République Algérienne Démocratique et populaire
Ministère d'Enseignements Supérieurs et de la Recherche Scientifique

UNIVERSITE MOHAMED KHIDER BISKRA
FACULTE DES SCIENCES ET TECHNOLOGIE
DEPARTEMENT GENIE-ELCTRIQUE

THESE

Présentée pour l'obtention du

Diplôme de Doctorat en Sciences en Electronique

Par :

ZITOUNI Athmane

Thème :

Ondelettes et techniques de compression d'images numérique

À soutenir devant la commission d'examen :

Pr. Noureddine Djeddi	Président	Prof	Université de Biskra
Dr. Zine-Eddine Baarir	Rapporteur	M.C.A	Université de Biskra
Pr. Malika Mimi	Examinateur	Prof	Université de Mostaganem
Pr. Abdelmalik Taleb-Ahmed	Examinateur	Prof	Université de Valenciennes
Pr. Youcef Ferdi	Examinateur	Prof	Université de Skikda
Dr. Salim Sbaa	Examinateur	M.C .A	Université de Biskra

2012/2013

Résumé

Cette thèse est consacré à l'étude des transformées appliquées dans la littérature (ondelettes, transformée directionnelle, fourrier …) dans le contexte de la compression d'images numériques. Nous abordons aussi l'étude des méthodes principale de codage utilisées dans la compression d'images comme (le codage de Shanon Fano, Huffman, jepg2000, les codeurs hiérarchiques…).

Nous présentons l'influence des nouvelles propriétés mathématiques apportées par la théorie des ondelettes dans le domaine du codage hiérarchique en vue de l'application à la compression d'images numériques. On montre, par une analyse théorique que la décomposition multi résolution de l'image, que l'apport pratique de la théorie des ondelettes est nécessaire. Pour cela nous étudions les techniques de compression des images numérique. L'intérêt de l'analyse multi résolution résulte dans sa décomposition en structures pyramidales.

L'emploi des codeurs hiérarchique est basé sur la notion d'arbre de zéros (zerotree). Nous proposons une nouvelle approche de compression d'images basée sur le principe de base de l'algorithme SPIHT. On constate que notre nouvelle approche notée MSPIHT (Modified SPIHT) consiste à minimiser les bits à coder après quantification. On cherche pour cela à coder plusieurs coefficients à l'aide d'un seul bit qui devient suffisant pour notre approche alors que pour la méthode de base (SPIHT) cela ne l'est pas. Les résultats obtenus par cette nouvelle approche que nous proposons en se basant sur la métrique suivante :

- PSNR
- Taux de compression

Son meilleurs que les résultats obtenus par la méthode de base (SPIHT de Amir SAID) [36]. Notre contribution est meilleurs surtout pour les moyens et hauts débits sans pour autant affecter le temps de calcul. Enfin, nos résultats sont comparables à ceux obtenus par les algorithmes SPIHT, EZW et JPEG 2000.

Mots-clés : *Compression d'images, SPIHT, MSPIHT, Entropie, Codage, PSNR, Taux de compression, SPIHT, EZW, JPEG2000.*

Abstract

This thesis is devoted to the study of transforms applied in the literature (wavelet transform directional, fourrier ...) in the context of digital image compression. We also address the study of main coding methods used in image compression as (Shannon Fano coding, Huffman jepg2000, hierarchical coders ...).

We present the influence of new mathematical properties provided by the wavelet theory in the field of hierarchical coding for application to digital image compression. We show by theoretical analysis that the multi-resolution decomposition of the image, the practical contribution of the wavelet theory is needed. Therefore, we study the techniques of digital image compression. The advantage of multi-resolution analysis results in its decomposition into pyramids.

The use of hierarchical coders is based on the concept of zero tree (zerotree). We propose a new approach for image compression based on the basic principle of the SPIHT algorithm. We note that our new approach denoted MSPIHT (Modified SPIHT) is to minimize the bit code after quantization. We are looking for it to encode several coefficients using a single bit which is sufficient for our approach as for the basic method (SPIHT) it is not. The results obtained by this new approach we propose based on the following metric:

• PSNR

• Compression ratio

Its better than the results obtained by the basic method (SPIHT Amir SAID) [36]. Our contribution is best especially for medium and high speeds without affecting the computation time. Finally, our results are comparable to those obtained by the algorithms SPIHT, EZW and JPEG 2000.

Keywords: Image Compression, SPIHT, MSPIHT, entropy coding, PSNR, compression ratio, EZW, JPEG2000.

ملخـــــص

في إطار هذه الرسالة قمنا بدراسة برامج ضغط الصورة المعتمدة على المويجات , التحويل المتجه ,فوريه في سياق ضغط الصور الرقمية . ونتناول دراسة أساليب الترميز الرئيسية المستخدمة في ضغط الصور " شانون فانو ، هوفمانjepg2000 ، المبرمج الهرمي" نقدم تأثير الخصائص الرياضية الجديدة التي تقدمها نظرية المويجات في مجال الترميز الهرمي للتطبيق على ضغط الصور.

وتبين لنا من خلال التحليل النظري أن تحليل متعدد القرار للصورة، هناك حاجة إلى مساهمة لنظرية المويجات.

يستند استخدام المبرمج الهرمي على مفهوم شجرة صفر (zerotree) لذلك نقترح مقاربة جديدة لضغط الصور على أساس المبدأ الأساسي للخوارزمية SPIHT ونلاحظ أن لدينا نهج جديد يرمز MSPIHT (معدلة SPIHT) وهو وتقليل عدد الرموز بعد التكميم, نحن نبحث عن ذلك لترميز العديد من معاملات باستخدام بت واحد وهو ما يكفي لنهجنا أما بالنسبة للطريقة الأساسية (SPIHT) فهو ليس كذلك.

النتائج التي حصل عليها هذا النهج الجديد و نقترح على أساس الخصائص التالية:

- نسبة تشويه الصورة
- نسبة الضغط

على نحو أفضل من النتائج التي تم الحصول عليها بواسطة الطريقة الأساسية (SPIHT أمير سعيد). [36] مساهمتنا هي أفضل وخاصة بالنسبة للتدفقات المتوسطة و العالية دون التأثير على الوقت اللازم للحساب أخيرا نتائجنا هي مماثلة لتلك التي حصل عليها كل من خوارزميات SPIHT ، EZW و2000 JPEG.

كلمات البحث :

ضغط الصور ، الانتروبيا، الترميز، نسبة تشويه الصورة ، MSPIHT، SPIHT، نسبة ضغط JPEG2000، EZW

Remerciements

Je tiens à remercier Dr. BAARIR Zine-Eddine d'avoir dirigé mes travaux de thèse et de son soutien durant tous mes années de thèse.

Je remercie Docteur KHELIFA Ali pour son aide précieuse.

Je remercie tout particulièrement Pr. TALEB-AHMED Abdelmalik, Directeur de Recherche au laboratoire LAMIH, Valenciennes, France, de m'avoir accueilli dans son équipe, et de son soutien moral au cours de mes travaux de thèse.

Je remercie Pr Djeddi Nourredine qui me fait l'honneur de présider ce jury.

Je remercie Pr. Malika Mimi, Pr Ferdi Youcef et Docteur Sbaa Salim d'avoir accepter de juger ce travail.

Je tiens également à remercier mes collègues du département de Génie Electrique et tous mes amis.

Dédicace

A mes chères parents,

A ma femme et mes enfants Meriem- Moundir,

A mes frères et sœurs et ces enfants

A toute ma famille,

A tous mes amis,

Table des matières

Chapitre 2 : Analyse multiresolution.

Chapitre 3 : Codeur SPIHT .Proposition d'une Optimisation MSPIHT

Chapitre 4 : Résultats et Discussions

Liste Des Figures

Liste Des Tables

Introduction générale

Depuis l'avènement de l'ère numérique, la quantité d'information échangée sur les réseaux du monde entier nécessite un traitement de l'information, visant à réduire la taille des données, tout en conservant leur intégrité. La compression numériques devient alors une nécessité afin d'assurer l'archivage d'une part et de faciliter leur transmission d'autre part.

La formidable avancée technologique de ces dernières années, la baisse des coûts du matériel et l'accès à des plateformes performantes pour le grand public incite les fournisseurs de contenus à proposer de plus en plus de choix dans le mode de transmission pour l'image numérique. En outre, l'augmentation des débits dans les réseaux mondiaux a permis le développement d'applications comme la télévision sur internet. Parallèlement, des efforts sont entrepris pour diminuer la taille des données transmises, aussi bien pour le multimédia (vidéos, images, musiques, sons...) que pour les données classiques (documents, pages web...). Les données les plus volumineuses restent à ce jour les images et les contenus vidéo, qu'il convient de traiter en tenant compte de leurs spécificités. Le principe général de toute compression est la réduction de la taille des données, en construisant des messages qui résumeront l'information initiale par élimination des redondances.

La compression de l'information peut être obtenue par une transformation des données en les projetant sur une base de fonctions orthogonales. Et ensuite réaliser un codage de cette transformée. Les Transformées en Ondelettes Discrètes (DWT) ont gagné un intérêt considérable pour le traitement des signaux. Elles permettent une représentation du signal par un nombre limité de coefficients tout en localisant les discontinuités avec précision.

Dans cette thèse, nous nous intéresserons principalement au premier de ces problèmes, qui est celui de la représentation des images numériques. Dans la plupart des cas, les problèmes liés à sa transmission ou à son stockage sont supposés indépendants et relèvent du domaine des télécommunications. La compression consiste à chercher comment décrire de manière la plus succincte possible l'information, en s'autorisant éventuellement à la dégrader. Ce traitement permet non seulement de réduire le nombre d'éléments nécessaire pour la représenter, mais également de simplifier les traitements ultérieurs en condensant l'information. Dans le cas particulier de l'image, la compression s'attache à extraire l'information visuelle des données brutes reçues de la caméra. Comme le simple fait de numériser l'image la dégrade, un autre

objectif essentiel de la compression est de trouver le meilleur compromis entre la quantité d'information conservée et l'impact visuel des dégradations apportées sur l'image. Enfin, du fait de la masse des données à traiter, un compromis est souvent nécessaire entre la complexité des opérations effectuées et la qualité des résultats obtenus.

Notre travail dans le cadre de cette thèse consiste en l'étude et l'utilisation de la transformées en ondelettes pour les images fixe et la proposition d'une optimisation de l'algorithme SPIHT nommée MSPIHT (Modified-SPIHT). Le principe général de notre méthode consiste à minimiser les bits à coder. On cherche à coder plusieurs coefficients à l'aide d'un seul bit qui devient suffisant alors que pour la méthode de base (SPIHT) cela ne l'est pas. Les résultats obtenus par notre nouvelle approche en termes de PSNR et de taux de compression obtenus sont meilleurs surtout pour les moyens et hauts débits sans pour autant affecter le temps de calcul. Enfin, nos résultats sont comparables à ceux obtenus par les algorithmes SPIHT, EZW et JPEG 2000.

Notre thèse est organisée selon les chapitres suivants :

Dans le premier chapitre, nous décrivons le principe général de la compression, (transformation, quantification et codage) ; ensuite, nous décrivons la classification des méthodes de codage qui peuvent être regroupées en deux sous-classes :

- Les méthodes sans pertes d'information : codage de shannon-fano, codage de Hufmann et codage arithmétique.

- Les méthodes avec pertes d'information : quantification vectorielle, méthodes prédictives, méthodes hybrides, JIBG, codage par transformée et codage imbriqué.

Dans le deuxième chapitre, nous détaillons les principales transformées que l'on trouve dans la littérature. Nous commençons par décrire les transformée directionnelles (Radon, Ridgelets, curvelets, bruchelets, Bandelettes…) et nous insistons sur les limitations de ces transformées dans leur usage pour le codage imbriqué. Ensuite, nous montrons les caractéristiques principales de la transformée de Fourier et ses limitations d'utilisation dans le cadre du codage imbriqué. La fin de ce chapitre est consacrée à la transformation en ondelettes continues et ondelettes discrètes et l'analyse multirésolutions en particulier. L'idée étant de montrer son intérêt lorsqu'elle est utilisée pour le codage imbriqué.

Le troisième chapitre décrit l'algorithme SPIHT avec un certain nombre d'exemples d'utilisation et détaille notre contribution appelée MSPIHT, qui consiste en l'optimisation de l'algorithme SPIHT,

Enfin, dans le dernier chapitre, nous présentons les résultats obtenus avec différentes images de test. Nous comparons plusieurs codeurs (JPEG2000, EZW, SPIHT) avec notre codeur MSPIHT. On fait aussi une analyse des résultats obtenus ainsi qu'une étude comparative en se basant sur la métrique suivante ; le taux de compression et le PSNR.

On termine notre thèse par une conclusion générale et les perspectives envisagées concernant notre travail.

Chapitre 1: Généralités sur la compression d'images

Résumé

Le but de ce chapitre est la présentation des principales méthodes de codage proposées dans la littérature. On commence par donner la définition et le schéma général de la compression d'images numériques. Ce schéma débute par l'étape de la transformation qui a pour but de réduire le volume d'informations en effectuant une opération de décorrélation des pixels. Ensuite on passe à l'étape de quantification pour réduire le nombre de bits nécessaires pour les représentations des coefficients enfin le codage de ces coefficients. Ensuite, on propose une classification des méthodes de codage que l'on peut regrouper en deux classes :

- Les méthodes sans pertes d'informations comme (codages de Shannon-Fano, codages de Huffman, codages arithmétiques, les méthodes statistiques, les méthodes de plage.

- Les méthodes avec perte d'informations comme (quantification vectorielle, méthodes de transformations, méthodes prédictives, méthodes hybrides et codeurs JPEG, JIBG).

Enfin, on explique quelques méthodes de codages qui se basent sur les codages hiérarchiques (EZW, SPECK), ces méthodes vont nous servir par la suite de base de comparaison et de base d'amélioration qui nous a conduit au chapitre trois à proposer notre contribution.

1. Introduction

Pour l'utilisation des images numériques, il faut compresser les fichiers dans lesquels elles sont enregistrées. L'image consomme une quantité impressionnante d'octets quand elle est numérisée. Pour s'en rendre compte, deux exemples suffisent. Aujourd'hui, on parle de " qualité megapixel " pour les appareils photo numériques; cela signifie que chaque image comporte environ un million de pixels, chaque pixel nécessitant trois octets pour les composantes RVB (rouge, vert, bleu). Sans compression, cela représenterait un peu plus de 3 Mo pour une seule photographie. L'équivalent d'une pellicule de trente-six poses occuperait ainsi 100 Mo !

Pour remédier à ces contraintes, il n'y a qu'une solution: comprimer les images. Les chercheurs ont imaginé de nombreuses méthodes de compression, que l'on classe en deux catégories: celles qui se contentent de comprimer les données sans les altérer, et celles qui les compactent en les modifiant. Les premières, dites non destructives, permettent de reconstituer, au bit près, le fichier dans l'état où il était avant la compression (ce qui est indispensable pour du texte).Les deuxièmes son les méthodes destructives, elles altèrent irrémédiablement les données de départ. Et ce sont elles qui servent presque toujours à comprimer les images (fixes ou animées) sur les CD-ROM, sur les chaînes numériques et sur Internet.

Un moyen de réduise le volume global des images tout en conservant l'image originale, consiste en la compression des images avec le minimum de dégradation et le maximum d'efficacité possible.

2. Définition de la compression d'image

Les méthodes de compression et de codage réduisent le nombre de bits par pixel à stocker ou à transmettre, en exploitant la redondance informationnelle dans l'image [1].

Les principaux critères d'évaluation de toute méthode de compression sont :

- La qualité de reconstitution de l'image.
- Le taux de compression.
- La rapidité du codeur et décodeur (codec).

3. Schéma fonctionnel de la compression des images

Le schéma fonctionnel de la compression est présenté dans la figure1.1 ci-dessous :

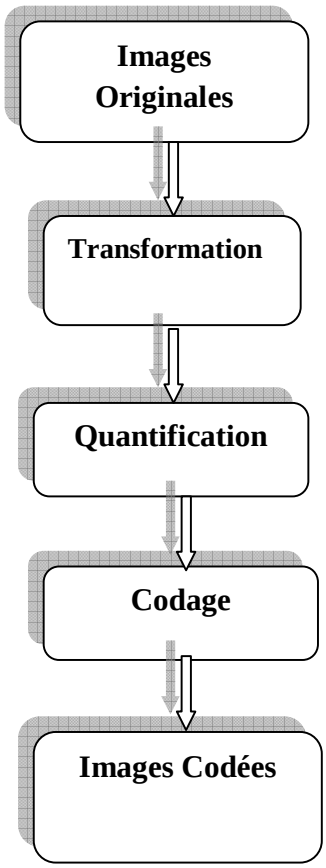


Figure1.1 : schéma d'un codeur source

A partir de ce schéma, nous allons revoir chacune de ses étapes à fin de préciser leur rôle.

3.1 Transformation

La dépendance existante entre chacun des pixels et ses voisins (la luminosité varie très peu d'un pixel à un pixel voisin) traduit une corrélation très forte sur l'image. On essaie donc de tirer partie de cette corrélation, pour réduire le volume d'information en effectuant une opération de décorrélation des pixels.

La décorrélation consiste à transformer les pixels initiaux en un ensemble de coefficients moins corrélés, c'est une opération réversible.

3.2 Quantification

La quantification des coefficients a pour but de réduire le nombre de bits nécessaires pour leurs représentations. Elle représente une étape clé de la compression. Elle approxime chaque valeur d'un signal par un multiple entier d'une quantité q, appelée quantum élémentaire ou pas de quantification. Elle peut être scalaire ou vectorielle. Un des résultats fondamentaux des travaux de Shannon concernant la relation : (débit /distorsion) montrent que l'on obtient de meilleures performances en utilisant la quantification vectorielle.

3.3 Codage

Une fois les coefficients quantifiés, ils sont codés. Un codeur doit satisfaire a priori les deux conditions suivantes :

- ✓ Unicité : deux messages différents ne doivent pas être codés de la même façon.
- ✓ Déchiffrabilité : deux mots de codes successifs doivent être distingués sans ambiguïté.

A partir de ce schéma fonctionnel qui nous avons explicite, nous passons à la classification des méthodes de compressions d'image numériques.

4. Classification des méthodes de compression

La plupart des méthodes de compression visent à enlever la redondance présente dans l'image de manière à diminuer le nombre de bits nécessaires à sa représentation [2].

Plusieurs types de redondance en termes de corrélation peuvent être considérés :

- ✓ La redondance spatiale entre pixels ou blocs voisins dans l'image.
- ✓ La redondance temporelle entre images successives dans une séquence vidéo.

Les méthodes de compression peuvent se regrouper, en deux classes :

- ✓ Les méthodes sans perte d'informations (sans distorsion ou réversible).
- ✓ Les méthodes avec perte d'informations (avec distorsion ou irréversible).

Les expérimentations menées montrent que généralement les méthodes qui atteignent des taux de compression très élevés sont les méthodes avec distorsion. Par contre, les méthodes

sans distorsion engendrent des taux de compression très faibles et ne sont utilisées que dans des applications sensibles telles que les images médicales et les images satellites.

4.1 Méthodes sans distorsion des données

Elles permettent de retrouver exactement les pixels de l'image numérique originale.

4.1.1 Codage de Shannon-Fano

Shannon du laboratoire Bells et R. M. Fano du MIT ont développé à peu près en même temps une méthode de codage basée sur de simples connaissances de la probabilité d'occurrence de chaque symbole dans le message [3].

Le procédé de Shannon-Fano construit un arbre descendant à partir de la racine, par divisions successives. Le classement des fréquences se fait par ordre décroissant, ce qui suppose une première lecture du fichier et la sauvegarde de l'en-tête.

Le principe est le suivant :

- ✓ Classer les *n* fréquences non nulles $\{f_i\}$ par ordre décroissant.
- ✓ Répartir la table des fréquences en deux sous tables de fréquences proches. Poursuivre l'arborescence jusqu'à ce que toutes les fréquences soient isolées.
- ✓ Attribuer dans l'arborescence le bit 0 à chaque première sous table.
- ✓ Attribuer aux symboles les codes binaires correspondant aux bits de description 1 de l'arborescence.

4.1.2 Codage de Huffman

Le codage de Huffman [4] crée des codes à longueurs variables sur un nombre entier de bits. L'algorithme considère chaque message à coder comme étant une feuille d'un arbre qui reste à construire. L'idée est d'attribuer aux deux messages de plus faibles probabilités, les mots codés les plus longs. Ces deux mots codés ne se différencient que par leur dernier bit. Contrairement au codage de Shannon-Fano qui part de la racine des feuilles de l'arbre et, par fusions successives, remonte vers la racine.

Le principe est le suivant :

- ✓ Répartir les fréquences f_i des lettres.

✓ Classer les symboles dans l'ordre décroissant des fréquences d'occurrence. Le résultat de l'algorithme ne change donc pas si l'on remplace les fréquences f_i par les probabilités $p_i = f_i \Big/ \sum f_i$.

✓ Regrouper par séquences les paires de symboles de plus faible probabilité, en les reclassant si nécessaire. Plus précisément : calculer $s = fi_n + fi_{n-1}$, la somme des deux plus faibles fréquences.

✓ Choisir le plus petit indice *k* tel que s soit supérieur ou égal à fi_k, remplacer *k* par *k+1*.

✓ Recomposer la table des fréquences en plaçant à la *k* ème position la valeur s et en décalant les autres d'une position vers le bas. Puis décrémenter n d'une unité, poursuivre jusqu'à ce que la table des fréquences ne comporte plus que deux éléments.

✓ Coder avec retour arrière depuis le dernier groupe, en ajoutant un 0 ou un 1 pour différencier les symboles préalablement regroupés.

