multimedia and Security Workshop at ACM Multimedia 98*, Bristol, UK, Sep. 1998.

[LC99] C.-Y. Lin and S.-F. Chang. Distortion Modeling and Invariant Extraction for Digital Image Print-and-Scan Process. *ISMIP 99*, Taipei, Taiwan, Dec. 1999.

[Mpeg97] MPEG-4 Requirements Group – CfP for Identification and Protection of Content in MPEG-4. ISO document JTC1/SC29/WG11 N1714, 1997.

[MSC96] B.S. Manjunath, C. Shekhar and R. Chellappa. A new approach to image feature detection with applications. *Pattern Recognition*, 29(4): 627-640, 1996.

[MT94] K. Matsui and K. Tanaka. Video-steganography: how to secretly embedded a signature in a picture. *Journal of the interactive Multimedia Association Intellectual Property Project*, 1: 187-206, 1994.

[Nas97] C. Nastar. Indexation d'Images par le Contenu : un Etat de l'Art. *CORESA 97*, Issy-les-Moulinaux, France, Mar. 1997.

[Opti] Optimark - http://poseidon.csd.auth.gr/optimark/

[PA98] F.A. Petitcolas and R.J. Anderson. Weakness of Copyright Marking Systems. *In Multimedia and Security – Workshop at ACM MM*, vol. 41, pp. 55-61, Bristol, UK, Sept. 1998.

[Per97] A. Perrig. A copyright protection environment for digital images. *Diploma dissertation*, EPFL, Lausanne, Switzerland, Feb. 1997.

[Pet98] F.A. Petitcolas *et al.* Attacks on copyright marking systems. *In second workshop on information hiding*, 1998.

[PGPJ94] R. Pyndiah, A. Glavieux, A. Picart and S. Jacq. Near-Optimum Decoding of Products Codes. *In Proceedings of IEEE Globecom'94 Conference*, vol. 1, San Francisco, Nov.-Dec 1994.

[PJ96] J. Puate and F. Jordan. Using fractal compression scheme to embed a digital signature into an image. *In Proceedings of SPIE Photonics East Symposium*, vol. 1, Boston, USA, Nov. 18-22 1996.

[PKB99] L. Piron, M. Kutter ad J.M. Boucqueau. Octalis benchmarking: comparisons of three watermarking techniques. *In Proceedings of SPIE, Electronic Imaging'99, Security and Watermarking of Multimedia Contents*, vol. 3657, Jan. 1999

[PP99] S. Pereira and T. Pun. Fast robust template matching for affine resistant image watermarking. *In International Workshop on Information Hiding*, vol. LNCS 1768 of lecture notes in Computer Science, pp. 200-210, Dresen, Germany, Sep. 1999.

[Pro95] J.G. Proakis, Digital Communications, *Third Edition, McGraw Hill*, New York, 1995.

[PSM82] R.L. Pickholtz, D.L. Schilling and L.B. Millstein. Theory of spread spectrum communications – a tutorial. *IEEE Transactions on Communications*, pp. 855-884, 1982.

[PVP00] S. Pereira, S. Voloshynovskiy and T. Pun. Effective Channel Coding for DCT Watermarks. *In Proceedings of IEEE International Conference on Image Processing ICIP'00*, Vancouver, Canada, Sep. 2000.

[Pyn98] R. Pyndiah. Near-Optimum Decoding of Products Codes: Block Turbo Codes. *IEEE Transactions on Communications*, vol. 46, pp. 1003-1010, Aug. 1998.

[Que98] M.P. Queluz. Towards Robust, Content Based Techniques for Image Authentication. *IEEE Signal Processing Society 1998 Workshop on Multimedia Signal Processing*, Dec. 1998.

[RD95] S. Roche and J.-L. Dugelay. Improvements in IFS formulation for its use in still image coding. *In proceedings of IEEE Workshop on Nonlinear Signal and Image Processing*, Halkidiki, Greece, Jun. 1995.

[RD97] S. Roche and J.-L. Dugelay. Mécanismes de sécurité liés à la transmission des images. *CORESA 97*, France, 1997.

[RDDC02] C. Rey, G. Doërr, J.-L. Dugelay and G. Csurka. Toward generic image dewatermarking? *In Proceedings of the International Conference on Image Processing (ICIP'02)*, Rochester, N-Y, Sep. 2002.

[Reut] Agence Reuters http://www.reuters.com

[RM01] D.L. Robie and R.M. Mersereau. Video Error Correcting Using Data Hiding Techniques. *In Proceedings of MMSP 2001*, pp. 59-64, Cannes, France, Oct. 2001.