Exemple de code de Huffman :

Soit la source ayant un alphabet de 8 symboles avec les probabilités ci-dessous :

Symbole	0	1	2	3	4	5	6	7
Probabilité	0.01	0.02	0.05	0.09	0.18	0.20	0.20	0.25

L'arbre de Huffman est alors (cf.figure 1.2) :

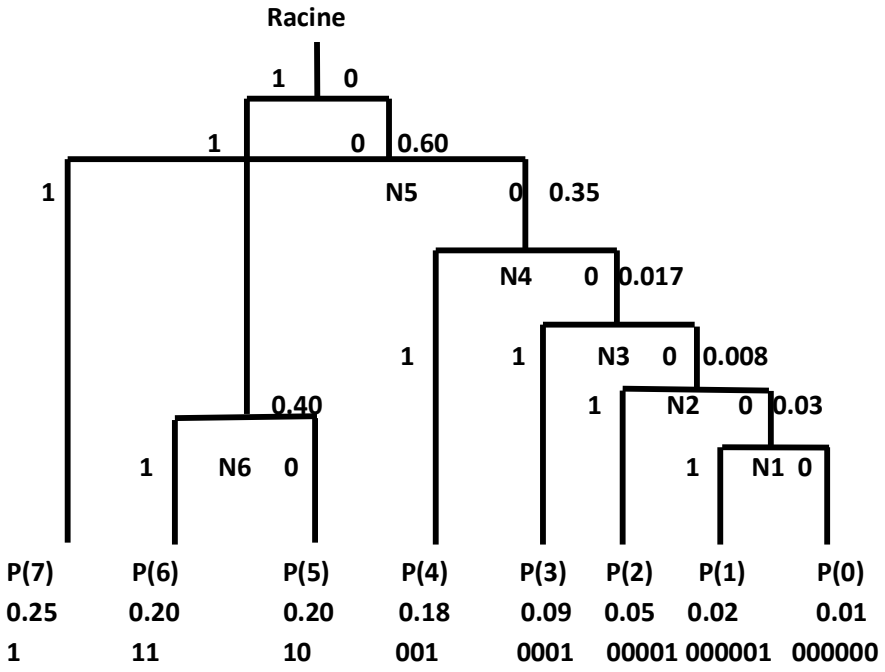


Figure1.2 : arbre de Huffman après construction

4.1.3 Codage arithmétique

Contrairement aux algorithmes de Huffman et de Shannon-Fano qui associent à des symboles des motifs binaires dont la taille dépend de leur distribution. Le codeur arithmétique traite le fichier dans son ensemble, en lui associant un unique nombre décimal rationnel.

Ce nombre compris entre 0 et 1, possède d'autant moins de chiffres après la virgule que le fichier dont il est redondant. Ces chiffres décimaux dépendent non seulement des symboles du fichier dans l'ordre où ils apparaissent, mais aussi de leur distribution statistique.

a.Algorithme du codage arithmétique

a.1. Codage

Pour générer le nombre de sortie, on suit les étapes suivantes :

1. Déterminer la probabilité de chaque symbole : par exemple, le message 'ondelettes' a la répartition de probabilités suivante :

Caractère	**Probabilité**
O	1/10
N	1/10
D	1/10
E	3/10
L	1/10
T	2/10
S	1/10

2. Affecter à chaque symbole un intervalle basé sur la probabilité, les bornes extrêmes de l'intervalle étant 0 et 1. On donne à chaque caractère une portion de l'intervalle 0 à 1 proportionnelle à sa probabilité d'occurrence :

Caractère	**Probabilité**	**intervalle**
O	1/10	0.00→0.10
N	1/10	0.10→0.20
D	1/10	0.20→0.30
E	3/10	0.30→0.60
L	1/10	0.60→0.70
T	2/10	0.70→0.90
S	1/10	0.90→1.00

3. On commence par le premier caractère 'O', l'intervalle où se trouve notre code sera compris entre 0.00 et 0.10 ; le prochain caractère à coder, la lettre 'N', occupe l'intervalle allant de 0.10 à 0.20, le nouveau nombre codé sera quelque part entre les 10% et 20% de l'intervalle actuellement établi (0.00–0.10). En appliquant cette logique, le nombre sera entre 0.01 et 0.02.

La table suivante est le résultat de ce processus complet appliqué a notre message :

Nouveau caractère	**Intervalle**	
O	0.00	0.10
N	0.01	0.02
D	0.012	0.013
E	0.0123	0.0126
L	0.01248	0.01251
E	0.012489	0.012498
T	0.0124953	0.0124971
T	0.01249656	0.01249692
E	0.012496668	0.012496776
S	0.0124967652	0.012496776

La limite inférieure de l'intervalle final, codera de façon unique le message 'ONDELLETES' en utilisant le modèle de codage actuel.

a.2. Décodage

Dans un message compressé par le codage arithmétique [5,47], c'est le premier symbole qui est le plus important : pour décoder correctement le premier caractère, le code final doit être un nombre entre 0.00 et 0.10.

1. On trouve le premier symbole du message en recherchant le symbole qui occupe l'intervalle dans lequel se trouve le message codé :
   Puisque [0.012496776] est compris entre 0.00 et 0.10, le premier caractère doit être 'O'.
2. Puisque nous connaissons la limite inférieure et supérieure du caractère 'O', on supprime son effet de la limite inférieure en inversant le processus qui les a générés :
   a. On soustrait la valeur basse de 'O', ce qui donne : 0.0124967652
   b. Ensuite, on divise le résultat par la longueur de l'intervalle de 'O', c'est-à-dire 0.1, on obtient ainsi : 0.124967652
3. Puis en recherche l'intervalle contenant cette valeur, celui de la prochaine lettre 'N'. En suivant les mêmes étapes, l'algorithme de décodage de notre message procédera ainsi :

Nombre codé	Symbole de sortie	Intervalle	longueur de l'intervalle
0.0124967652	O	0.00→0.10	0.1
0.124967652	N	0.10→0.20	0.1
0.24967652	D	0.20→0.30	0.1
0.4967652	E	0.30→0.60	0.3
0.655884	L	0.60→0.70	0.1
0.55884	E	0.30→0.60	0.3
0.8628	T	0.70→0.90	0.2
0.814	T	0.70→0.90	0.2
0.57	E	0.30→0.60	0.3
0.9	S	0.90→1.00	0.1

4.1.4 Méthode des plages

Lorsqu'on considère une ligne de la matrice représentant une image numérique, plusieurs échantillons successifs sur cette ligne peuvent posséder la même valeur. L'ensemble de ces échantillons est appelé " plages ". Cette méthode consiste donc à décrire les suites des pixels identiques par leurs longueurs et leurs valeurs. Par exemple, une plage de vingt pixels noirs équivaut à la donnée de 2 nombres : 20 et 0.

4.2 Méthodes avec distorsion des données

Ces méthodes permettent de retrouver une approximation de l'image numérique. Les pertes sont généralement indécelables à l'œil nu.

4.2.1 Quantification vectorielle (QV)

Les techniques de compression d'images exploitent généralement la redondance statistique présente dans l'image. La quantification scalaire qui associe à une variable Continue une variable discrète pouvant prendre un nombre plus faible, et fini de valeurs. Ces valeurs ne sont jamais totalement décorrélées, ou indépendantes. Shannon a montré qu'il était toujours possible d'améliorer la compression de données en codant des vecteurs plutôt que des scalaires [6].

La QV, développée par Gersho et Gray a pris une place très importante dans le domaine de la compression d'image que ce soit dans le but de transmission ou d'archivage.

- **Principe de la quantification vectorielle**

La quantification vectorielle dans son sens le plus général est l'approximation d'un signal d'amplitude continue par un signal d'amplitude discrète. Elle peut être vue comme une application *Q* associant à chaque vecteur d'entrée x de dimension *K* un vecteur $y = Q(x)$ de même dimension appartenant à un ensemble fini *Y* appelé DICTIONNAIRE de taille finie *N*,

$Y = (y_j , j = 1...N)$.

Elle se décompose en deux applications : codeur, décodeur (cf. figure 1.3) :

- **Codeur**

Le rôle du codeur consiste, pour tout vecteur xi du signal en entrée à rechercher dans le dictionnaire *Y* le code vecteur y_j le plus proche du vecteur source *x*. C'est uniquement l'adresse

du code vecteur y_j ainsi sélectionnée qui sera transmise ou stockée. C'est à ce niveau donc que s'effectue la compression.

- **Décodeur**

Il dispose d'une réplique du dictionnaire et consulte celui-ci pour fournir le code vecteur d'indice correspondant à l'adresse reçue. Le décodeur réalise l'opération de décompression.

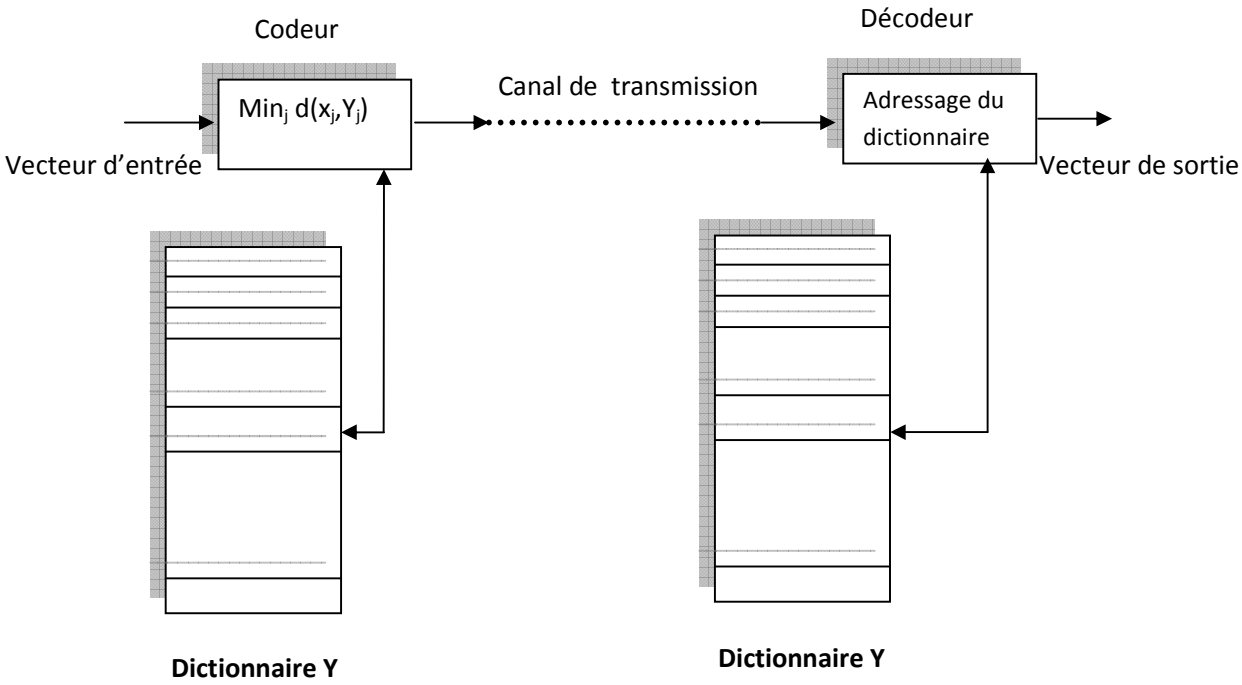


Figure1.3 : Principe du quantificateur vectoriel

Où $Min_j(x,y_j)$ = distorsion minimale entre les vecteurs x et y_j.

4.2.2 Méthodes par transformée

Dans ces méthodes, l'image de dimension *NxN* est subdivisée en sous images ou blocs de taille réduite (la quantité de calcul demandée pour effectuer la transformation sur l'image entière est très élevée). Chaque bloc subit une transformation mathématique orthogonale inversible linéaire du domaine spatial vers le domaine fréquentiel, indépendamment des autres blocs (transformée en un ensemble de coefficients plus ou moins indépendants). Les coefficients obtenus sont alors quantifiés et codés en vue de leur transmission ou de leur stockage. Pour retrouver l'intensité des pixels initiaux, on applique sur ces coefficients la transformation inverse. Parmi les transformations linéaires existantes [7]:

- ✓ Transformation de Karhunen-Loeve (TKL).
- ✓ Transformation de Fourrier Discrète (TFD).
- ✓ Transformation de Hadamard (TH).
- ✓ Transformation en Cosinus Discrète (TCD)[8].
- ✓ Transformation en ondelettes (TO).

Le principe d'un système de codage par transformation est le suivant (cf figure 1.4) :

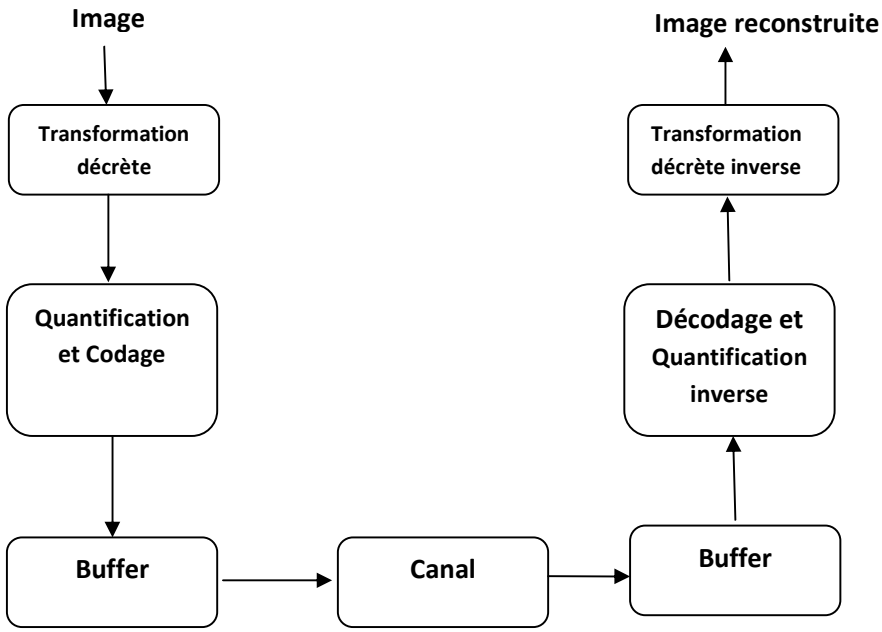


Figure1.4 : Principe d'un système de codage par transformation

4.2.3 Méthodes prédictives

La méthode prédictive est l'une des plus anciennes, c'est une méthode décorrélative dont le principe est le suivant. Dans le codage par prédiction la valeur de chaque pixel est prédite à partir des pixels précédemment codés. Seul l'écart entre la valeur prédite et la valeur réelle est quantifié puis codé et transmis. L'écart étant en général faible, sa représentation nécessite moins de bits que le pixel lui même. Les méthodes prédictives permettent une mise en œuvre facile et conduisent à de bons taux de compression. Elles sont efficaces pour les images dont les variations temporelles ou spatiales sont petites.

4.2.4 Méthodes hybrides

Le terme hybride fait référence aux techniques qui combinent le codage prédictif et le codage par transformée.

Dans le cas des images fixes, on effectue une transformation à une dimension le long des lignes et ensuite une prédiction le long des colonnes. Pour les images animées, on effectue une combinaison entre une transformation bidimensionnelle dans le domaine spatial et une prédiction le long de la composante temporelle pour exploiter la redondance temporelle du signal d'image. Le codeur hybride regroupe les avantages des deux techniques qui le composent.

5. Les normes de compression des images avec et sans pertes

5.1 La norme de compression JPEG

La norme JPEG [9] (Joint Photographic Experts Group) est conçue par le groupe ISO (International Standards Organisation) et le groupe CEI (Commission Electronic International). Elle est destinée à la compression des images fixes en couleurs et à niveaux de gris en vue de leur stockage sur les supports numériques. Elle a été réalisée dans la perspective de couvrir les applications les plus diversifiées en tenant compte des contraintes réalistes par rapport aux applications les plus visibles : publication, transmission, banques d'images.

Les techniques définies par la norme JPEG se divisent en deux classes :

- les méthodes de compression avec pertes qui sont basées sur la TCD suivie d'une quantification et d'un codeur entropique (cf. figure 1.5).

- La seconde classe, concerne les processus de codage sans pertes, cette classe de codeurs n'est pas basée sur la TCD mais sur le codage MICD suivi d'un codage entropique (cf. figure 1.6). Pour les méthodes avec pertes, quatre codeurs ont été spécifiés : un codage de base où l'image compressée puis décompressée n'est plus identique à l'image originale, ce processus utilise la TCD et un codage de Huffman.

Les trois autres types de codage sont une extension de codage de base. Ils diffèrent de codage de base principalement par le codage entropique en utilisant un codage arithmétique ou par restitution progressive de l'image.

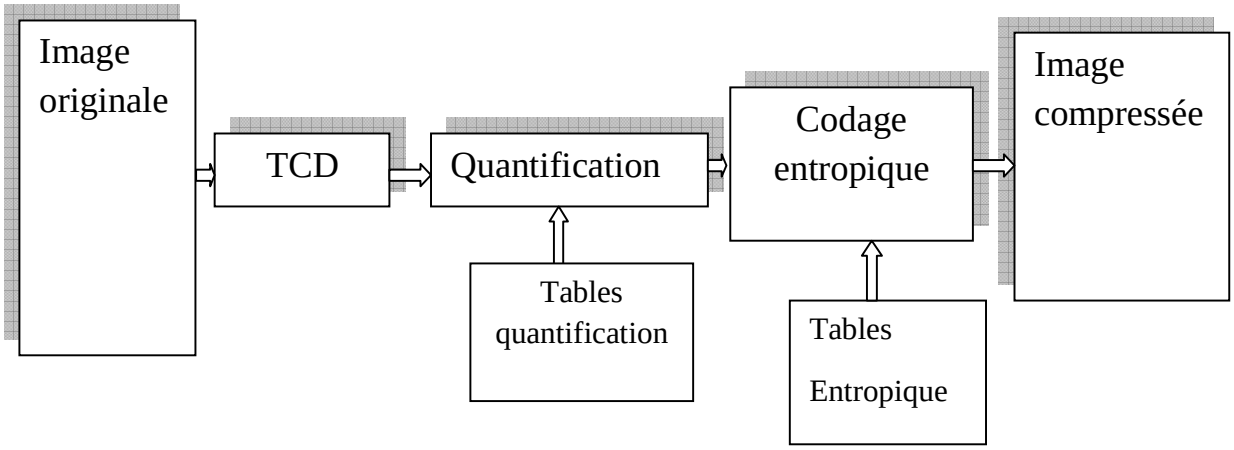


Figure 1.5 : principe de l'algorithme JPEG avec pertes.

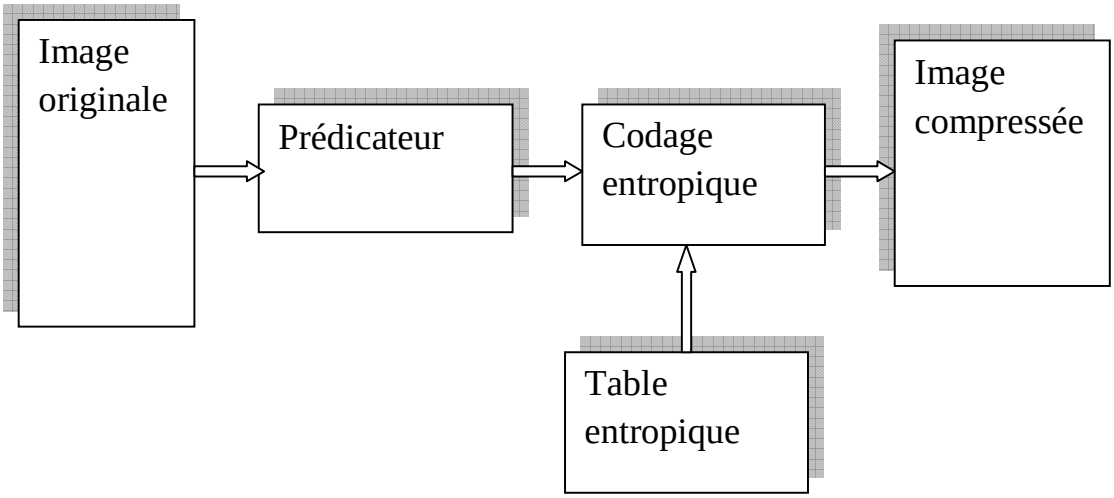


Figure 1.6: principe de l'algorithme JPEG sans pertes.

5.2 La norme de compression JBIG

La norme JBIG [10] (Joint Bi-level Image Group) est destinée à la compression d'images photographiques représentées en deux tons (noir & blanc), images textes.

Cette norme est destinée aussi, pour des débits variant de 9.6 Kbits/S à 64 Kbits/S. Elle utilise un codage sans perte avec codeur arithmétique adaptatif. Sa structure est sous forme de couches, chaque couche est un codeur indépendant.

5.3 La norme de compression H-261

C'est une norme développée par C.C.I.T.T. (Commission Consultative Internationale de la Télégraphie et de la Téléphonie). Ce standard est destiné au codage des images animées pour la visiophonie (Téléphonie Visuelle).

Le H-261 utilise un codage hybride combinant la TCD et le codage prédictif. La TCD est utilisée pour la réduction de la redondance spatiale (codage intra-trames). Le codage prédictif pour la réduction de la redondance temporelle entre les images de la séquence (codage inter-trames).

5.4 La norme de compression MPEG

Les efforts développés par les équipes du CCITT (Comité Consultatif International de Téléphonie et Télécommunication) pour le H.261 ont été utilisés comme point de départ pour le développement d'un standard de codage d'images animées par ISO, ce standard s'intitule MPEG pour Moving Pictures Experts Group.

Contrairement au H.261 en première phase, MPEG [11] est destinée au codage des images animées en vue de leur stockage sur les supports numériques d'où les contraintes plus souples que celles du H.261 (c'est pour cela qu'on l'appelle le standard des applications multimédia).

La première phase de MPEG intitulée MPEG-I spécifie une compression du signal vidéo à un débit de 1.5 Mbits/s. Les deux autres phases ont pour but d'améliorer la qualité du codage vidéo en sacrifiant une augmentation de débit, MPEG-II est destinée à la compression du signal vidéo à des débits d'ordre de 10 Mbits/s, MPEG-III est destinée à la télévision haute définition à des débits de 30 à 40 Mbits/s. Une quatrième phase, MPEG-IV est destinée au codage d'images animées à très faibles débits (10 Mbits/s), MPEG-A tourné vers les applications multimédia. En cours de standardisation.

6. Codages imbriqués

La transformation en ondelettes génère des sous-bandes qui correspondent à des projections orthogonales dans des espaces vectoriels disjoints. Il subsiste des ressemblances structurelles dans les sous-bandes détails dans la même direction à des résolutions successives (cf. figure 1.7). Le principe de l'arbre de zéro proposé par Shapiro en 1993, permet d'exploiter cette ressemblance.

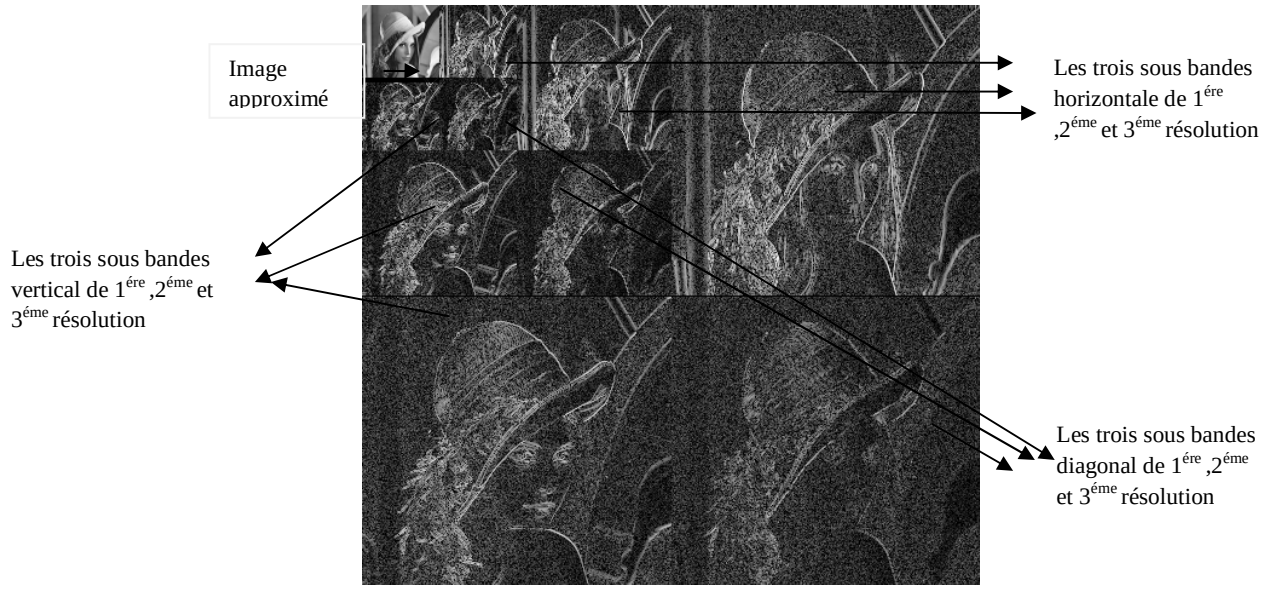


Figure 1.7: Exemple de transformation en sous-bande de l'image Lena, la ressemblance entres les sous bandes est remarquable.

6.1 Algorithme EZW de Shapiro

C'est le premier codeur en sous-bande par '*zerotree*' à avoir été introduit [12]. Il procède au regroupement des coefficients non significatifs sous forme d'arbre de zéros (*zerotree*). La structure zerotree permet de détecter les zones de l'image qui ne contiennent pas d'information significative et sont codées ensuite en arbre.

L'algorithme EZW peut être résumé en trois étapes, comme suit :

- La définition des cartes de signifiance indiquant les positions des coefficients significatifs par rapport à un seuil donné.
- Une approximation successive, par passes, des coefficients significatifs, qui permet donc une notion de progressivité du codage selon un critère d'arrêt de débit-distorsion.

- Un codeur arithmétique dynamique de la chaine de symboles.

Le codeur EZW offre la propriété de la transmission progressive de l'image codée tout en apportant d'excellentes performances débit/distorsion par rapport à la norme JPEG.

6.2 Algorithme SPECK

Offrant des performances comparables à l'algorithme SPIHT, l'algorithme SPECK [13] exploite des structures d'ensembles de coefficients non significatifs en blocs plutôt qu'en arbres. Ces structures de blocs permettent de s'affranchir efficacement du non stationnarité (d'ordre 1) des coefficients en adaptant localement la statistique utilisée pour le codage.

- **Principe**

Les coefficients sont initialement séparés en deux ensembles, l'un noté S contenant les coefficients de basses fréquences et l'autre, noté τ contenant le reste des coefficients (cf. figure 1.8). De la même manière que dans SPIHT, deux listes sont tenues à jour, l'ensemble (LSP) pour représenter les coefficients significatifs et l'ensemble de coefficients non significatifs (LIS). La liste d'ensembles non significatifs LIS contient des blocs de coefficients de taille variable y compris les coefficients isolés vus comme des blocs de 1x1 (stockés dans la liste LIP dans le cas de SPIHT).

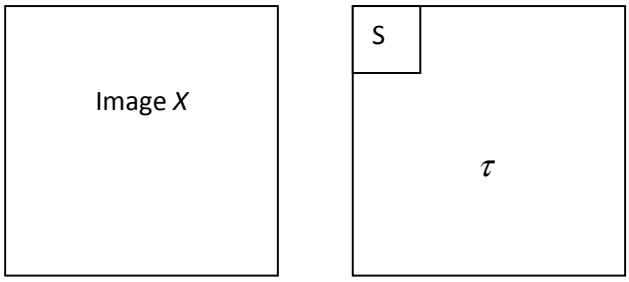


Figure 1.8 : Partitionnement de l'image X en deux sous-ensembles : S et le reste

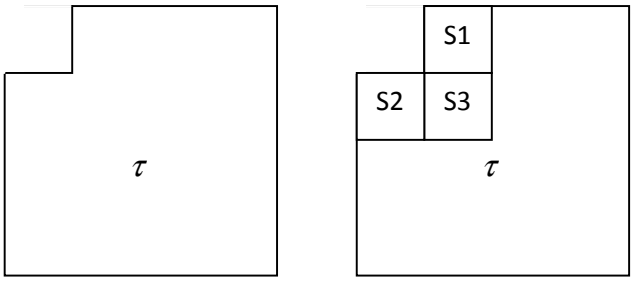


Figure 1.9: Principe de partitionnement des ensembles

L'algorithme SPECK peut être résumé comme suit [13]:

1. Initialisation :
   - Partitionner l'image X en deux sous-ensembles : $S \equiv Racine$, et $\tau = X - S$ (cf. figure 1.7).
   - Envoyer $n_{\max} = \left\lfloor \log_2 \left(\max_{(i,j) \in X} \left\{ \left| C_{i,j} \right| \right\} \right) \right\rfloor$.
   - Ajouter S à LIS et mettre LSP=ϕ.
2. Passe de test de signifiance :
   2.1. Dans l'ordre croissant de taille des listes S (les ensembles les plus petits d'abord).
   - Pour chaque ensemble S ∈ LIS faire :
   - Si l'ensemble est signifiant et non réduit à un seul coefficient alors,

   Retirer l'ensemble de la liste, le partitionner récursivement en quatre sous-blocs, sur lequel ce test est effectué à nouveau.
   - Si le bloc est réduit à un seul coefficient significatif, alors,

Ajouter le coefficient à LSP.

- Sinon, l'ensemble est laissé dans la LIS.

2.2. Test de l'ensemble τ :

- Si τ est signifiant, alors,

Le bloc τ est séparé en trois blocs correspondants au sous-bandes de plus basse-fréquences et un ensemble τ contenant le reste des coefficients (cf. figure 1.8). Les trois nouveaux blocs sont traités comme précédemment.

- Répéter le processus de séparation de l'ensemble τ jusqu'à ce qu'il soit insignifiant.

3. Passe de raffinement (même principe que SPIHT) :

Pour chaque (i,j) $\in$ LSP, à l'exception de ceux testés par la passe précédente, envoyer le nième bit de poids le plus fort de $C_{i,j}$

4. Réitération :

Décrémenter n par un et, et aller à l'étape 2.

L'algorithme SPECK, donne généralement des résultats similaires à ceux obtenus par SPIHT.

6.3 Algorithme EZBC

Le principe de codage par l'algorithme EZBC (Embedded ZeroBlocks coding based on Context modeling) [14] est similaire à l'algorithme SPECK. L'innovation de ce codeur provenant principalement de l'exploitation de la dépendance entre les nœuds du quad-tree de signifiance. Le partitionnement en quad-tree est de plus réalisé indépendamment dans chaque sous-bande permettant une meilleure séparation des statistiques de signifiance et un apprentissage plus efficace à l'aide de contextes plus étendus [15].

Ce codeur offre des performances comparables à EBCOT, et a également été adapté au codage vidéo sous le nom de MC-EZBC [16].

7. Conclusion

La compression des images est désignée à prendre un rôle encore plus important en raison de l'évolution des réseaux et du multimédia. Son importance est surtout due au décalage qui existe entre les possibilités matérielles des dispositifs que nous utilisons (capacité des mémoires de masse, débits sur Internet, …) et les besoins qu'expriment les utilisateurs (visiophonie, vidéo plein écran, transfert de quantités d'informations toujours plus importantes dans des délais toujours plus brefs. Les méthodes déjà utilisées couramment sont efficaces et sophistiquées (Huffman, MPEG, JPEG….) et utilisent des théories assez complexes. Notre travail, est consacré au codeurs hiérarchiques SPIHT et notamment à son optimisation notée MSPIHT (détaillé chapitre 3).

Chapitre 2 : Analyse multirésolution

Résumé

Nous nous limitons, dans ce deuxième chapitre à la présentation des principales transformée d'images utilisée dans l'étape de compression existantes dans la littérature. On commence par décrire les transformées directionnelles (radon, ridjelets, curvelet, brushelet……) et on insiste sur les limites dans leur utilisation dans le codage imbriqué. Puis on parle des caractéristiques principales de la transformée de Fourier. Ensuite on continue avec un bref rappel sur la Transformation de Fourier à fenêtre glissante et la Transformation de Gabor. La dernière partie de ce chapitre sera consacrée à la transformation en ondelettes continues et ondelettes discretes plus spécifiquement l'analyse multirésolution. L'idée étant de montrer son intérêt lorsqu'elle est associe avec le codage imbriqué.

1. Introduction

Depuis leur introduction il y a deux décennies, les ondelettes ont gagné un intérêt considérable en traitement du signal. L'idée de représenter un signal à différentes résolutions permet d'en extraire ses tendances principales en un nombre restreint de coefficients, tout en localisant précisément les discontinuités. Dans le contexte du traitement d'image, les ondelettes ont été utilisées pour des applications variées, telles que le débruitage et la compression, menant à des standards comme JPEG2000. Il est bien connu que les ondelettes sont optimales pour la représentation de signaux unidimensionnels (1D) possédant un nombre fini de discontinuités. En effet, l'erreur quadratique moyenne d'une approximation non-linéaire faite à partir des k coefficients d'ondelette maximaux décroît en *O(k-1)* dans le cas des images, pour des raisons de simplicité et d'efficacité, les ondelettes ont souvent été utilisées de manière séparable sur les axes horizontal et vertical. Il en résulte une décorrélation partielle de l'image, qui se traduit par la présence de nombreux coefficients de forte énergie le long des contours.

Dans le premier temps, nous parlons des transformées directionnelles et de la transformée de Fourier en soulignant ses insuffisances ; nous évoquerons les notions utiles par l'intermédiaire de définitions sur la transformée de Fourier à court terme puis nous définirons la transformée en ondelettes continue. Ensuite, nous présentons la théorie de l'analyse multirésolution (AMR) et l'algorithme pyramidal permettant d'obtenir les coefficients de la transformée en ondelettes discrète par filtrage et décimation successifs.

2. Transformées directionnelles

2.1 Transformée de Radon

La transformée de Radon [15,17] consiste à projeter l'image sur un certain nombre d'orientations en intégrant l'image le long de la direction orthogonale à la projection (Figure 2.1), puis à réaliser la transformée de Fourier de ces projections. La reconstruction s'obtient en plaçant, pour chaque orientation de projection choisie, les coefficients de Fourier obtenus le long de cette même orientation, dans le domaine fréquentiel on obtient l'image reconstruite en effectuant ensuite une transformée de Fourier 2D inverse. La reconstruction parfaite pour cette transformée continue s'obtient pour un nombre de projections infini, parcourant l'ensemble des orientations possibles. La transformée de Hough est un cas particulier de transformée de Radon pour une image à valeurs binaire, et s'utilise principalement pour la reconnaissance de formes.

La transformée de Radon est très utilisée en tomographie, où les données capturées correspondent précisément à des projections du contenu de l'objet dont on cherche à obtenir une image. Une reconstruction approximative de l'image recherchée, d'autant plus précise que le nombre de directions de projection est élevé, est obtenue par transformée de Radon inverse. Notons que pour une image discrète carrée dont la taille est un nombre premier, la discrétisation proposée peut s'appliquer pour obtenir une transformée de Radon discrète à reconstruction parfaite peu redondante. Une approximation discrète rapide (en $o(n^2 \log(n))$ opérations) et inversible de la transformée de Radon.

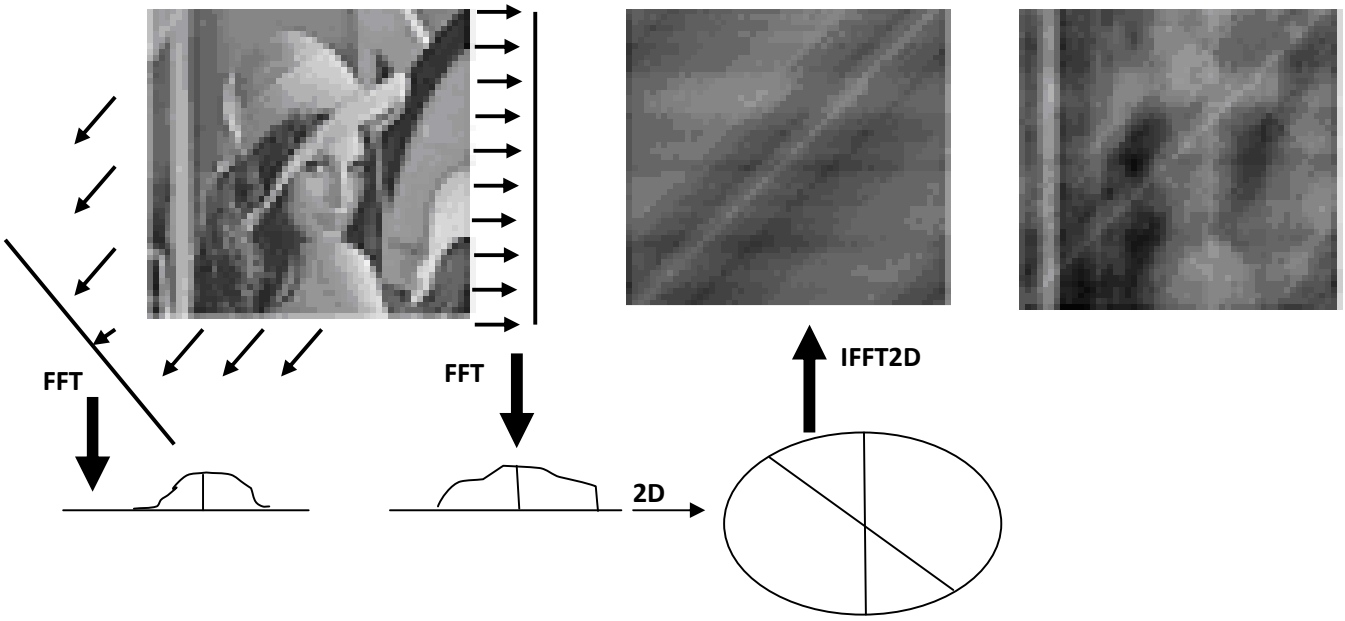


Figure 2.1 : Transformée de Radon

2.2 Transformée de Ridgelet

Les ridgelets [18] forment une extension naturelle de la transformée de Radon pour un nombre limité de directions, en se basant sur des fonctions d'ondelettes pour contrôler la précision en orientation et garantir la reconstruction parfaite. Étant donné une fonction d'ondelette $\psi(t)$, Candes [18] propose de construire une base de fonctions :

$$R_{l,n,\theta(t)} = M^{-\frac{1}{2}} \psi\left(\frac{\left(\cos(\theta, \sin(\theta))^T\right) t}{M^l} - n\right)$$

Où $l \in R_*^+$ représente l'échelle $\theta \in [0,2\pi]$, l'orientation et $n \in R$ la position de la ridgelet.

Cette famille est donc constituée de lignes d'orientation variable et dont la largeur dépend du support de l'ondelette choisie. Le facteur de redondance n'est pas donné de manière quantitative dans le cas général, et dépend du choix du facteur d'échelle M, et de la résolution angulaire et spatiale.

Une famille de ridgelets fenêtrées est proposée sous le nom d'ortho-ridgelets. Du fait du compromis sur l'incertitude temps-fréquence, ce fenêtrage implique une discrétisation en orientation pour une seule échelle donnée. Cependant, en partant d'une fenêtre de la taille de l'image et en considérant toutes les familles d'ortho-rigdelets définies sur le partitionnement dyadique de cette fenêtre initiale en sous-fenêtres, il est possible d'obtenir une décomposition en ridgelets multiéchelle. La famille totale obtenue est alors consistuée d'éléments dont le support est inscrit dans un rectangle de longueur l inférieure ou égale à la taille de l'image, et de largeur 0 < w < l. Bien que pour chaque échelle, la famille d'ortho-ridgelets correspondante forme une frame, l'ensemble des fonctions ridgelets multiéchelle n'en est malheureusement pas une. D'une certaine manière, la famille de fonctions proposée par cette transformée est trop riche, dans le sens où la somme des énergies des coefficients n'est pas bornée pour tout signal d'énergie finie. Notons qu'il existe également une transformée en ridgelets finie, en se basant sur la discrétisation des transformées de Radon finies. Il est remarquable de préciser que cette transformée est non-redondante, nécessitant toutefois d'avoir une image carrée dont la taille est un nombre premier.

2.3 Transformée de Curvelet

La transformée en curvelet [15,19] se dérive des ridgelets multiéchelles. Une première décomposition permet d'obtenir une analyse multiéchelles en sous-bandes centrées sur les couronnes de fréquences $f \in [0,1/2^{2s}]/[0,1/2^{2s+2}] ou\ s \in n$ où $s \in N$ représente l'échelle. Notons que cette découpe de l'espace fréquentiel n'est pas classique. Ces sous-bandes sont en suites analysées par une transformée en orthoridgelets sur des blocs de taille $2^s * 2^s$. Les blocs d'analyse sont alors des éléments rectilignes de taille $2^s * 2^{2s}$. Ces éléments suivent donc une loi de changement d'échelle parabolique $l = w^2$. L'ondelette utilisée pour construire les orthoridgelets est l'ondelette de Meyer tandis que la fonction d'échelle de Lemarié est utilisée pour la représentation des basses fréquences. Une construction différente et plus générale, reposant sur la théorie des

frames. La frame d'analyse est construite directement à partir d'une fonction mère C bidimensionnelle de haute fréquence selon l'un des axes et de basse fréquence selon l'autre (typiquement le produit tensoriel d'une fonction d'ondelette et d'une fonction d'échelle). La famille de curvelets $(Cl, n, \theta)_{l \in n, n \in Z,}\ \theta \in 2\pi / 2^l Z / 2^l Z$ correspondante est alors donnée par :

$$C_{l,n,\theta(t)} = 2^{3l/2} C(D_l R_\theta t - n)$$

$D_l = \begin{pmatrix} 2^{2l} & 0 \\ 0 & 2^l \end{pmatrix}$ Est la matrice de sous-échantillonage du changement d'échelle

Notons au passage que $2^{3l/2}$ correspond à la racine carrée de son déterminant), $R\theta$ est la matrice de rotation d'angle θ et n indique la position de la curvelet. Cette transformée a été utilisée avec succès dans le cadre du débruitage .

2.4 Transformée de Contourlet

Inspirés par les curvelets, Do et Vetterli [21] ont développé la transformée en contourlets qui obéit à un nouveau schéma de décomposition multirésolution qui permet de représenter l'image comme un ensemble compacte de niveaux de résolution spatiale et directionnelle (orientées). Contrairement au curvelets, la transformée en contourlets est conçue directement dans le domaine discret. L'esprit de la méthode est cependant similaire à celui des curvelets car il s'agit d'une base fixe avec une grande sélectivité directionnelle. Le but principal de la construction contourlets est de fournir une représentation éparse des données aussi bien aux résolutions spatiales que fréquentielles. L'appellation «contourlet» vient du fait que cette nouvelle décomposition est un frame composé de segments de contour.

Les ondelettes 2D, résultant du produit tensoriel de deux transformées en ondelettes 1D, fournissent un nombre limité et fixe d'orientations et peuvent capturer uniquement les discontinuités ponctuelles sans prendre en compte la régularité des contours. Les contourlets ont été développées comme remède à cette inefficacité des ondelettes. En effet, la transformée en contourlets possède non seulement les principales propriétés des ondelettes (à savoir, sa multirésolution, sa bonne localisation en espace et en fréquence, ainsi que sa décimation critique) mais offre aussi une grande sélectivité directionnelle. Pour mettre en évidence la différence entre la transformée en ondelettes et la transformée en contourlets, la Figure 2.2 montre quelques images de base de ces deux transformées.

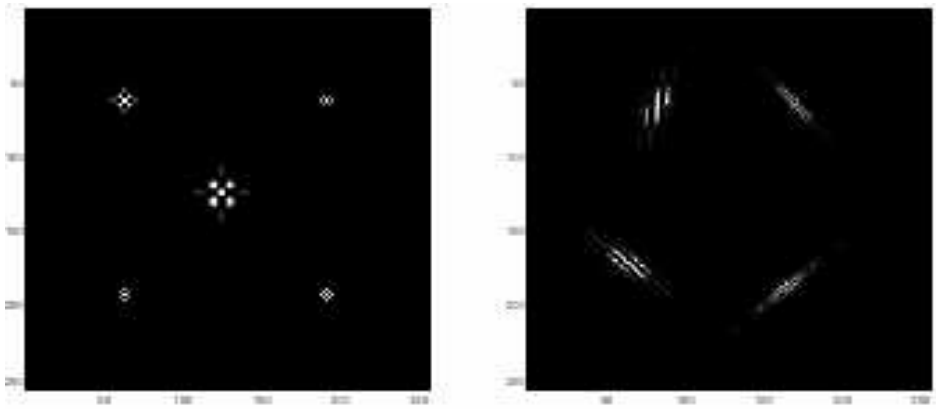


Figure 2.2 : Représentations d'image par Ondelettes et Contourlets. (a) Exemples de cinq fonctions de base pour la transformée en ondelettes 2-D. (b) Exemples de quatre fonctions de base pour la transformée en contourlets.

2.5 Transformée de Brushlet

La transformée en brushlets [22] a été proposée par Meyer et Coifman . Elle donne un partitionnement adaptatif du plan fréquentiel offrant plus de flexibilité que les paquets d'ondelettes en s'affranchissant de la contrainte de séparabilité. Il est alors possible de représenter un motif orienté à l'aide d'un seul coefficient. La décomposition est effectuée en appliquant successivement des opérateurs de fenêtrage à l'image suivis chacun d'une transformée de Fourier. La reconstruction s'obtient alors en effectuant la transformée inverse de chaque composante puis en appliquant l'opérateur de fenêtrage dual et en sommant finalement chaque contribution. Des constructions de base orthogonale et bi -orthogonales sont proposées dans le cas 1D et 2D, ainsi qu'une procédure de discrétisation. Une optimisation par quad-tree est mise en oeuvre pour sélectionner le partitionnement fréquentiel adéquat dans le cadre de la compression d'image.