[Roc99] S. Roche. Mécanismes de sécurité liés à la diffusion des images : tatouage d'image. *Mémoire de thèse*, May 1999.

[Ros95] E. Roskis. Images truquées. *Le Monde Diplomatique*, Jan. 1995.

[RP97] J.J.K Ó Ruanaidh and T. Pun. Rotation, translation and scale invariant digital image watermarking. *IEEE Signal Processing Society 1997 International Conference on Image Processing (ICIP'97)*, vol. 1, pp. 536-539, Santa-Barbara, CA, Oct. 1997.

[RS60] I.S. Reed and G. Solomon. Polynomial codes over certain finite fields. *J. Soc. Ind. Appl. Math.*, vol. 8, pp. 300-304, Jun. 1960.

[RSA77] R.L. Rivest, A. Shamir and L. Adelman. On Digital Signatures and Public Key Cryptosystems. *MIT Laboratory for Computer Science Technical Memorandum 82*, Apr. 1977.

[RY90] K.R. Rao and P. Yip. Discrete cosinus transform: algorithms, advantages, applications. *Academic Press Inc.*, 1990.

[SC96] J. Smith and B. Comiskey. Modulation and Information Hiding in Images. *In Proceedings of the First International Workshop on Information Hiding, Springer Lecture Notes in Computer Science*, vol. 1174, pp. 207-227, 1996.

[Siga] SignIt! from AlpVision. http://www.alpvision.com

[Sigb] Signafy from NEC. http://www.informix.com

[Stir] F.A. Petitcolas and M. Kuhn. Stirmark – Image Watermarking Robustness Test. http://www.cl.cam.ac.uk/~fapp2/watermarking/stirmark/

[STN+01] V. Solachidis, A. Tefas, N. Nikolaidis, S. Tsekeridou, A. Nikolaidis and I. Pitas. A benchmark protocol for watermarking methods. *In Proceedings of the International Conference on Image Processing (ICIP'01)*, 2001.

[STO94] R.G. Van Schyndel, A. Z. Tirkel, and C. F. Osborne. A Digital Watermark. *Proceedings IEEE International Conference on Image Processing (ICIP'94)*, Vol. 2, pp. 86-90, Austin, Texas, Nov. 1994.

[Sure] SureSign from Signum. http://www.signumtech.com

[Sysc] MediaSec Technologies LLC. Software: Syscop. http://www.mediasec.com

[SZT98] M.D. Swanson, B. Zhu and A. Tewfik. Multi-resolution Scene-Based Video Watermarking Using Perceptual Models. *IEEE journal on Selected Areas in Communications*, vol. 16, No. 4, pp. 540-550, May 1998.

[Tal98] TALISMAN ACTS Project AC019f. http://ns1.tele.ucl.ac.be/TALISMAN/ 1998.

[TP00] A. Tefas and I. Pitas. Multi-bit image watermarking robust to geometric distortions. *In Proceedings of SPIE – Security and Watermarking of Multimedia Contents III*, San Jose, USA, Jan. 2000.

[TRS+93] A.Z. Tirkel, G.A. Rankin, R. van Schyndel, WJ Ho, N. Mee and CF Osborne. Electronic Watermark. *Digital Image Computing, Technology and Applications*, pp. 666-672, Sydney Australia, 1993.

[Unzi] UnZign. Is your watermark secure? http://www.altern.org/watermark/

[Vit67] A.J. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, vol. IT-13, pp. 260-269, apr. 1967.

[VPIP01] S. Voloshynovskiy, S. Pereira, V. Iquise, T. Pun. Attack modeling: towards a second generation watermarking benchmark. *IEEE Transactions on Signal Processing nº 81(2001)*, pp. 1177-1214, 2001.

[Wal95] S. Walton. Information Authentification for a Slippery New Age. *Dr. Dobbs Journal*, vol. 20, No. 4, pp. 18-26, Apr. 1995.

[Wat93] A.B. Watson. DCT quantization matrices visually optimized for individual images. *Human Vision, Visual Processing and Digital Display IV, Bernice E. Rogowitz, Editor, Proc. SPIE 1913-14*, 1993.

[WD96] R.B. Wolfgang and E. J. Delp. A watermark for digital images. *In Proceedings of the 1996 International Conference on Image Processing*. Vol. 3, pp. 219-222, Lausanne, Switzerland, Sept. 1996.

[WD99] R.B. Wolfgang and E. J. Delp. Fragile Watermarking Using the VW2D Watermark. *SPIE International Conference. on Security and Watermarking of Multimedia Contents*, vol. 3657, No. 22, EI '99, San Jose, USA, Jan. 1999.