2.6 Transformée de Beamlet

La décomposition en beamlets [23] considère un partitionnement de l'image en quad-tree, puis effectue une transformée de Radon dans chaque bloc. Les coefficients de beamlets sont liés par une relation multiéchelle, où chaque beamlet à un niveau donné est décomposée en trois beamlets connexes au niveau suivant. Cette transformée permet d'approximer les courbes dans les images et d'en extraire les contours par sélection dans le graphe de connexité des beamlets.

2.7 Transformée de Wedgelet

Donoho [24] a proposé une décomposition adaptative qui soit adaptée aux contours lisses dans les images, les wedgelets. Il s'agit d'une représentation optimale des images appartenant au modèle Horizon [24] des fonctions $\forall(x1,x2)\in[0,1]: f(x)=1 \quad x2 \succ H(x1)$ H étant à valeurs sur [0,1]. Cette transformée procède à une décomposition dyadique récursive de l'image en carrés dyadiques, à l'aide d'un quadtree, et représente l'image dans chacun de ces carrés par une fonction constante par morceaux avec une discontinuité linéaire. Plus simplement, chaque carré contient deux zones de valeur constante, séparées par une ligne droite. Romberg et al. [44] ont ensuite proposé une implémentation rapide en calculant les échelles larges à partir des fines, et en ajoutant une condition d'admissibilité sur les angles des wedgelets, c'est à dire qu'il y a une certaine cohérence, similarité entre les décompositions à différentes échelles, et plus tard Friedrich et al. [45] ont proposé une autre implémentation rapide.

2.8 Transformée de Bandelette

La transformée en bandelettes (ondelette seconde génération) [25,48] est une transformée adaptative de l'image. Elle s'effectue en trois parties principales: tout d'abord l'image subit une transformée en ondelettes orthogonales séparables. Puis chaque sous-bande est segmentée en blocs et sous blocs suivant un quadtree, dans les feuilles duquel on recherche un flot régulier. Une fois la segmentation faite et le flot calculé, les coefficients sont réordonnés dans un vecteur suivant le flot et subissent une transformée en ondelettes unidimensionnelle. Concrètement, la seconde partie peut être de rechercher la direction principale de l'imagette, et de lire les coefficients en considérant cette direction comme l'horizontale.

A noter que la transformée originale effectuait un processus similaire, mais décomposait directement l'image et non sa transformée en ondelettes. La figure 2.3 illustre les étapes de la décomposition.

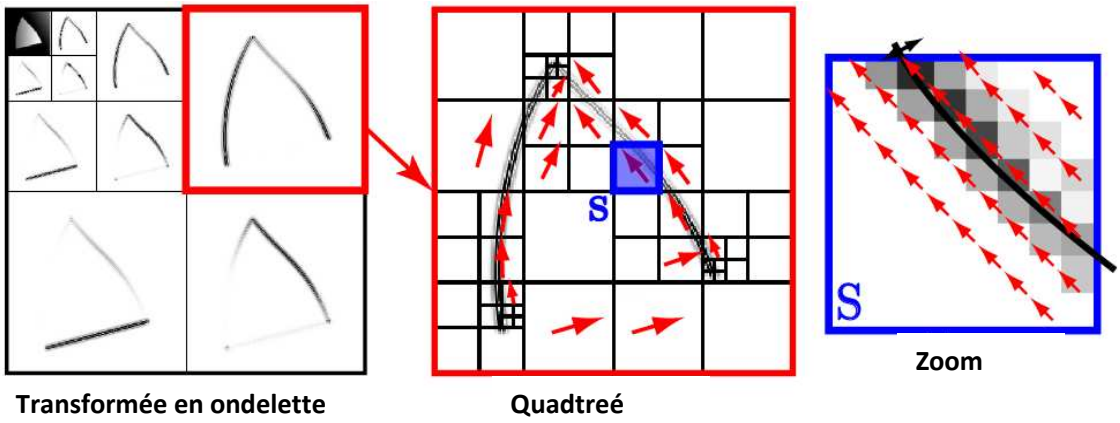


Figure : 2.3 Décomposition en bandelettes obtenue par déformation d'une ondelette séparable le Long du champ d'orientations.

3. Transformée de Fourier

La transformée de Fourier d'un signal *x(t)* est donner par la relation 2.1

$$X(f) = \int_{-\infty}^{+\infty} x(t) e^{-j2\pi ft} dt = \langle x(t), ej2\pi ft \rangle \qquad (2.1)$$

La première solution qui vient naturellement à l'esprit est de limiter le domaine d'intégration temporel (ou spatial) à l'aide d'une fonction «fenêtre» que l'on pourra faire glisser pour explorer le signal ; on obtient ainsi la transformée de Fourier à fenêtre glissante ; voir l'équation 2.2.

3.1 Transformation de Fourier à fenêtre glissante

$$STFTx(f, t) = \int_{-\infty}^{+\infty} g^*(t-\tau) x(t) e^{-j2\pi ft} dt \qquad (2.2)$$

Si on pose $G_{f,\tau}(t) = g(t-\tau)$ (2.3)

On peut interpréter cette transformée comme la projection de f sur la «base» des fonctions fenêtres glissantes : $STFTx(f, t) = \langle G_{f,\tau}(t), x(t) \rangle$ (2.4)

La notation $\langle g, x \rangle$ représente le produit scalaire :

$$\langle g, x \rangle = \int_{-\infty}^{+\infty} g(t) * x(t) dt \qquad (2.5)$$

Un certain nombre de fonctions fenêtres sont utilisées, les plus connues sont les fenêtres de Hanning, de Hamming, et de Gauss:

$$g(t) = \pi^{-1/4} e^{-x^2/2} \qquad (2.6)$$

Dans ce dernier cas la transformation a été baptisée transformation de Gabor (équation 2.8) et on appelle « gaborette » la fonction analysante. On notera que les fonctions enveloppes sont normalisées à 1 ; la norme étant défi nie par :

$$\|X\|^2 = \int_{-\infty}^{+\infty} x(t)x^*(t)\,dt \qquad (2.7)$$

3.2 Transformation de Gabor

La transformée deGabor d'un signal *x(t)* est donner par la relation 2.8

$$Tgabor\ X(f,t) = \pi^{-1/4} \int_{-\infty}^{+\infty} x(t) e^{-(t-\tau)/2} e^{-jwt}\,dt \qquad (2.8)$$

On trouvera sur la Figure 2.4 La représentation de la partie réelle de «gaborettes» pour deux fréquences différentes. On peut vérifier que l'étendue temporelle de la fonction est indépendante de la fréquence analysée par cette fonction. La résolution dans le plan temps-fréquence de la transformation peut être estimée par les variances de la fonction analysante dans l'espace temporel et dans l'espace fréquentiel :

$$\delta_t^2 = \int_{-\infty}^{+\infty} x^2 |\psi(t)|^2\,dt \qquad (2.9)$$

Si on considère une gaborette $|\psi(t)| = e^{-t^2/2}$ on a évidemment $\delta_t = 1$ et dans ce cas comme la transformée d'une gaussienne est une gaussienne ($e^{-\pi t^2}$ se transforme par Fourier en $e^{-\pi f^2}$) on trouve un écart type δ_f dans le domaine fréquentiel tel que : $\delta_f = \frac{1}{2\pi}$. On constate que la résolution temporelle δ_t et fréquentielle δ_f sont indépendantes, de sorte que le pavage de l'espace temps-fréquence (figure 2.5) est régulier.

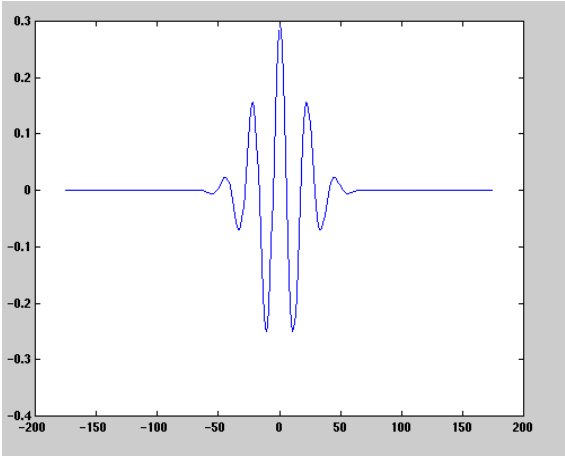


Figure 2.4: Gaborettes

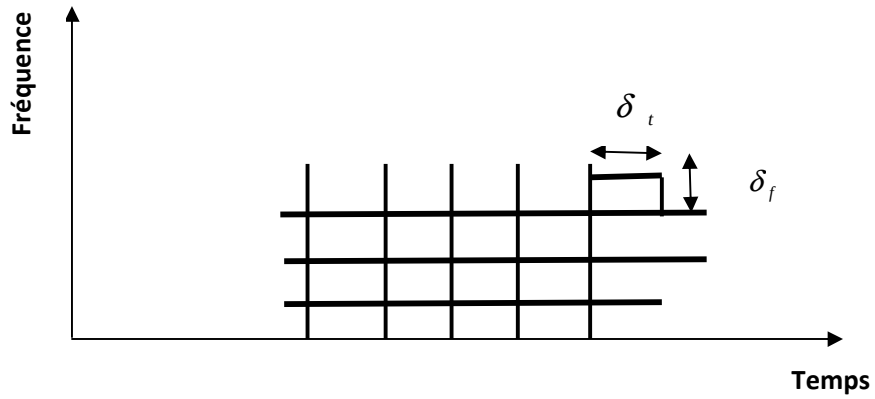


Figure 2.5 : Pavage temps-fréquence pour la transformée à fenêtre glissante

Dans ce cas, on comprend que l'analyse n'est pas idéale car si une résolution temporelle faible est automatiquement liée à la détection des basses fréquences, la détection des composantes hautes fréquences du signal peut être faite avec une résolution temporelle supérieure. Les deux résolutions doivent varier en sens inverse en conservant un produit constant pour un pavage énergétiquement régulier de l'espace temps-fréquence. Ceci doit conduire à une utilisation rationnelle de cet espace par la réalisation dans tous les cas du meilleur compromis possible entre la résolution temporelle et la résolution fréquentielle. Cette transformation est réalisé par la transformation en ondelettes dont le principe est précisé dans l'équation (2.10).

4. Transformée en ondelettes

$$T^{ond} f(a,b) = \frac{1}{\sqrt{a}} \int_{-\infty}^{+\infty} f(t) \psi(\frac{t-b}{a}) dt \qquad (2.10)$$

Dans cette expression 2.10, *a* est le facteur d'échelle et *b* le paramètre de translation. La variable a joue le rôle de l'inverse de la fréquence : plus *a* est petit moins l'ondelette (la fonction analysante) est étendue temporellement, donc plus la fréquence centrale de son spectre est élevée. On peut également interpréter cette expression comme une projection du signal sur une famille de fonctions analysantes $\psi_{a,b}$ construite à partir d'une fonction "mère" ψ conformément à l'équation suivante [26,27,28] :

$$\psi_{a,b}(t) = \frac{1}{\sqrt{a}} \psi(\frac{t-b}{a}) \qquad (2.11)$$

On notera que la norme est conservée lors du changement de facteur d'échelle :

$$\left\| \psi_{a,b} \right\|^2 = \int_{-\infty}^{+\infty} \frac{1}{a} \left| \psi(\frac{t-b}{a}) \right|^2 dx = \frac{1}{a} \int_{-\infty}^{+\infty} \left| \psi(x) \right|^2 a dx = \left\| \psi \right\|^2 \qquad (2.12)$$

On pourra noter :

$$T^{ond} f(a,b) = \left\langle f, \psi_{a,b} \right\rangle \qquad (2.13)$$

La résolution spatio-temporelle est calculée de la même manière que précédemment : Si la «largeur» temporelle de ψ (l'écart type) est prise comme unité :

$\sigma = 1$ Alors on peut calculer la «largeur» de $\psi_{a,0}$ avec l'équation 2.14 :

$$\begin{aligned} \sigma_t^2 &= \int t^2 \left| \psi_{a,0}(t) \right|^2 dt \\ &= \int t^2 \frac{1}{a} \left| \psi(\frac{t}{a}) \right|^2 dt \qquad (2.14) \\ &= \int_{-\infty}^{+\infty} a^2 x^2 \frac{1}{a} \left| \psi(x) \right|^2 a dx \end{aligned}$$

Ce lui qui donne : $\sigma_t = a$

On peut de même calculer l'occupation fréquentielle de l'ondelette en calculant l'écart type pour la transformée de Fourier $\psi_{a,0}{}^{\wedge 1}$ de $\psi_{a,0}$ en prenant comme unité l'écart type de la transformée de Fourier de l'ondelette mère ψ :

$$\begin{aligned}\sigma_w^2 &= \int w^2 \left|\psi_{a,0}{}^{\wedge}(w)\right|^2 dw \\ &= \int w^2 \frac{1}{a}\left|a\,\psi(aw)\right|^2 dw \qquad (2.15) \\ &= \int \frac{\xi^2}{a^2}\frac{1}{a}\left|a\,\psi(\xi)\right|\frac{\xi}{a}\end{aligned}$$

On trouve $\sigma_w = \frac{1}{a}$. De sorte que le pavé élémentaire dans l'espace temps-fréquence est de surface constante tandis que la résolution temporelle est proportionnelle à *a* et que la résolution fréquentielle est inversement proportionnelle à *a* comme on le voit sur la figure 2.6.

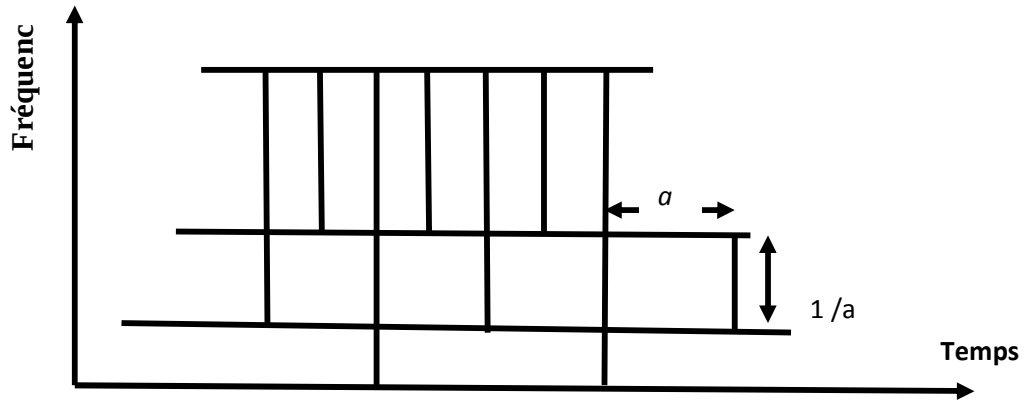


Figure 2.6 : Pavage temps-fréquence pour la transformée en ondelette discrète

5. Transformée en ondelettes discrète

$$T^{ond} f(m,n) = a_0^{\frac{-m}{2}} \int_{-\infty}^{+\infty} f(t)\psi(a_0^{\frac{-m}{2}} t - nb_0)dt \qquad (2.16)$$

Si on choisit $a_0 = 2$ et $b_0 = 1$ on parle alors de transformée dyadique.

5.1 Inversion - Admissibilité

On peut montrer [29] que si la fonction analysante (l'ondelette) est convenablement choisie, la transformation en ondelettes est inversible et la fonction peut être reconstruite après analyse suivant l'équation (2.17) :

$$f = C_{\psi}^{-1} \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \frac{1}{a^2} \left\langle f, \psi_{a,b} \right\rangle \psi_{a,b} \, da db \qquad (2.17)$$

Cette possibilité reste théorique car le calcul n'est possible que numériquement et sa convergence peut-être très lente. Le coefficient C_{ψ}, s'il existe, est donné par l'équation (2.18) :

$$C_{\psi} = 2\pi \int_{-\infty}^{+\infty} \left| \hat{\psi}(w) \right|^2 \frac{dw}{w} \qquad (2.18)$$

La condition d'existence de ce coefficient est également la condition d'admissibilité de la fonction ondelette analysante. Cette condition est explicitée par l'équation 2.19 :

$$\int_{0}^{+\infty} \left| \hat{\psi}(w) \right|^2 \frac{dw}{|w|} = \int_{-\infty}^{0} \left| \hat{\psi}(w) \right|^2 \frac{dw}{|w|} < \infty \qquad (2.19)$$

Cette relation se ramène le plus souvent à la condition exprimée par l'équation (2.20) qui n'est pas très contraignante et indique seulement que la fonction ondelette doit être à moyenne nulle. Le choix de l'ondelette est donc en principe très ouvert, il faut cependant noter que la robustesse et la vitesse de convergence de l'algorithme de reconstruction donné par l'équation (2.20) sont très dépendantes du choix de l'ondelette. Il est clair, en n, que la transformée en ondelettes ne sera intéressante comme outil d'analyse du signal que si la fonction analysante (l'ondelette) reste bien localisée dans le temps et en fréquence.

$$\int_{-\infty}^{+\infty} \psi(t) dt = 0 \qquad (2.20)$$

On peut classer les transformées en ondelettes selon la famille à laquelle appartiennent les fonctions analysantes choisies. Les transformées obtenues sont suivant les cas discrètes ou continues, redondantes ou non. Les transformées continues sont obtenues en prenant le facteur d'échelle *a* et le pas de translation *b* dans l'ensemble des nombres réels. Ces transformées sont évidemment très redondantes car l'espace temps-fréquence est parcouru continûment, ce type de transformation ne peut, dans la pratique, être effectué que de façon approximative et il y a

toujours en fait une discrétisation du calcul qui est opérée. L'approche discrète du problème a le mérite de traiter le problème de l'échantillonnage de l'espace temps-fréquence avec rigueur et de fournir une mesure de l'éventuelle redondance de la transformation obtenue. De plus, dans ce cas, les algorithmes de calcul conduisent souvent à des résultats exacts sur des intervalles donnés de l'espace temps-fréquence. Pour résumer, on peut donner le classement sommaire suivant :

• Transformées redondantes :

- ✓ Transformée continue,
- ✓ Trame d'ondelettes (frames),
- ✓ Paquet d'ondelettes.

• Transformées non redondantes :

- ✓ Analyse multirésolution : base orthonormée,
- ✓ Analyse multirésolution : base biorthogonale,
- ✓ Paquet d'ondelettes.

Cela étant, il reste à examiner les généralisations de la transformée en ondelette unimodale de signaux à une dimension. La première extension envisageable est le passage du traitement des signaux mono- dimensionnels au traitement des signaux bi-dimensionnels, tri-dimensionnels, voire au-delà. L'intérêt de cette extension est évident pour qui se préoccupe de traitement des images 2D et 3D. Dans ce domaine, nous présenterons les éléments de base sur lesquels s'appuient les principales méthodes de généralisation et nous illustrerons par deux techniques choisies parmi les plus utilisées dans les applications actuelles. Le deuxième type de généralisation qu'il conviendrait d'examiner est le problème du traitement des signaux vectoriels ou multispectraux. Le cas le plus commun est celui des images couleur. Les multi-ondelettes pourraient constituer une piste intéressante. Mais là, nous sommes vraiment dans le domaine de la recherche et il ne semble pas que les résultats soient suffisamment probants pour qu'ils puissent être présentés ici. Alors, avis aux amateurs...Bases orthonormées, analyse multirésolution [30, 31,32].

5.2 Espaces d'approximation

Nous nous plaçons dans l'espace $L^2(R)$ des fonctions continues d'une variable réelle et de carré intégrable. Une analyse à la résolution j de la fonction f sera obtenue par action d'un opérateur linéaire A_j sur *f*, tel que : $A_j f \in V_j$

V_j étant un sous espace de L^2, A_j sera un projecteur (idempotent).

On construira une analyse multi-résolution à l'aide de sous-espaces V_j emboîtés les uns dans les autres, tels que le passage de l'un à l'autre soit le résultat d'un changement d'échelle (zoom).

Par exemple, dans le cas dyadique on aura :

$$f(x) \in V_j \Leftrightarrow f\left(\frac{x}{2}\right) \in V_{j+1} \qquad (2.21)$$

Ce qui correspond à une dilatation d'un facteur 2. L'espace V_{j+1} contient des signaux plus "grossiers" que l'espace V_j et il est clair que :

L'axiomatique correspondante peut s'exprimer comme suit :

Soit un ensemble de sous espaces de L2 tels que :

$$..... \subset V_2 \subset V_1 \subset V_0 \subset V_{=1} ..... \subset V_{j+1} \subset V_j ..... \qquad (2.22)$$

$$\bigcup_{j \in Z} V_j = L^2(R) \qquad (2.23)$$

$$\bigcap_{j \in Z} V_j = \{0\} \qquad (2.24)$$

$$\forall j \in Z \; si \; f(x) \in V_j \Leftrightarrow f(2^{-1}x) \in V_{j+1} \;\; (ou \; f(2^j) \in V_0 \qquad (2.25)$$

$$\forall \; j \in Z \; si \; f(x) \in V_0 \Leftrightarrow f(x-k) \in V_0 \; (in\,\mathrm{var}iance \; par \; translation) \qquad (2.26)$$

Cet ensemble définit une analyse multirésolution de $L^2(R)$.

Remarque 1 : La propriété 2.27 assure la convergence de l'analyse et peut aussi s'écrire :

$$\lim_{j \to -\infty} V_j = L^2(R) \qquad (2.27)$$

On dit parfois que $\bigcup_{-\infty}^{+\infty} V_j$ et dense dans $L^2(R)$

Dans ces conditions, on peut montrer qu'il existe une fonction dite fonction d'échelle qui par dilatation et translation engendre une base orthonormée de V_j. Cette fonction sera notée :

$$\varphi(x) \in L^2(R) \qquad (2.28)$$

Et les fonctions de bases sont construites suivant la relation :

$$\varphi_{j,n}(x) = 2^{\frac{-j}{2}} \varphi(2^{-j}x - n) \; avec \; n \in Z \qquad (2.29)$$

Il suffit d'ailleurs que $\varphi(.,-n)$ soit une base de V_j.

La base sera orthonormée si :

$$\int_{-\infty}^{+\infty} \varphi(x)\varphi^*(x+n)dx = \delta(n) \; \forall n \in Z \qquad (2.30)$$

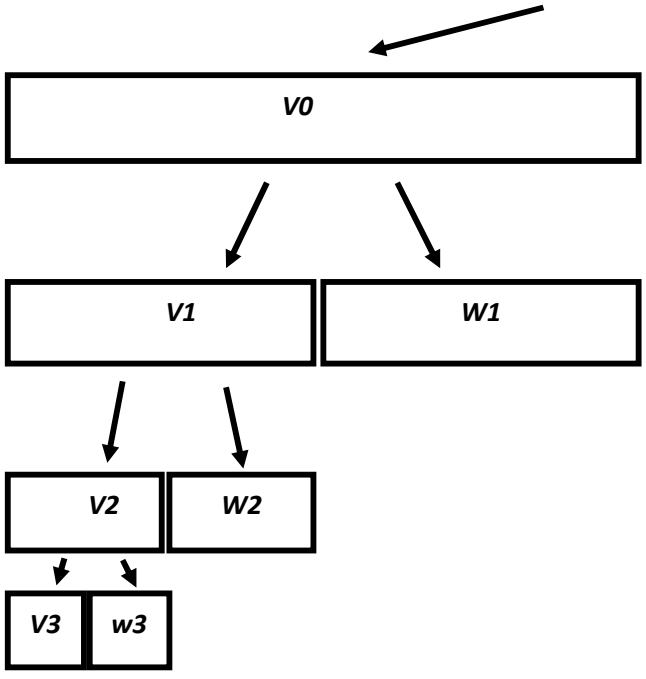


Figure 2.7 : Schéma de l'analyse multi-résolution

La relation d'orthogonalité entre les fonctions de base pour une échelle donnée 2.31 pourra donc s'écrire :

$$\left\langle \varphi_{j,n}, \varphi_{j,k} \right\rangle = \delta(n-k) \ \forall \ n,k,\ j \in Z \qquad (2.31)$$

Remarque 2 : On peut utiliser plusieurs fonctions pour construire par translation une base du sous-espace V_0, cette liberté est mise à profit dans la construction des multi-ondelettes. Les fonctions doivent, bien entendu, être orthogonales entre elles.

L'action du projecteur sur *f* fournira sa décomposition sur la base des fonctions d'échelle et les coefficients de cette décomposition constituent l'approximation à l'échelle *j* de *f*.

$$A_j f = \sum_n \left\langle f, \varphi_{j,n} \right\rangle \varphi_{j,n} \qquad (2.32)$$

On pose :

$$a_n^j = \left\langle f, \varphi_{j,n} \right\rangle \qquad (2.33)$$

L'approximation à la résolution *j* de la fonction *f* sera définie par la suite discrète des nombres (réels ou complexes) a_n^j.

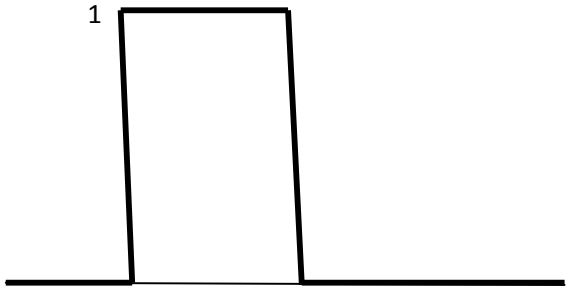


Figure 2.8 : Fonction d'échelle de l'analyse de Haar

Une suite numérique formée par échantillonnage d'un signal continu réel pourral base étant orthonormée, la norme de la fonction (l'énergie) peut être calculée à partir de ses "coordonnées" :

$$\left\| A_j f \right\|^2 = \sum_{n=-\infty}^{+\infty} \left| a_n^j \right|^2 \qquad (2.34)$$

5.3 Espaces des détails

L'espace des détails vient compléter l'analyse.

On peut définir pour chaque V_j son complément orthogonal W_j dans V_{j-1} tel que :

$$V_{j-1} = V_j \oplus W_j \qquad (2.35)$$

$$L^2 = \oplus_{j \in Z} W_j \qquad (2.36)$$

Comme W_{j-1} est orthogonal à V_{j-1}, alors W_{j-1} sera orthogonal à W_j ; cette propriété s'écrit :

$$\forall\, j \quad k \neq j \quad alors\, W_j \perp W_k \qquad (2.37)$$

Les sous-espaces W_j ne forment pas une famille d'espaces emboîtés, mais les propriétés d'échelle et d'invariance par translation sont conservées.

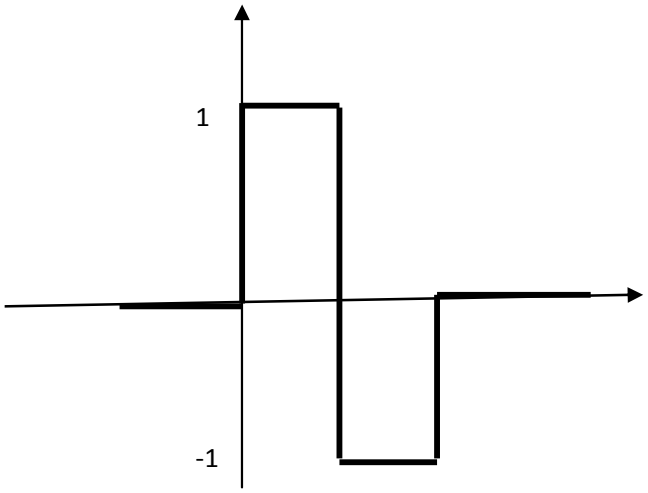


Figure 2.9. Ondelette mère de l'analyse de Haar

Dans ces conditions, on peut montrer qu'il existe une fonction appelée ondelette qui par dilatations et translations engendre une base orthonormée des W_j et donc de L^2.

Cette fonction est notée :

$$\psi(x) \in L^2(R) \qquad (2.38)$$

Et les fonctions de base sont construites suivant la relation :

$$\psi_{j,n}(x) = 2^{\frac{-j}{2}} \psi(2^{-j}x - n) \; avec \; n \in Z \qquad (2.39)$$

L'orthonormalité de la base d'ondelettes s'écrit :

$$\langle \psi_{j,n}, \psi_{j,k} \rangle = \delta(j-i)\delta(n-k) \; \forall \; i, j, n, k \in Z \qquad (2.40)$$

L'approximation à l'échelle immédiatement plus fine pourra donc être reconstruite en utilisant les détails du signal fournis par sa projection sur la base de W_j sj,nuivant la relation suivante :

$$A_{j-1} = A_j f + \sum \langle f, \psi_{j,n} \rangle \psi_{j,n} \qquad (2.41)$$

On notera D_j le projecteur sur W_j et le signal de détail sera décrit par la suite numérique :

$$d_n^j = \langle f, \psi_{j,n} \rangle \qquad (2.42)$$

Donc :

$$D_j f = \sum \langle f, \psi_{j,n} \rangle \psi_{j,n} \qquad (2.43)$$

Et la formule de reconstruction s'écrit :

$$A_{j-1} = A_j f + D_j f \qquad (2.44)$$

Le signal de détail est constitué d'une suite numérique dont les éléments sont aussi les coefficients de la transformée en ondelettes.

5.4 Algorithme pyramidale (algorithme rapide de Mallat)

Nous avons vu que l'AMR d'un signal revient à le décomposer à différentes échelles, en approximations et en détails. S. Mallat propose un algorithme rapide permettant de calculer les coefficients de détails et d'approximations en utilisant des filtrages et décimations successifs [33].

La figure 2.10 présente cet algorithme : Les coefficients de détails correspondant à l'espace W1 sont obtenus par filtrage passe-haut (filtre h1) puis décimation par 2, les approximations sont obtenues de la même manière par filtrage passe-bas (g1). Pour obtenir les coefficients de détails aux résolutions supérieures, il suffit de réitérer ces étapes sur les coefficients d'approximations.

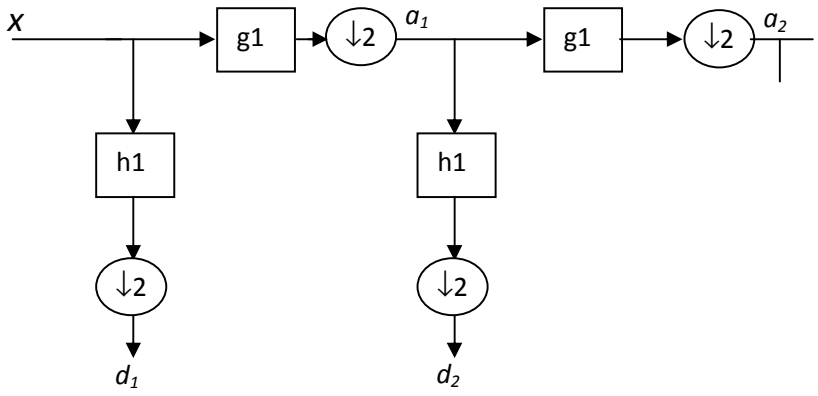


Figure.2.10 : Algorithme pyramidal de Mallat (les a_i sont les coefficients d'approximations, les d_j sont détails).

On peut reconstruire le signal grâce à des filtres h2 et g2 selon l'algorithme présenté en figure.2.11. L'approximation an à un niveau donné n est la somme des coefficients de détails d_{n+1} et d'approximations a_{n+1} du niveau supérieur préalablement filtrés et reéchantillonnés.

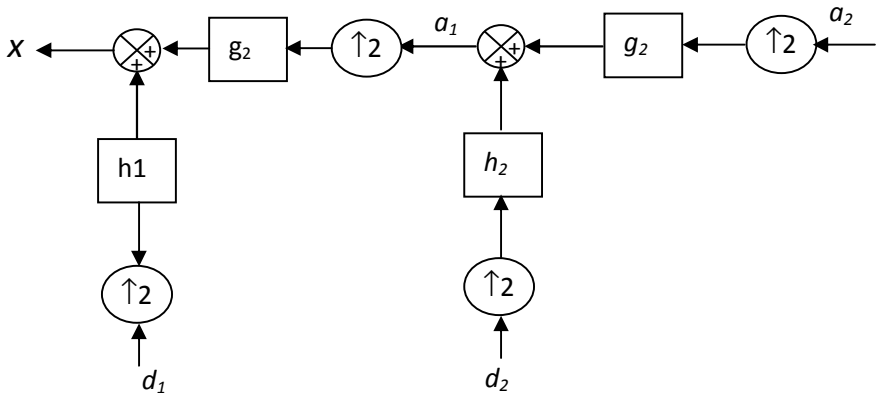


Figure.2.11 : Algorithme de reconstruction du signal.

Les filtres h1 et g1 sont liés aux filtres exhibés par les relations à deux échelles 3.6 et 3.7. Les conditions imposées par la définition de l'AMR impliquent que le filtre g soit passe bas, on a de plus :

$$\begin{cases} h_1(k) = \bar{h}(k) = h(-k) \\ g_1(k) = \bar{g}(k) = g(-k) \end{cases} \quad (2.45)$$

Les filtres de reconstruction h2 et g2 sont liés aux fonctions des bases duales. Dans le cas où les bases sont orthonormales on a :

$$\begin{cases} h_2(k) = h(k) \\ g_2(k) = g(k) \end{cases} \quad (2.46)$$

Dans le domaine fréquentiel :

$$\begin{aligned} &|H(w)|^2 + |H(w+\pi)|^2 = 2 \\ &|H(w)| = |G(w+\pi)| = |G(w-\pi)| \\ &|H(w)|^2 + |G(w)|^2 = 2 \end{aligned} \quad (2.47)$$

Les filtres sont alors symétriques par rapport à $\pi/2$, de puissances complémentaires. Ce sont des filtres miroirs en quadrature (QMF, Quadrature Mirror Filter).

La figure 2.12 présente la décomposition en base d'ondelettes du signal x composé de *N* points. L'arbre de décomposition a $\log_2(N)$ niveaux. A chaque niveau, la résolution temporelle est divisée par 2, au dernier niveau les coefficients de détails sont représentés par un unique point, la résolution temporelle est nulle et la résolution fréquentielle est maximale. Le signal est décomposé en *N-1* coefficients de détails et 1 coefficient d'approximations (composante de plus basse fréquence du signal).

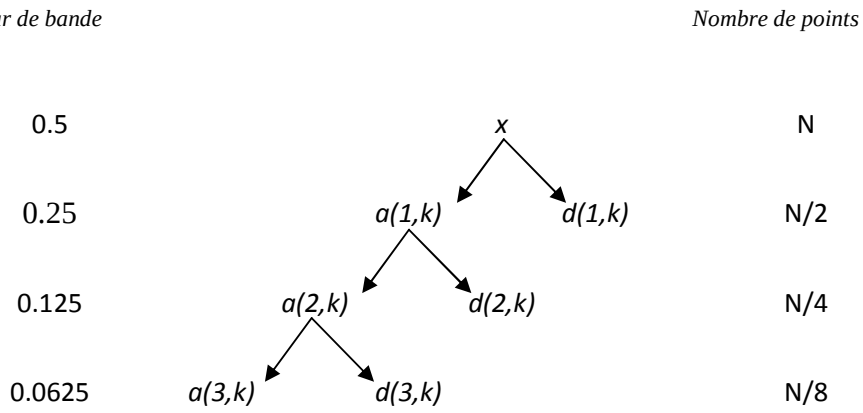


Figure.2.12 : Arbre de décomposition d'un signal sur une base d'ondelettes

5.5. Généralisation aux images

La généralisation de l'AMR aux signaux de plusieurs dimensions ne pose aucune difficulté d'ordre théorique si l'on utilise des ondelettes séparables : Soient ϕ et ψ les fonctions échelle et ondelette générant une base d'ondelettes orthonormale de $L^2(R)$, on définit les fonctions ondelettes séparables à deux dimensions de la façon suivante, [34,35] :

$$\begin{cases} \psi^1(x,y) = \phi(x)\psi(y) \\ \psi^2(x,y) = \psi(x)\phi(y) \\ \psi^3(x,y) = \psi(x)\psi(y) \end{cases} \qquad (2.48)$$

En reprenant les notations utilisées précédemment :

$$\text{Pour } 1 \le i \le 3, \quad \psi^{i}_{j,n,m}(x,y) = \frac{1}{2^j}\psi^i\left(\frac{x-2^j n}{2^j}, \frac{y-2^j m}{2^j}\right) \qquad (2.49)$$

alors la famille d'ondelettes : $\left\{\psi^1_{j,n,m}(x,y), \psi^2_{j,n,m}(x,y), \psi^3_{j,n,m}(x,y),\right\}_{j,n,m\in Z^3}$, définit une base biorthonormale de $L^2(R^2)$. La fonction échelle associée est $\phi^0(x,y) = \phi(x)\phi(y)$.

En pratique, pour calculer les coefficients d'approximations et de détails d'une image I, nous utilisons la généralisation de l'algorithme pyramidal présenté aux figures 2.13 et 2.14. Chaque étape de cet algorithme est appliquée successivement aux lignes puis aux colonnes de

l'image. On obtient pour un niveau de décomposition une sous-image d'approximations $a_j(n,m)$ et trois sous-image de détails : $d_j^1(n,m), d_j^2(n,m), d_j^3(n,m)$ selon l'orientation fréquentielle (horizontale, verticale et diagonale). L'arbre de décomposition et de reconstruction quaternaire est représenté en figure 2.14. La décomposition est représentée sous la forme d'une image où les basses fréquences sont en haut à gauche, les hautes en bas à droite (figure 2.14). La figure 2.16 présentent cette décomposition pour l'image X et une image composée d'une croix diagonale dans un carré. Cette dernière figure 2.16 permet d'observer les orientations fréquentielles des coefficients de détails.

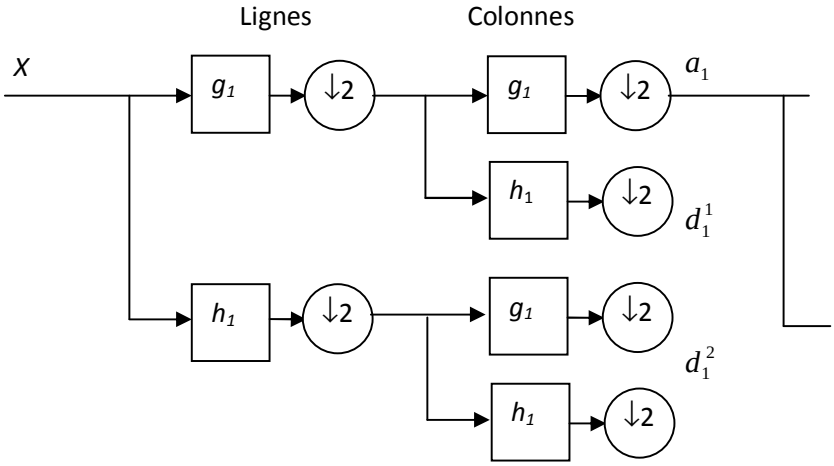


Figure.2.13 : Algorithme pyramidal de décomposition d'une image

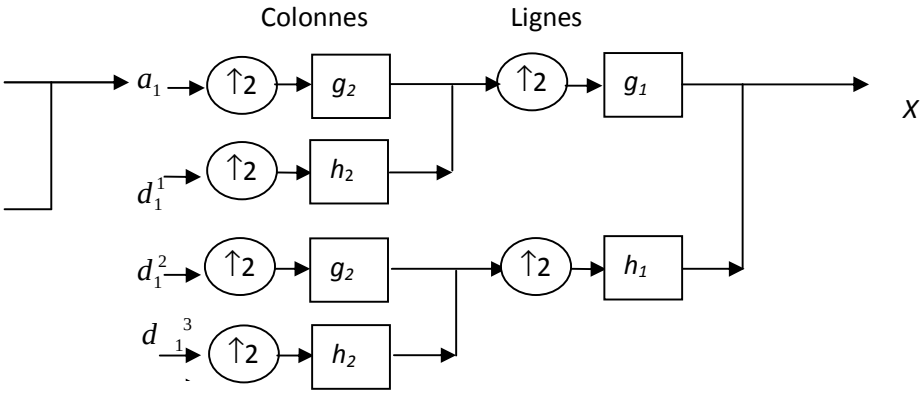


Figure.2.14 : Algorithme pyramidal de reconstitution d'une image

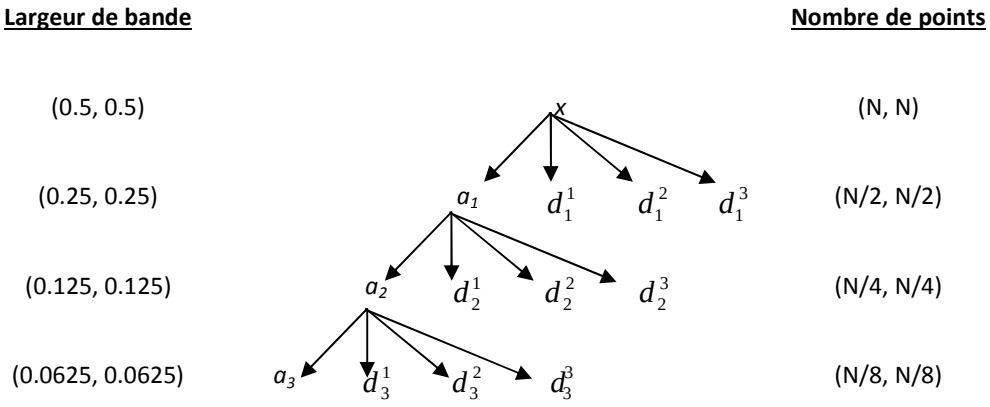


Figure.2.15: Arbre de décomposition d'une image sur une base d'ondelettes

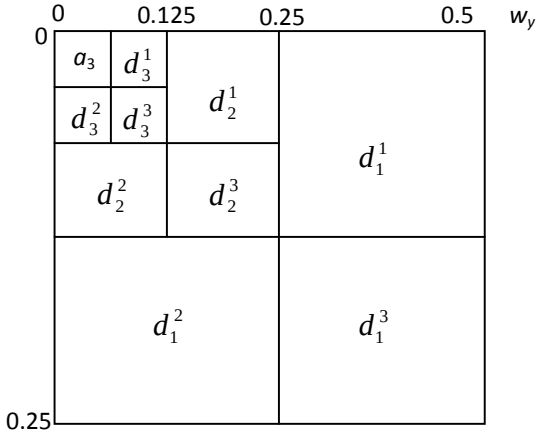


Figure.2.16 : Disposition des coefficients de décomposition d'une image X pour la profondeur 3

6. Conclusion

Dans ce chapitre nous avons rappelé les transformées directionnelles et on constate que un nombre très restreint de ces transformée directionnelles a été appliqué à la compression. En effet, la plupart de ces transformées ont le désavantage d'être redondantes.

Par contre la transformée en ondelettes peut être assimilée à une segmentation fréquentielle de l'information contenue dans l'image numérique à la manière d'un filtrage horizontale et verticale présentant une structure pyramidale.

Les principes de base de l'analyse multirésolution et des ondelettes ont été posés par Meyer (Meyer 1986,1988) pour les aspects mathématiques, puis Stéphane Mallat (Mallat 1989) pour les aspects signaux et images.

Nous avons examine dans ce chapitre comment les principes de l'analyse multirésolution multidimensionnels présentés précédemment peuvent être étendus aux images numériques.

L'intérêt de l'analyse multirésolution est d'avoir une structure pyramidale, cette structure pyramidale va être exploite dans notre codeur que nous avons proposé dans le chapitre 3 ou se trouve les sous bandes qui sont le résultat de la transformée en ondelettes après plusieurs résolutions.

Le codeur hiérarchique basé sur une telle transformation peut ainsi prendre en compte la redondance spatiale et fréquentielle de l'image et devient par conséquent plus efficace.

Chapitre 3 : Codeur SPIHT, Proposition d'une Optimisation MSPIHT

Résumé :

Notre travail dans ce chapitre consiste en l'étude des algorithmes de compression par ondelettes utilisant les codages imbriqués SPIHT et la proposition d'une optimisation de cet algorithme. Notre approche nommée MSPIHT (Modified-SPIHT) est basée sur deux points :

- *On cherche toujours à minimiser les bits à coder après quantification. Pour cela, et après la transformation de l'image par ondelettes en trois résolutions, on forme la structure pyramidale. On cherche les fils directs insignifiant de chaque parents de la troisièmes résolution et puis on les code par un seul bit au lieu de quatre comme dans le SPIHT.*
- *On utilise un nouveau programme de décodage qui permet une bonne reconstitution.*

Ce chapitre détaillé le principe de l'algorithme MSPIHT . Une étude comparative entre les deux algorithmes est aussi présentée (SPIHT et MSPIHT).