[Win98] S. Winkler A perceptual distortion metric for digital color images. *In Proceedings of the International Conference on Image Processing (ICIP'98)*, vol. 1, pp. 399-403, Chicago, Illinois, US, Oct. 1998.

[WJ98] H.-J. Wang and C.-C Jay Kuo. Image protection via watermarking on perceptually significant wavelet coefficients. *In Proceedings of IEEE Multimedia Signal Processing Workshop (MMSP'98),* pp. 279-284, Redondo Beach CA USA, Dec. 1998.

[WL98] M. Wu and B. Liu. Watermarking for Image Authentication. *In Proceedings of IEEE International Conference on Image Processing (ICIP'98)*, Chicago, USA, Oct 1998.

[WLB95] S. Western, R. Lagendijk and J. Biermond. Perceptual image quality based on a multiple channel HVS model. *In Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'95)*, vol. 4, pp. 2351-2354, 1995.

[XBA98] X.-G. Xia, C.G. Boncelet and G.R. Arce. Wavelet transform based watermark for digital images. *Optis Express*, 3(12): 497-511, Dec. 1998.

[YM97] M.M. Yeung and F. Mintzer. An Invisible Watermarking Technique for Image Verification. *In Proceedings of IEEE International Conference on Image Processing*, Santa Barbara, USA, Oct. 1997.

[Zha96] J. Zhao. A WWW service to embed and prove digital copyright watermarks. *In Proceedings of European Conference on Multimedia Applications, Services and Techniques (ECMAST'96)*, 1996.

[ZLL99] W. Zeng, B. Liu and S. Lei. Extraction of multi-resolution watermark images for claiming rightful ownership. *In Proceedings of SPIE, Electronic Imgaging'99, Security and Watermarking of Multimedia Contents*, vol. 3657, San-Jose, USA, Jan. 1999.

Publications et Brevets

[DRR99a] J.-L. Dugelay, C. Rey and S. Roche. Introduction au tatouage d'image. *Journées d'études et d'échanges en Compression et Représentation des Signaux Audiovisuels*, Sophia-Antipolis, France, Jun. 1999.

[DRR99b] J.-L. Dugelay, C. Rey and S. Roche. A Fractals-Inspired Approach to Data Embedding Digital Images for Autentication Services. *In proceedings of IEEE-MMSP'99*, Copenhagen, Denmark, Sept. 1999.

[RD00a] C. Rey and J.-L. Dugelay. Blind Detection of Malicious Alterations On Still Images Using Robust Watermarks. *IEE Secure Images and Image Authentication colloquim*, London, UK, Apr. 2000.

[RD00b] C. Rey and J.-L. Dugelay. Image watermarking for Owner and content authentication. *ACM Multimedia – Technical demomstration*, Los Angeles, California, Nov. 2000.

[RD00c] C. Rey and J.-L. Dugelay. Analyse de l'adéquation ou de l'adaptation du tatouage pour l'intégrité des images, *Délivrable projet RNRT – Aquamars D 6.1*, Mar. 200.

[[GD+01] G. Csurka, J.L. Dugelay, C. Mallauran, J.P. Nguyen and C. Rey. Attaque malveillante d'images tatouées basées sur l'auto-similarité. *Journées d'études et d'échanges en Compression et Représentation des Signaux Audiovisuels*, Dijon, France, Nov. 2001.

[RD02a] C. Rey and J.-L. Dugelay. Un panorama des méthodes de tatouage permettant d'assurer un service d'intégrité. *Revue Traitement du Signal, numéro spécial*, vol. 18, n° 4, France, Jun. 2002.

[RDDC02] C. Rey, G. Doerr, J.-L. Dugelay and G. Csurka. Toward Image Dewatermarking. *In proceedings of ICIP'02*, Rochester, N-Y, Sep. 2002.

[RD02b] C. Rey and J.-L. Dugelay. A survey of watermarking algorithms for image authentication. *EURASIP Journal on Applied Signal Processing*, Jun. 2002.

[PDG+02] G. Petitjean, J.-L. Dugelay, S. Gabriele, C. Rey and J. Nicolaï. Toward real-time video watermarking for system-on-chip. *IEEE International Conference on Multimedia and Expo*, Lausanne, Suisse, Aug. 2002.

[RD+03] C. Rey, K. Amis, J.-L. Dugelay, R. Pyndiah and A. Picart, Enhanced robustness in image watermarking using block turbo codes, *In Proceedings of SPIE, Electronic Imgaging'02, Security and Watermarking of Multimedia Contents*, Jan. 2003.

[DR01] J.L. Dugelay and C. Rey. Method of marking a multimedia document having improved robustness, *Pending Patent EUP 99480753 (EURECOM 14 EP)*, May 2001.