1. Introduction

Le but de ce chapitre est de présenter l'algorithme SPIHT et ensuite notre algorithme MSPIHT afin de montrer l'amélioration qui est apportée.

L'utilisation de la transformée en ondelettes dans l'étape de transformation offre une analyse multirésolution de l'image. Cette propriété est très intéressante pour la compression progressive des images numériques.

Ce chapitre se présente donc comme suit : après un bref rappel sur la méthode originale de quantification développée par Amir Said (SPIHT), on propose une optimisation de l'algorithme SPIHT nommée MSPIHT (Modified-SPIHT).

2. Principe de l'algorithme SPIHT

La transformée en ondelettes retenue pour la compression d'images numériques dans le standard du SPIHT et notre optimisation MSPIHT, est la transformée en ondelettes discrète. C'est celle qui permet d'atteindre les meilleures performances en compression. Le nombre de niveaux de décomposition en ondelettes a été fixé à trois résolutions. Ce nombre de niveaux de décomposition est un compromis entre performance en compression et complexité calculatoire. De plus, un faible niveau de décomposition réduit le nombre de pixels affectés par la perte de données compressées. La décomposition en ondelettes sur trois niveaux d'une image est représentée sur la figure 3.1.

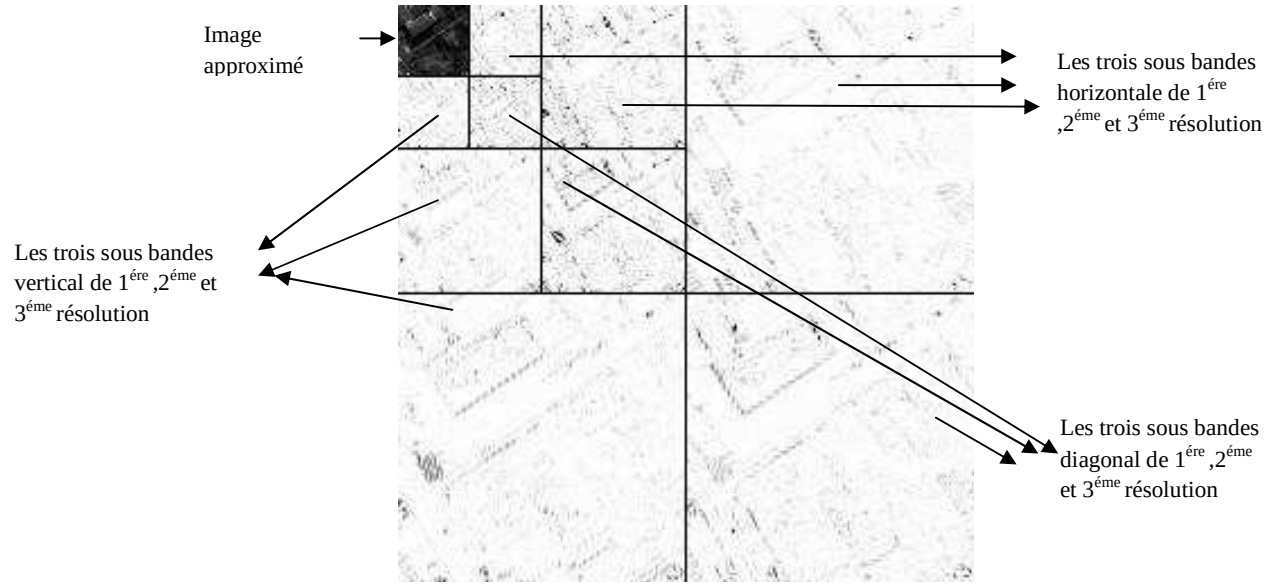


Figure 3.1 : Exemple de décomposition de l'image satellite par ondelettes en trois résolutions.

L'objectif de la transmission progressive de l'image est de sélectionner l'information la plus importante pour la transmettre en premier. L'information la plus importante est celle qui apportera, lors de sa réception, la correction la plus importante de la distorsion du signal. SPIHT utilise l'erreur quadratique moyenne comme métrique (EQM). Ainsi le coefficient de la transformée de plus large amplitude doit être transmis en premier, de même son bit de poids le plus fort avant les bits de poids plus faibles.

Cependant, la relation entre EQM et qualité d'image perçue n'est pas systématique. Il faut noter que les coefficients de la transformée peuvent avoir une valeur importante sans que ces derniers n'apportent de détails importants au niveau de la perception visuelle de l'image. C'est le cas par exemple lorsqu'un coefficient important se situe dans une zone fortement bruitée ; il n'apporte pas beaucoup d'information pour améliorer la qualité de l'image.

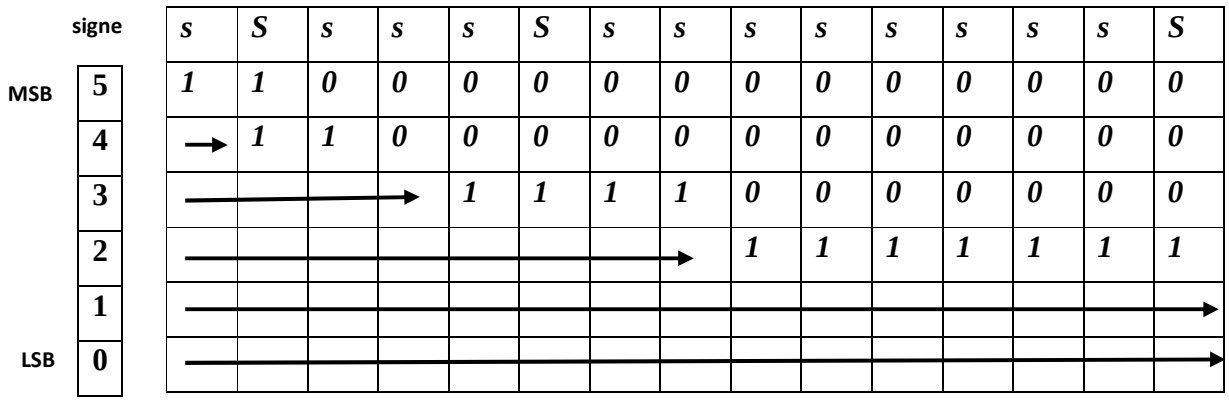


Fig. 3.2 : Représentation binaire des coefficients triés selon leurs amplitudes.

SPIHT propose de trier les coefficients suivant leurs amplitudes afin de transmettre l'information de façon progressive. Les bits de poids forts seront transmis en premier puis par les étapes de raffinement successives, les bits de poids plus faibles seront ensuite transmis. La figure 3.2 montre ces coefficients (codés sur 7 bits, un bit de signe (s) et 6 bits pour la valeur) en colonnes triées. Le bit de poids le plus fort (MSB) est ici noté 5, et le bit de poids le plus faible (LSB) par 0. Dans cet exemple, les deux indices de plus fortes amplitudes, correspondant aux deux premières colonnes, sont transmis en priorité dans le flux. Leurs signes sont d'abord codés, puis intervient une phase de raffinement qui consiste à transmettre le bit le plus significatif de l'indice pas encore codé. Le problème lié à l'ordonnancement des indices selon leurs amplitudes est qu'il faut transmettre de l'information supplémentaire pour localiser spatialement chaque indice transmis dans un espace à deux dimensions, ce qui, pour *i* indices, représente 2^i entiers. Pour pallier à ce problème, l'algorithme de tri des partitions d'ensembles ne transmet pas explicitement l'ordre des données. Le principe est basé sur le fait que le décodeur est capable de reproduire le chemin d'exécution de l'encodeur puisque cet ordre est défini par des règles communes. Avec un tel algorithme, il n'est pas nécessaire de trier complètement les coefficients mais seulement de détecter ceux compris entre $|2^n|$ et $|2^{n+1}|$, n étant le rang du bit transmis.

L'information la plus importante d'une image est contenue dans les basses fréquences. En conséquence, la variance décroît plus on se trouve dans les bas niveaux de la pyramide des sous-bandes. De plus il a été observé qu'il existe une dépendance entre les niveaux des sous-bandes, et plus précisément qu'un indice a généralement une plus petite amplitude que l'indice du niveau

supérieur à la même localisation spatiale dans l'image. L'utilisation d'un arbre spatial permet de hiérarchiser les données, les indices des plus bas niveaux de résolutions seront transmis avant les autres. La figure 3.3 illustre un arbre utilisé dans le cadre de l'algorithme SPIHT. La représentation de l'arbre spatial est la suivante : un nœud est soit une feuille soit pointe vers quatre fils, un carré de 2×2 pixels adjacents. Les pixels du plus haut niveau de la hiérarchie sont les racines. Un fils d'un nœud correspond aux pixels à la même position spatiale mais avec un niveau plus fin dans la pyramide. Cet algorithme repose sur la gestion de trois listes : les coefficients significatifs (LSP), les coefficients insignifiants (LIP) et les ensembles insignifiants (LIS).

La liste des coefficients significatifs est initialement vide, tandis que la liste de coefficients insignifiants contient les racines de chaque arbre (coefficients de la bande basse fréquence) et la liste des ensembles insignifiants contient l'ensemble des descendants de chaque arbre.

Cette partition initiale est segmentée récursivement au moyen de deux règles. Si un ensemble de descendants d'un nœud est significatif, il est séparé en quatre coefficients fils directs de ce nœud, et l'ensemble des autres descendants. Les fils directs sont ajoutés à la LIP ou à la LSP en fonction de leur signifiance.

Si au moins un élément de l'ensemble des autres descendants est significatif, cet ensemble est séparé en quatre ensembles insignifiants ajoutés à la LIS. Le fait de traiter les coefficients par groupes de quatre permet d'effectuer un codage entropique efficace par la suite.

La passe de raffinement consiste à coder progressivement les bits de poids plus faibles des coefficients significatifs.

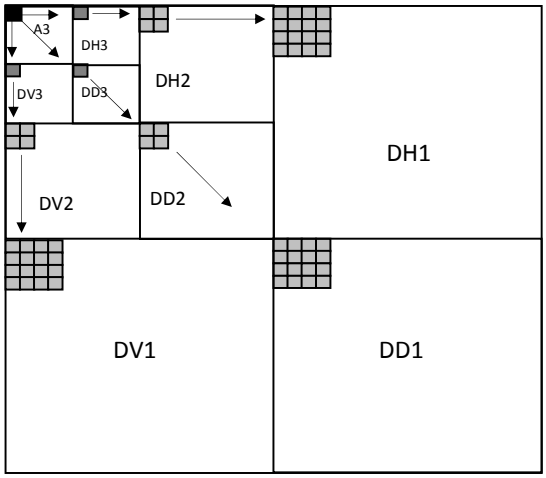


Figure 3.3 : Modèle de dépendances inter-bandes pour SPIHT

3. Algorithme proposé MSPIHT

La structure de MSPIHT est aussi un arbre comme le codeur SPIHT [36], chaque parent de la sous-bande de plus basse fréquence (A3 dans la figure 3.3) (c'est à dire chaque coefficient ondelette) admet quatre fils dans la sous-bande de même orientation et de résolution juste supérieure. En ce qui concerne la sous-bande de plus basse fréquence (image approximée (A3 dans la figure 3.3)), chaque coefficient admet trois fils : un dans chacune des trois sous-bandes correspondant à la même résolution.

Nous pouvons organiser sa descendance sous la forme de quatre ensembles :

- Un ensemble regroupant les descendants directs de $(i, j) \rightarrow O(i, j)$;
- Un ensemble regroupant tous les descendants de $(i, j) \rightarrow D(i, j)$ avec $O(i ,j) \in D(i ,j)$;
- Un ensemble $L(i ,j)$ défini comme étant : $D(i ,j) - O(i ,j)$;
- Un ensemble regroupant les descendants directs de (i, j) de la dernière résolution$\rightarrow$ $DO(i, j)$

Le mode de partitionnement sous forme de listes à lui aussi été revu. Ainsi, les coefficients sont regroupés en trois listes notées *LSP*, *LIP* et *LIS* correspondant respectivement à la liste des coefficients significatifs, la liste des coefficients non-significatifs et enfin à la liste des coefficients dont les ensembles de descendance sont non-significatifs. Les listes sont non distinctes et leur initialisation se fait comme suit:

- $LSP = \emptyset$;
- *LIP* contient les coordonnées de tous les coefficients de la dernière résolution ;
- *LIS* contient les coordonnées des coefficients (de la sous-bande de plus basse fréquence) qui admettent des descendants.

L'algorithme MSPIHT se déroule en trois passes :

- Une pour l'étude de la signifiance des coefficients.
- Une pour l'insignifiance des fils directe de la troisième résolution.
- Enfin une pour le raffinement.

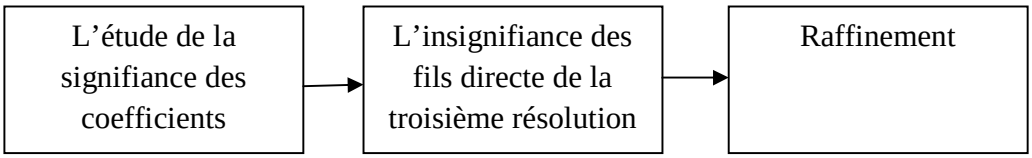


Figure 3.4 : Bloc de diagramme de l'algorithme MSPIHT

Une méthode avec codage entropique (arithmétique) est aussi utilisée et permet d'obtenir des PSNR supérieurs à ceux lorsqu'on n'utilise pas le codage entropique (codage MSPIHT seulement) (chapitre 4).

3.1 Test de signifiance MSPIHT

Soit le seuil *n* initialisé comme suit :

$$n=\left\lfloor \log_2(\max_{(i,j)\in image}\left|coeff_{(i,j)}\right|)\right\rfloor \qquad (3.1)$$

Une fois le seuil fixé, nous pouvons définir la fonction de signifiance d'un ensemble τ *(équation 3.2)*, notée S_n, par l'expression suivante :

$$s_n(\tau)=\begin{cases}1 & si \quad \max_{(i,j)\in T}\left|coeff(i,j)\right|\geq 2^n \\ 0 & \text{si } non\end{cases} \qquad (3.2)$$

Dans un premier temps il nous faut parcourir la *LIP* à la recherche des coefficients tels que

$$s_n(\{i,j\})=1 \qquad (3.3)$$

Les coordonnées de chaque coefficient de la *LIP* qui vérifie l'équation (3.3) passent de la *LIP* à la *LSP* et le signe du coefficient est transmis au décodeur. Une fois le parcours de la *LIP* terminé, nous nous attachons à traiter la *LIS*. Etant donné la définition des sites donnée précédemment, il peut y avoir deux types de coefficients dont les coordonnées se trouvent dans la *LIS* :

- Les coefficients de type *A* sont ceux dont la descendance est de type *D(i , j),* ce dernier étant un site non significatif ;
- Les coefficients de type *B* sont ceux dont la descendance non significative est de type *L(i ,j)* c'est à dire ceux dont les descendants directs sont devenus significatifs.

Dés lors, nous parcourons la LIS de façon à faire évoluer le partitionnement. A cet effet nous étudions la valeur de la fonction de signifiance de *D(i ,j*) (resp *L(i ,j))* pour les coefficients de type A d'arbres de zéros (resp. type B d'arbres de zéros).

La seconde phase de l'algorithme n'est autre qu'une phase de raffinement des coefficients donnés comme significatifs à la passe (n-1) et qui sont donc présents dans la *LSP*.

A chaque passe un nouveau seuil est utilisé, le passage d'un seuil au suivant est effectué par un décrément de un.

3.2 Test d'insignifiance MSPIHT

Notre approche MSPIHT réside dans le test d'insignifiance des coefficients (les fils direct de la troisième résolution) dans le cas où ces coefficients sont trouvés insignificatifs on ajoute un bit de zéro à l'ensemble *DO(i,j)* et en code les quatre fils insignifiant par un seul bit au lieu de quatre dans le SPIHT.

Si les coefficients ne sont pas tous insignificatifs (un ou plus sont significatif) on ajoute un bit de un a l'ensemble *DO(i,j)* et en code chaque bit séparément .

3.3 Algorithme du codeur MSPIHT

Sortie $n = \left|\log_2(\max_{(i,j)\in image}) \left| coeff_{(i,j)} \right|\right| \Leftrightarrow T_i = 2^{\left|\log_2 \max_{(i,j)\in' i,j)}\right|}$

$LSP = \phi$ *et* $LIP = \{(i, j)\} \in$ *image*

LIS contient les mêmes coefficients que *LIP* excepté ceux qui n'ont pas de descendants.

1. Pour chaque $(i, j) \in LIP$ faire :
   - 1.1 Sortie $S_n\{(i, j)\}$
   - 1.2 Si $S_n\{(i, j)\} = 1$ alors mettre (i, j) dans LSP et coder le signe de (i, j)
2. Pour chaque $(i, j) \in LIS$ faire :
   - 2.1 Si l'entrée est de type A
     - a) Sortie $S_n\{D(i, j)\}$
     - b) si $S_n(D(i, j)) = 1$ alors
     - ➢ si $S_n(DO(i, j)) = 1$
     - ➢ mettre $DO(i, j) = 1$

- Pour chaque $(i, j) \in O(i, j)$ faire :
- Sortie $S_n\{(i, j)\}$
- si $S_n(i, j)$=1 alors mettre (l,m) dans LSP et coder le signe de (l,m)
- si $S_n(i, j)$= 0 alors mettre (l,m) à la fin de LIP

c) si $S_n(DO(i, j)) = 0$

- mettre $DO(i, j) = 0$ coder tous les (l,m) par un seul bit
- Si $L(i, j) \neq \phi$ alors mettre (l,m) à la fin de LIS comme une entrée de type B

2.2 Si l'entrée est de type B

a) Sortie $S_n\{L(i, j)\}$

b) si $S_n(L(i, j)) = 1$ alors

- mettre (l,m) à la fin de LIS comme une entrée de type A
- supprimer (i, j) de LIS

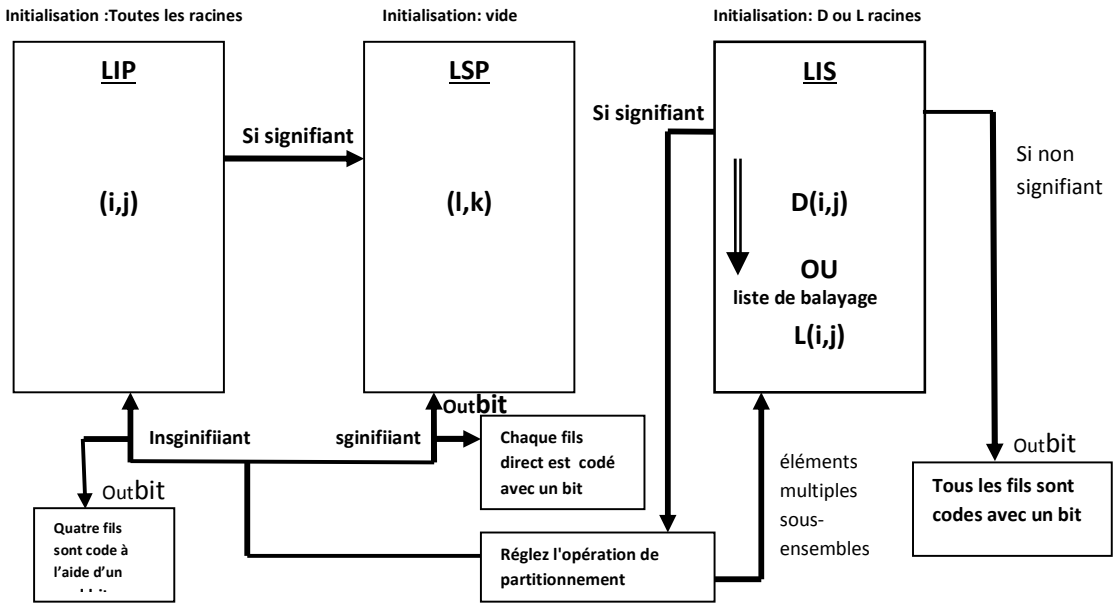


Figure 3.5: Processus de regroupement des coefficients

3.4 Raffinement

La première étape étant terminée, nous passons maintenant à la seconde phase qui s'attache au codage des éléments de la liste LSP. En effet, c'est dans cette liste que se trouvent les amplitudes des coefficients significatifs par rapport au seuil T_i et il nous faut les quantifier puis les coder.

La particularité de l'algorithme étant de réaliser un flux emboîté. L'étape de quantification se déroule en deux temps :

- Raffinement des coefficients qui étaient significatifs aux passes précédentes;
- Quantification des nouveaux arrivants dans la liste LSP.

Nous utilisons pour cela un quantificateur scalaire uniforme défini par rapport au seuil T_i fixé par la passe de travail.

Si globalement les coefficients significatifs sont compris dans l'intervalle $[T_i, 2T_0[$, l'intervalle concernant les nouveaux arrivant peut se restreindre à $[T_i, T_{i-1}[$ tandis que les autres, ceux qu'il faut simplement raffiner, se trouvent dans $[T_{i-1}, 2T_0[$. A chaque passe, nous réalisons un raffinement des intervalles d'incertitude en les divisant par deux. Ainsi, l'indice de quantification ″0″ est mis pour les coefficients appartenant à la première moitié de l'intervalle alors que l'indice de quantification ″1″ est utilisé pour la seconde moitié, (Figure 3.5.1, 3.5.2) :

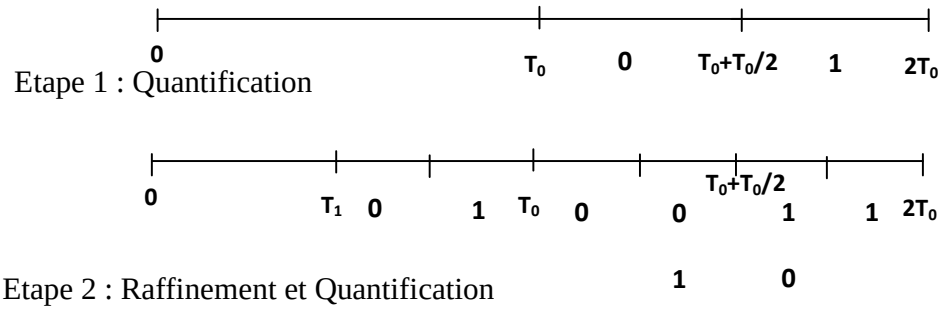


Figure 3.6 : étape de raffinement

3.5 Quantification

La figure 3.7 présente comme exemple une matrice qui contient un coefficient d'ondelettes dans la sous bande approximée et ces details de toute les autre sous bande dans la structure pyramidale).

Un bit correspondant à 2^{j-1} est émis pour toutes les valeurs importantes dans la liste LSP afin d'augmenter la précision de ces valeurs transmises [12]. Les valeurs significatives {63, -34, 49 et 47} de la matrice test (figure 3.7) sont quantifiées respectivement par les bits "1 0 1 0" [13]. Ensuite, l'étape B de l'algorithme est répété sur le résidu d'image en incrémentant j par un. Ce processus est répété jusqu'à ce que la qualité de l'image reconstruite est souhaitée ou jusqu'à ce que le nombre de bits transférables requise est dépassé.

	0	1	2	3	4	5	6	7
0	63	-34	49	10	7	-13	12	7
1	-31	23	-14	-13	3	4	6	1
2	**15**	**14**	3	-12	5	-7	3	9
3	**-9**	**-7**	-14	8	4	-2	3	9
4	-5	9	-1	**47**	4	-6	-2	2
5	3	0	-3	2	2	-2	0	4
6	2	-3	6	4	3	6	3	6
7	5	11	5	6	0	3	-4	4

Figure 3.7 : Exemple 1 décomposition à trois résolutions d'une matrice de taille 8x8.

4. Différence entre SPIHT et MSPIHT

La différence entre l'algorithme MSPIHT que nous proposons et l'algorithme SPIHT réside dans le processus de test utilisé pour l'insignifiance de l'ensemble des coefficients de tous les descendants de coefficient (i, j) et la procédure de codage utilisé pour les symboles outbit.

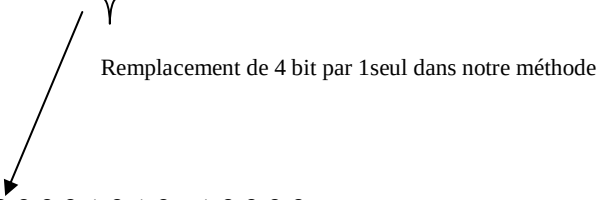
4.1 Processus de teste d'insignifiance

Les résultats des algorithmes SPIHT et MSPIHT appliqués sur l'exemple 1 de la matrice de figure 3.7 sont comme suit :

SPIHT

Out Bit : +1 -1 0 0 1 +1 0 0 0 1 0 0 **0 0 0 0** 1 0 1 0 +1 0 0 0 0

LSP : 1 0 1 0



MSPIHT

Out Bit : +1 -1 0 0 1 +1 0 0 0 10**0** 0 0 0 1 0 1 0 +1 0 0 0 0

LSP : 1 0 1 0

Comme dans l'exemple 1, la matrice (cf. figure 3.7) des descendants direct (-31) sont encodées dans MSPIHT avec un seul symbole "0" dans la liste outbit au lieu de quatre symbole "0000" dans l'algorithme SPIHT.

Les paramètres initiaux de MSPIHT sont:

- Le seuil initial est fixé à 32 (32 est le max des éléments de la matrice de la figure 3.7 divisé sur deux)
- La notation (i, j) A ou (i, j), B indique que l'entrée de la liste LIS est de type 'A' ou 'B', respectivement (tableau 1).

Notez la duplication des coordonnées dans les listes, comme l'établit dans la liste LIS sont des arbres sans racines. Le coefficient (0,0) n'est pas considéré comme une racine.

MSPIHT commence le codage par des pixels individuels 'importants dans la liste LIP. Quand un coefficient est jugée significatif, il est déplacé vers la liste LSP, et son signe est également codé. Nous avons utilisé la notation +1 et -1 pour indiquer le signe, quand un coefficient est significatif immédiatement suivi par un bit de signe.

Après le test des pixels en commence à tester les ensembles contenons dans LIS .

Dans cet exemple (figure 3.7) *D (1, 0)* est l'ensemble des 20 coefficients {(2,0), (3,0), (2,1), (3,1), (4,0), (5,0), (6,0), (7,0), (4,1), (5,1), (6,1), (7,1), (4,2), (5,2), (6,2), (7,2), (4,3), (5,3), (6,3), (7,3)}.

D (1, 0) est significatif, MSPIHT teste l'insignifiance des quatre fils direct {2,0), (3,0), (2,1), (3,1)}. Pour savoir si *DO(1 ,0)=0 ou DO(1 ,0)=1*.

Après le teste de tous les descendants, si *DO(1 ,0)=0* tous les fils sont codées par un seul bit de zéro dans la liste outbit de l'algorithme MSPIHT au lieu à quatre bits de zéro lorsque on utilise algorithme SPIHT. (1, 0) est déplacé à la fin de la LIS, et on change le type 'A' en type 'B', ceci entraine que la prochaine test de l'ensemble (1,0) passe de l'ensemble des descendent *D(1,0)* à *L(1,0)* (Tableau 1).

Si *DO(1 ,0)=1* on code chaque coefficient avec un bit à zéro si insignifiant ou un bit à un si le bit est sinificatif . (1, 0) est déplacé à la fin de la LIS, et ses changements de type 'A' par 'B', ce qui signifie que la signification nouvelle entrée LIS changé de *D (1, 0) à L (1,0)*. Même procédure s'applique pour les ensembles *D (0, 1) et D (1, 1).*

4.2 Codage de l'ensemble outbit

- Si *D (i, j)* = 0 le fils et leurs fils sont tous codés par un bit zéro.
- Si *D (i, j)* = 1 et *DO (i, j)* = 1 (un ou plusieurs fils est significatif chaque fils directs est codé par un bit) (tableau 1).
- Si *D (i, j)* = 1 et *DO (i, j)* = 0 (fils directs sont insignifiants l'ensemble des quatre fils est codé par un bit à zéro) (tableau 1).

	0	1	2	3	4	5	6	7
0	63	-34	**9**	**10**	7	-13	12	7
1	-31	23	**-14**	**-13**	3	4	6	1
2	**15**	**14**	**3**	**-12**	5	-7	3	9
3	**-9**	**-7**	**-14**	**8**	4	-2	3	**45**
4	-5	9	-1	7	4	-6	-2	2
5	3	0	-3	2	2	-2	0	4
6	2	-3	6	4	3	6	3	6
7	5	11	5	**61**	0	3	-4	**49**

Figure 3.8 : Exemple 2 décomposition à trois résolutions d'une matrice de taille 8x8.

Exemple 2

SPIHT :

Out Bit : +1 0 0 0 1 **0 0 0 0** 1 **0 0 0 0** 1 **0 0 0 0** 1 0 0 0 1 0 0 0 +1 1 0 0 0 1 0 0 0 +1 1 0 0 0 1 0 0 0 +1

Lsp : 1 0 1 1

Remplacement de 4 bit par 1seul dans notre méthode

MSPIHT : Out Bit +1 0 0 0 10 **0** 10 **0** 10 **0** 1 0 0 0 1 0 0 0 +1 1 0 0 0 1 0 0 0 +1 1 0 0 0 1 0 0 0 +1

Lsp : 1 0 1 1

Le codage pour les quatre fils directs insignifiants de coefficient (0,1) {9,10, -14, -13}, (1,0) {15,14, -9, -7} et (1,1) {3, -12, -14,8} sont mees à zéro. au lieu de quatre bits pour le SPIHT.la conséquence de cette réduction permet une réduction notable du nombre de bits.

Le nombre de symboles pour calculer les coefficients de la liste Outbit à la fois pour l'exemple 1 (voir figure 3.8) et l'exemple 2 (voir Figure3.8) est donner par :

	Example1		Examlpe2	
	SPIHT	MSPIHT	SPIHT	MSPIHT
Nombre de Symboles	29	28	50	44

On constate que, sans l'aide de toute autre forme de codage entropique supplémentaire :

- Pour l'exemple 1 on a 29 bits pour le SPIHT et 28 pour le codeur MSPIHT.
- Pour l'exemple 2 on 50 bits pour le SPIHT et 44 pour le codeur MSPIHT.

Un codage progressif consiste en la possibilité de reconstruire l'image à n'importe quel instant. Pour être progressif, l'algorithme MSPIHT code chaque symbole obtenu, après chaque étape de test de signifiance et chaque étape de quantification et raffinement, par un codage entropique adaptatif. Le nombre de bits obtenus après l'étape de codage sera ensuite comparé avec le taux de compression désiré. Si le taux de compression désiré est atteint, le codage est terminé et la trame de bits codés est envoyée au récepteur. Sinon, on continue l'algorithme par le

test des coefficients d'ondelettes restants (non testés). Si tous les coefficients sont testés, on divise le seuil par deux et on continue l'opération de test de signifiance.

Le décodeur peut reconstruire l'image à n'importe quel instant. Il suit le même principe du codeur pour le balayage des coefficients. Il débute par une matrice vide (toutes les valeurs sont nulles) et commence à remplir les emplacements indiqués par la liste outbit on prenant en considération la valeur *DO(i ,j)* et les valeurs quantifiées indiquées par la liste LSP. Par l'application de la transformée en ondelettes inverse sur la matrice obtenue, on peut avoir une image reconstruire à n'importe quel instant.

Etape	Pixel ou ensemble testés	Bit de sortie outbit	Action	Listes de control
(1)				LIS={(0,1)A, (1,0)A, (1,1)A} LIP={(0,0), (0,1) (1,0), (1,1)} LSP=ϕ
(2)	(0,0)	1+	(0,0) à LSP	LIP={(0,1),(1,0), (1,1)} LSP={(0,0)}
	(0,1)	1–	(0,1) à LSP	LIP = {(1,0),(1,1)} LSP = {(0,0),(0,1)}
	(1,0)	0	Aucun	
	(1,1)	0	Aucun	
(3)	D(0,1)	1	Teste tous les descendants	LIS={(0,1)A, (1,0)A, (1,1)A}
	DO(0,1)	1	Teste tous les descendants directs	
	(0,2)	1+	(0,2) à LSP	LSP = {(0,0),(0,1),(0,2) }
	(0,3)	0	(0,3) à LIP	LIP={(0,1),(1,0), (1,1),(0,2),(0,3)}
	(1,2)	0	(1,2) à LIP	LIP={(0,1),(1,0), (1,1),(0,2),(0,3),(1,2)}
	(1,3)	0	(1,3) à LIP	LIP={(0,1),(1,0), (1,1),(0,2),(0,3),(1,2),(1,3)}
(4)			Changes de type	LIS={(1,0)A, (1,1)A, (0,1)B}
(5)	D(1,0)	1	Teste tous les descendants	LIS={(1,0)A, (1,1)A, (0,1)B}
(6)	DO(1,0)	0	Teste tous les descendants directs	
	(2,0)	0	(2,0) à LIP	LIP={(0,1),(1,0), (1,1),(0,2),(0,3),(1,2),(1,3),(2,0)}
	(2,1)		(2,1) à LIP	LIP={(0,1),(1,0),(1,1),(0,2),(0,3),(1,2),(1,3),(2,0),(2,1)}
	(3,0)		(3,0) à LIP	LIP={(0,1),(1,0),(1,1),(0,2),(0,3),(1,2),(1,3),(2,0),(2,1),(3,0)}
	(3,1)		(3,1) à LIP	LIP={(0,1),(1,0),(1,1),(0,2), (0,3),(1,2),(1,3),(2,0),(2,1),(3,0),(3,1)}
			Changement de type	LIS={(1,1)A,(0,1)B,(1,0)B}
(7)	D(1,1)	0	Teste tous les descendants	LIS={(1,1)A,(0,1)B,(1,0)B}
(8)	L(0,1)	0	Aucun	LIS={(1,1)A,(0,1)B,(1,0)B}
(9)	L(1,0)	1	Ajout nouvelles listes	LIS = {(1,1)A,(0,1)B,(2,0)A,(2,1)A,(3,0)A,(3,1)A}
(10)	D(2,0)	0	Aucun	LIS = {(1,1)A,(0,1)B,(2,0)A,(2,1)A,(3,0)A,(3,1)A}
(11)	D(2,1)	1	Teste tous les descendants directs	LIS = {(1,1)A,(0,1)B,(2,0)A,(2,1)A,(3,0)A,(3,1)A}
	(4,2)	0	(4,2) à LIP	LIP={(0,1),(1,0),(1,1),(0,2),(0,3),(1,2),(1,3),(2,0),(2,1),(3,0),

				(3,1),(4,2)}
	(4,3)	1+	(2,7) à LSP	LSP = {(0,0),(0,1),(0,2) ,(4,3)}
	(5,2)	0	(5,2) à LIP	LIP={(0,1),(1,0),(1,1),(0,2),(0,3),(1,2),(1,3),(2,0),(2,1),(3,0),(3,1),(4,2),(5,2)}
	(5,3)	0	(5,3) à LIP	LIP={(0,1),(1,0),(1,1),(0,2),(0,3),(1,2),(1,3),(2,0),(2,1),(3,0),(3,1),(4,2),(5,2),(5,3)}
			enlever (2,1)	LIS = {(1,1)A,(0,1)B,(2,0)A,(3,0)A,(3,1)A}
(12)	D(3,0)	0	Aucun	LIS = {(1,1)A,(0,1)B,(2,0)A,(3,0)A,(3,1)A}
(13)	D(3,1)	0	Aucun	LIS = {(1,1)A,(0,1)B,(2,0)A,(3,0)A,(3,1)A}
				LIP={(0,1),(1,0),(1,1),(0,2),(0,3),(1,2),(1,3),(2,0),(2,1),(3,0),(3,1),(4,2),(5,2),(5,3)} LIS = {(1,1)A,(0,1)B,(2,0)A,(3,0)A,(3,1)A} LSP = {(0,0),(0,1),(0,2) ,(4,3)}

Table 3.1: Différentes étapes de l'algorithme MSPIHT appliqué sur la matrice de test de la figure 3.6 pour une itération (T_0=32)

5. Conclusion

Dans ce chapitre, nous avons proposé un algorithme de compression d'images le M-SPIHT, basé sur le même principe que l'algorithme SPIHT. Cet algorithme est basé sur l'utilisation d'un seul symbole au lieu de quatre symboles non significatifs utilisés par l'algorithme SPIHT pour minimiser le nombre de symboles redondants d'une part, et le regroupement binaire de l'information avant le codage entropique pour une meilleure optimisation de la quantité d'information, d'autre part

Chapitre 4 : Résultats et Discussions

Résumé

Dans ce chapitre, nous allons défini le protocole de tests que nous avons utilisé pour la validation de notre méthode.

Nous avons basé nos tests sur cinq images ; trois niveaux gris de taille 512*512 codées sur 8 bits et deux images couleurs de taille 248*248et 2000*2000 codées 24 bits sur .La métrique utilisée concerne : le PSNR, le taux de compression, le temps de calcul.

Nous avons aussi cherché à montrer l'intérêt du bon Choix de l'ondelette dans la transformation pour cela nous avons comparé : les ondelettes db4, bior3.3, Haar.

Nous avons aussi analysé l'intérêt du codage entropique (arithmétique) dans notre codage, enfin nous terminons notre étude en comparons notre méthodes avec quelques codeurs notamment JPG2000, EZW, SPIHT.

1. Introduction

Après l'étude de l'algorithme SPIHT et la proposition d'une optimisation de ce dernier, nommée MSPIHT (cf. chapitre 3); Nous effectuons dans ce chapitre une étude détaillée des résultats obtenus par les deux algorithmes (SPIHT et MSPIHT) appliqués sur des images de test fréquemment utilisées dans la littérature. Une étude comparative avec les algorithmes de compression d'images actuels comme la norme JPEG 2000[38] et l'algorithme EZW est aussi effectuée.

2. Paramètres de validation

En fonction de l'application recherchée, l'algorithme de compression doit pouvoir vérifier un certain nombre de critères de qualité, entre autres, on peut citer : le taux de compression, le rapport de dégradation et la vitesse de compression et de la décompression (temps de calcule). Autres paramètres très utilisés dans le domaine de la compression, la quantité d'information et l'entropie. Ces paramètres permettent de caractériser l'image originale et d'estimer l'efficacité du codage (d'une manière globale) avant le calcul du taux de compression.

Nous avons rappeler très brièvement ces différente critères que nous avons utilisé.

3. Quantité d'information et entropie

Dans la théorie de l'information, chaque point (pixel) d'une image est considéré comme une variable aléatoire. Soit *x* un pixel d'une image en niveau de gris. Ce pixel est une variable aléatoire dont les valeurs sont des entiers de l'intervalle [0, 255]. Soit $p(n_i)$ la probabilité pour que le niveau de gris en *x* soit n_i. La quantité d'information Q d'une variable aléatoire *x* est donnée par [39] :

$$Q_x(n_i) = -\log(p(n_i)) \qquad (4.1)$$

On appelle *entropie* de la variable aléatoire x l'espérance mathématique de la quantité d'information des observations de x :

$$H(x) = E\{Q_x\} = E\{-\log(p(x))\} \qquad (4.2)$$

Considérons une source d'information S qui émet des messages constitués de symboles de l'alphabet : $A = \{\lambda_{1,\dots\dots}\}$ et de la loi de probabilité : $P(A)=\{p_1=P(\lambda_1), \dots, p_n=P(\lambda_n)\}$, Son entropie est donnée par la formule [31] :

$$H(S) = -\sum_{i=1}^{n} p_i \log(p_i) \qquad (4.3)$$

Shannon a montré, d'une part, que l'entropie d'une source correspond à la longueur moyenne minimale de description de ses messages, et d'autre part qu'en choisissant convenablement son code, il était possible d'approcher cette valeur minimale d'aussi près que l'on veut.

4. Taux de compression

Le taux de compression est défini comme le rapport entre le nombre total de bits nécessaires pour représenter l'information originale et le nombre total de bits du fichier binaire à stocker qui résulte de la méthode de compression :

$$RC\ (\%) = \frac{nombre\ \ de\ \ bits\ \ codés}{nombre\ \ de\ \ bits\ \ de\ \ l'image\ \ originale} \times 100 \qquad (4.4)$$

Dans la pratique, on utilise plutôt le débit pour mesurer le pouvoir de compactage d'une méthode. Le débit est exprimé en bits par pixel :

$$RC\ (bpp) = \frac{nombre\ \ de\ \ bits\ \ codés}{nombre\ \ de\ \ bits\ \ de\ \ l'image\ \ originale} \qquad (4.5)$$

4.1 Distorsion

L'information perdue entre le signal original et le signal décodé en fin de chaîne, s'appelle *distorsion*. La mesure de distorsion la plus couramment utilisée est l'Erreur Quadratique Moyenne (EQM). Ce critère se calcule comme la moyenne des carrées des écarts entre les pixels de l'image reconstruite et les pixels correspondants de l'image originale [40] :

$$EQM = \frac{1}{n \times m} \sum_{i=1}^{n} \sum_{j=1}^{m} (f(i,j) - \hat{f}(i,j))^2 \qquad (4.6)$$

$n \times m$: La taille de l'image.

f , $\hat{f}$:Les valeurs des intensités de l'image originale et de l'image reconstruite respectivement.

L'autre critère d'évaluation, directement déduit de l'EQM est le rapport *signal/bruit* (SNR) donné par [7] :

$$SNR = 10 \log_{10} \left(\frac{1}{EQM}\right) \quad (4.7)$$

Est la variance du signal (l'image). La grandeur SNR ce mesure en décibel (dB).

Un autre critère très utilisé est le rapport signal/bruit de crête ou PSNR (Peak Signal to Noise Ratio) défini par :

$$PSNR\ (db) = 10 \log_{10} \left(\frac{\max(f(i,j))}{EQM}\right) \quad (4.8)$$

Où *max* (f_i) est l'amplitude maximale de l'image originale (soit 255 pour une image codée sur 8 bits).

4.2 Temps de calcul

La contrainte du temps est un facteur essentiel dans l'évaluation des performances de toute méthode de compression, elle revient à calculer le temps pris par la compression et la décompression des images. Cette contrainte est plus au moins imposée selon l'application visée par la compression (transmission ou archivage). En effet, il serait dommage, dans une application de transmission, que le temps gagné par une réduction de la taille des données à transmettre soit inférieur au temps passé à la compression décompression [41]. Cette qualité sera cependant moins cruciale dans des applications visant l'archivage de données.

5. Images de test

Pour notre application, nous avons utilisé les images suivantes.

- Lena codée sur 8 bits, niveaux gris de taille 512x512 pixels (fig4.1).
- Texture codée sur 8 bits, niveaux gris de taille 512x512 pixels (fig4.2).
- image IRM-genou codée sur 8 bits, niveaux gris de taille 512x512 pixels (fig4.3).
- image earth codée sur 24 bits, couleur de taille 248 x248 pixels (fig4.4).
- image earth codée sur 24 bits, couleur de taille 2000 x2000 pixels (fig4.5).

Figure 4.1 : image originale Lena 512 x512

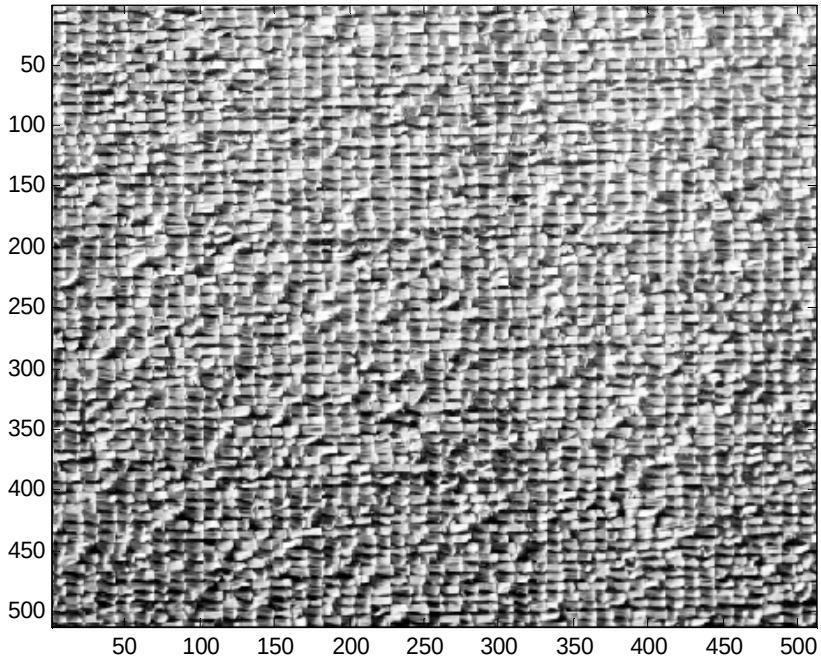


Figure 4.2 : image originale Texture 512 x512

Figure 4.3 : image originale genou 512 x512

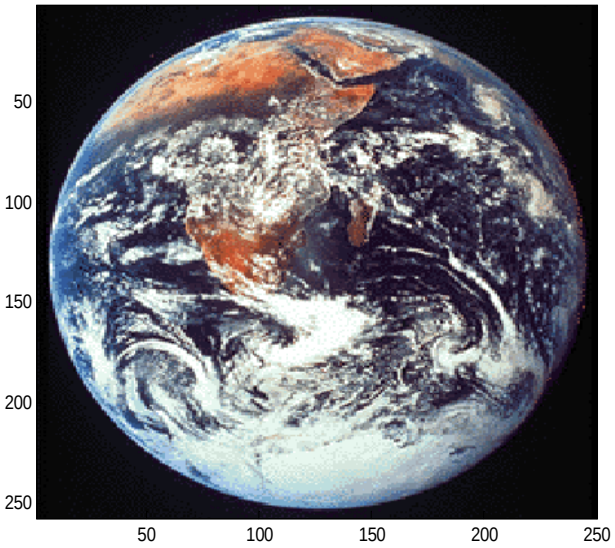


Figure 4.4 : image originale earth 248 x 248

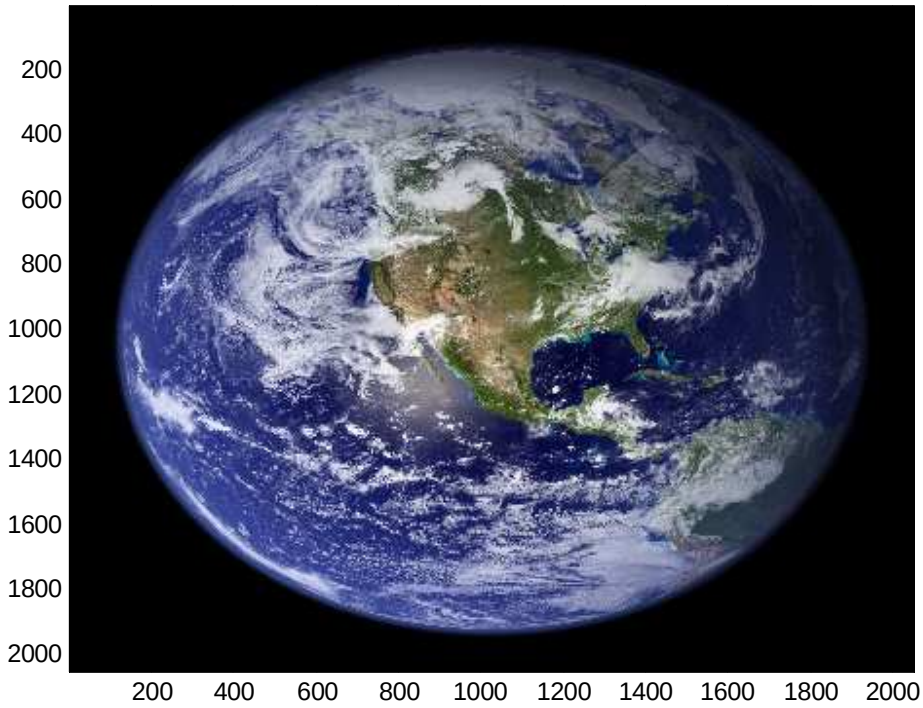


Figure 4.5 : image originale earth 2000 x 2000

6. Choix de l'ondelette

Comme nous l'avons mentionné dans la chapitre 2, nous allons revenir sur le protocole de chois de notre ondelettes et des principales propriétés que nous somme besoin vérifiées.

Elle peut être choisie de diverses manières en fonction des propriétés désirées sur la base de décomposition. Parmi les principales propriétés on retiendra :

1. la reconstruction parfaite : la projection de *x* sur la famille W doit être inversible.

2. l'orthogonalité : les fonctions de base W sont orthogonales.

3. la phase linéaire : les fonctions d'ondelette mère sont symétriques ou antisymétriques.

4. la régularité: la projection sur une fonction d'ondelette mère de tout polynôme d'ordre inférieur ou égal à P orienté est nulle.

En traitement d'image, la propriété de reconstruction parfaite est essentielle pour ne pas dégrader le signal lors de son traitement et former une bijection entre le domaine transformé et le

domaine spatial. La propriété de phase linéaire évite les distorsions de phase très désagréables visuellement (délocalisation des contours).

7. Niveaux de décomposition

Un autre critère très important dans la compression est le niveau de décomposition de la transformée par ondelettes dans notre travail on a choisi seulement trois résolution par ce on peut montrer l'efficacité de notre algorithme.

8. Algorithme proposé MSPIHT

Nous rappelons le bloc diagramme de l'algorithme que nous avons proposé el utilise pour la phase de validation (cf 4.6). L'ensemble du processus à être réaliser sur un Micro ordinateur intel core2duo 1.8, RAM = 2Go, DD=180GO en utilisant le logiciel Matlab 7.3.

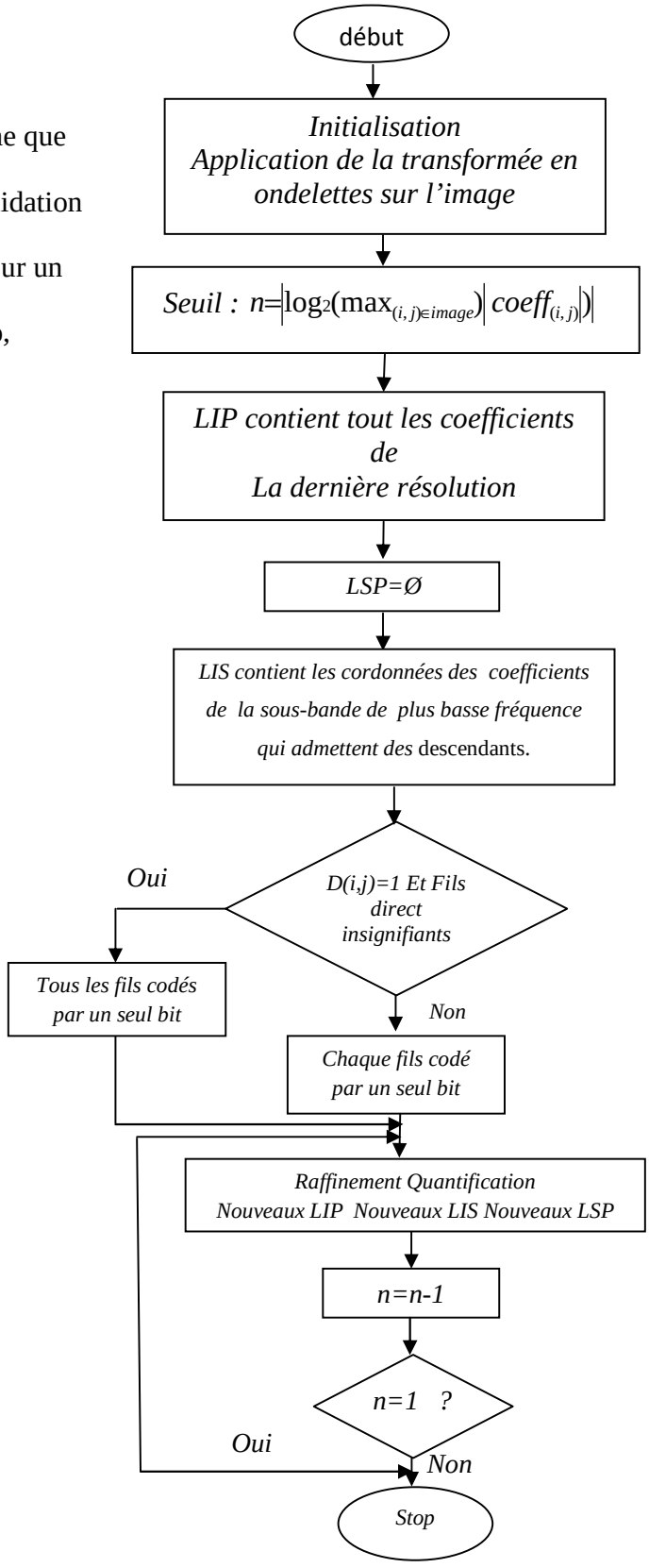


Figure 4.6 Structure l'algorithme MSPIHT

9. Résultat et discussion

Dans ces tableaux, on commence par comparer notre méthode MSPIHT avec le codeur SPIHT par différente seuil dans chaque cas d'ondelette utilisé pour la transformation.

Puis on regarde l'influence du codeur arithmétique pour différente décomposition (2bit, 4bit et 8bit).

Ensuite on fait varier le choix de l'ondelettes (bior3.3, db4 et haar) et aussi l'utilisation de la transformée bandelettes.

En termine ensuite par une discussion des résultats trouvés.

9.1 image Texture

	MSPIHT			**SPIHT**		
Seuil	OUTBIT	LSP	TOTAL INFORMATION	OUTBIT	LSP	TOTAL INFORMATION
seuil /8	118177	22667	140844	129200	22667	151867
seuil /16	244268	61811	306079	257227	61811	319038
seuil /32	364700	112087	476787	375328	112087	487415
seuil /64	464076	162466	626542	470293	162466	632759
seuil /128	531029	204123	735152	534263	204123	738386(bits)

Table 4.1: OUTBIT et LSP de codeurs MSPIHT et SPIHT algorithmes appliqués a l'image Texture 512×512 seuil =252 et ondelette db4.

	MSPIHT + CODAGE ARITHMETIQUE			**SPIHT +CODAGE ARITHMETIQUE**		
Seuil	**DECOMPOSITION: 2 BITS**	**DECOMPOSITION : 4BITS**	**DECOMPOSITION: 8 BITS**	**DECOMPOSITION: 2 BITS**	**DECOMPOSITION: 4 BITS**	**DECOMPOSITION: 8 BITS**
seuil/8	118376	117616	119488	117952	117672	119704
seuil/16	274936	273368	275424	279840	278784	281032
seuil/32	440584	438088	438984	447560	445224	446960
seuil/64	580016	575048	572560	585184	581184	581112
seuil/128	677856	662576	657120	678360	671976	668888(bits)

Table 4.2: Résultat du codage entropique appliqué sur MSPIHT et SPIHT pour l'image Texture 512×512 ave un seuil =252 et avec ondelette db4.

	MSPIHT			**SPIHT**		
Seuil	OUTBIT	LSP	TOTAL INFORMATION	OUTBIT	LSP	TOTAL INFORMATION
seuil/8	54620	10860	65480	55478	10860	66338
seuil/16	118868	28029	146897	127760	28029	155789
seuil/32	224487	59358	283845	235006	59358	294364
seuil/64	325941	99949	425890	334161	99949	434110
seuil/128	417729	141949	559678	422515	141949	564464(bits)

Table 4.3: OUTBIT et LSP de codeurs MSPIHT et SPIHT algorithmes appliqués a l'image Texture 512×512 seuil=252 et ondelette bior3.3.

	MSPIHT +CODAGE ARITHMETIQUE			**SPIHT + CODAGE ARITHMETIQUE**		
Seuil	**DECOMPOSITION: 2 BITS**	**DECOMPOSITION : 4BITS**	**DECOMPOSITION: 8 BITS**	**DECOMPOSITION: 2 BITS**	**DECOMPOSITION: 4 BITS**	**DECOMPOSITION: 8 BITS**
Seuil/8	53520	53456	55144	45272	45464	46768
Seuil/16	128632	128120	131400	125280	124560	127336
Seuil/32	257200	256304	259368	256016	254112	256096
Seuil/64	393032	390624	392824	393128	389944	392080
Seuil/128	517472	514296	513760	517448	513992	515232 (bits)

Table 4.4: Résultat du codage arithmétique appliqué sur MSPIHT et SPIHT pour l'image Texture 512×512 ave un seuil =252 et avec ondelette bior 3.3.

	MSPIHT			**SPIHT**		
Seuil	**OUTBIT**	**LSP**	**TOTAL INFORMATION**	**OUTBIT**	**LSP**	**TOTAL INFORMATION**
Seuil/8	172454	33760	206214	185626	33760	219386
Seuil/16	299418	77309	376727	312444	77309	389753
Seuil/32	412762	129512	542274	422483	129512	551995
Seuil/64	496643	178565	675208	502329	178565	680894
Seuil/128	543793	213329	757122	546805	213329	760134(bits)

Table 4.5: OUTBIT et LSP de codeurs MSPIHT et SPIHT algorithmes appliqués a l'image Texture 512×512 seuil=252 et ondelette haar.

	MSPIHT +CODAGE ARITHMETIQUE			**SPIHT + CODAGE ARITHMETIQUE**		
Seuil	**DECOMPOSITION: 2 BITS**	**DECOMPOSITION : 4BITS**	**DECOMPOSITION: 8 BITS**	**DECOMPOSITION: 2 BITS**	**DECOMPOSITION: 4 BITS**	**DECOMPOSITION: 8 BITS**
Seuil/8	175776	174568	175520	178112	177920	179960
Seuil/16	340888	338896	340264	347248	346312	348416
Seuil/32	497048	493984	494096	504872	502888	504096
Seuil/64	611088	6092241	605720	620864	617544	616896
Seuil/128	684176	666344	660856	695648	677784	675384(bits)

Table 4.6: Résultat du codage arithmétique appliqué sur MSPIHT et SPIHT pour l'image Texture 512×512 avec un seuil =252 et avec ondelette Haar.

Les résultats obtenus montrent que le nombre de bits dans la liste outbit de MSPIHT est inferieur au nombre de bits de la liste outbit utilisés dans l'algorithme original SPIHT quelque soit la valeur du seuil et de l'ondelette utilisée pour la transformation de l'image. Celà nous permet de diminuer le nombre total de bits à coder (cf. tables 4.1 - 4.3- 4. 5). En effet dans notre approche MSPIHT, nous trouvons dans la liste outbit des symboles qui sont représentés par quatre symboles dans le codeur SPIHT (cf. chapitre 3, exemple1, exemple 2). On remarque que le nombre total de bits dans le codeur MSPIHT est inférieur au nombre de bit dans le SPIHT (cf. tables tables 4.1 - 4.3- 4. 5). On montre aussi que l'utilisation du codage arithmétique pour les codeurs SPIHT et MSPIHT diminue de façon notable le nombre de bits à coder dans tous les cas

(cf. tables 4.1 - 4.3- 4. 5) quelque soit l'ondelette, le seuil, et les décompositions utilisés dans le codage arithmétique (cf. tables 4.2 - 4.4- 4. 6). On peut aussi affirmer après avoir testé un grand nombre d'images texturées que le codage arithmétique appliqué au codeur MSPIHT est toujours le plus performant par rapport au SPIHT.

Enfin, on montre aussi que le nombre total de bits à coder et inferieur pour l'ondelette bior 3.3 par rapport à db4 et Haar et ce pour tous les seuils considérés.

9.2 Image Lena

	MSPIHT			**SPIHT**		
Seuil	OUTBIT	LSP	TOTAL INFORMATION	OUTBIT	LSP	TOTAL INFORMATION
seuil/8	33947	4349	38296	34130	4349	38479
seuil/16	42775	6354	49129	44198	6354	50552
seuil/32	68595	13125	81720	72050	13125	85175
seuil/64	127901	28553	156454	134703	28553	163256
seuil/128	274611	66139	340750	286716	66139	352855(bits)

Table 4.7: OUTBIT et LSP de codeurs MSPIHT et SPIHT algorithmes appliqués a l'image Lena 512×512 seuil =184 et ondelette db4.

	MSPIHT + CODAGE ARITHMETIQUE			**SPIHT + CODAGE ARITHMETIQUE**		
Seuil	**DECOMPOSITION: 2 BITS**	**DECOMPOSITION : 4BITS**	**DECOMPOSITION: 8 BITS**	**DECOMPOSITION: 2 BITS**	**DECOMPOSITION: 4 BITS**	**DECOMPOSITION: 8 BITS**
seuil/8	12496	9912	11312	8640	9936	9288
seuil/16	30392	28880	31440	27960	28264	29896
seuil/32	67520	66552	70920	68112	66112	69776
seuil/64	142224	140880	144288	142080	141536	144624
seuil/128	311448	309208	310272	316032	314232	316392(bits)

Table 4.8: Résultat du codage arithmétique appliqué sur MSPIHT et SPIHT pour l'image Lena 512×512 avec un seuil =184 et avec ondelette db4.

	MSPIHT			**SPIHT**		
Seuil	OUTBIT	LSP	TOTAL INFORMATION	OUTBIT	LSP	TOTAL INFORMATION
seuil/8	33442	4330	37772	33442	4330	37772
seuil/16	38244	5919	44163	38560	5919	44479
seuil/32	51579	9912	61491	53252	9912	63164
seuil/64	84537	19244	103781	88194	19244	107438
seuil/128	171505	42035	213540	179557	42035	221592(bits)

Table 4.9: OUTBIT et LSP de codeurs MSPIHT et SPIHT algorithmes appliqués a l'image Lena 512×512 seuil =184 et ondelette bior3.3.

	MSPIHT + CODAGE ARITHMETIQUE			**SPIHT +CODAGE ARITHMETIQUE**		
Seuil	**DECOMPOSITION: 2 BITS**	**DECOMPOSITION : 4BITS**	**DECOMPOSITION: 8 BITS**	**DECOMPOSITION: 2 BITS**	**DECOMPOSITION: 4 BITS**	**DECOMPOSITION: 8 BITS**
seuil/8	13112	12952	10592	9584	12712	10224
seuil/16	25096	23120	25568	23272	22120	22976
seuil/32	47088	45920	49736	45064	43928	47224
seuil/64	90696	90248	94568	88624	88616	92040
seuil/128	193680	193088	196192	194352	193728	197216(bits)

Table 4.10: Résultat du codage arithmétique appliqué sur MSPIHT et SPIHT pour l'image Lena 512×512 avec un seuil =184 et avec ondelette bior3.3.

	MSPIHT			**SPIHT**		
Seuil	OUTBIT	LSP	TOTAL INFORMATION	OUTBIT	LSP	TOTAL INFORMATION
seuil/8	35923	4717	40640	36487	4717	41204
seuil/16	54435	8697	63132	56848	8697	65545
seuil/32	95774	19879	115653	100312	19879	120191
seuil/64	177890	43326	221216	186196	43326	229522
seuil/128	335390	90760	426150	347726	90760	438486(bits)

Table 4.11: OUTBIT et LSP de codeurs MSPIHT et SPIHT algorithmes appliqués a l'image Lena 512×512 seuil=184 et ondelette Haar.

	MSPIHT + CODAGE ARITHMETIQUE			**SPIHT +CODAGE ARITHMETIQUE**		
Seuil	**DECOMPOSITION: 2 BITS**	**DECOMPOSITION : 4BITS**	**DECOMPOSITION: 8 BITS**	**DECOMPOSITION: 2 BITS**	**DECOMPOSITION: 4 BITS**	**DECOMPOSITION: 8 BITS**
seuil/8	16712	14200	15320	12536	14032	14552
seuil/16	43672	41696	45112	43504	42008	45456
seuil/32	98752	96760	101168	97904	98040	101344
seuil/64	200184	198968	202256	202896	202016	205544
seuil/128	388672	386704	386832	396520	394216	396408(bits)

Table 4.12: Résultat du codage arithmétique appliqué sur MSPIHT et SPIHT pour l'image Lena 512×512 avec un seuil =184 et avec ondelette Haar.

On constate pour l'application des codeurs SPIHT et MSPIHT avec l'image Lena les mêmes observations que les résultats obtenus avec l'image texturée. Toutefois, on remarque que les résultats obtenus lors, de l'application des codeurs SPIHT et MSPIHT avec ou sans codage arithmétique, que le nombre de bits codés est inferieur à ceux obtenus avec l'image Texture ; celà est du à la nature de l'image qui présente beaucoup plus de textures que l'image Lena. On a alors les détails qui augmentent avec les résolutions. Bien entendu il faut aussi prendre en compte la valeur du seuil qui a aussi son rôle, pour le MSPITH par rapport au SPIHT, dans les performances des codeurs.

9.3 Image IRM-genou

	MSPIHT			**SPIHT**		
Seuil	OUTBIT	LSP	TOTAL INFORMATION	OUTBIT	LSP	TOTAL INFORMATION
seuil/8	30970	1971	32941	31003	1971	32974
seuil/16	34464	3002	37466	34912	3002	37912
seuil/32	50389	6559	56944	52955	6559	59510
euil/64	102070	18657	120727	108090	18657	126747
seuil/128	170137	42596	212733	176102	42596	218698(bits)

Table 4.13: OUTBIT et LSP de codeurs MSPIHT et SPIHT algorithmes appliqués a l'image IRM-genou 512×512 seuil =164 et ondelette db4.

	MSPIHT + CODAGE ARITHMETIQUE			**SPIHT + CODAGE ARITHMETIQUE**		
Seuil	**DECOMPOSITION: 2 BITS**	**DECOMPOSITION : 4BITS**	**DECOMPOSITION: 8 BITS**	**DECOMPOSITION: 2 BITS**	**DECOMPOSITION: 4 BITS**	**DECOMPOSITION: 8 BITS**
seuil/8	6944	7720	8448	6744	7512	8272
seuil/16	14752	15096	17120	18672	14480	16176
seuil/32	37248	37312	39880	36480	36800	36904
seuil/64	96776	97120	101440	97208	97368	101480
seuil/128	184520	184616	190216	187576	187072	192992(bits)

Table 4.14: Résultat du codage entropique appliqué sur MSPIHT et SPIHT pour l'image IRM-genou 512×512 avec un seuil =164 et avec ondelette db4.

	MSPIHT			**SPIHT**		
Seuil	OUTBIT	LSP	TOTAL INFORMATION	OUTBIT	LSP	TOTAL INFORMATION
seuil/8	30863	1927	32790	30870	1927	32797
seuil/16	33440	2896	36336	33574	2896	36470
seuil/32	43916	5915	49831	44957	5915	50872
seuil/64	77417	15045	92462	81296	15045	96341
seuil/128	129500	31413	160913	134388	31413	165801(bits)

Table 4.15: OUTBIT et LSP de codeurs MSPIHT et SPIHT algorithmes appliqués a l'image IRM-genou 512×512 seuil =164 et ondelette bior3.3.

	MSPIHT + CODAGE ARITHMETIQUE			**SPIHT +CODAGE ARITHMETIQUE**		
Seuil	**DECOMPOSITION: 2 BITS**	**DECOMPOSITION : 4BITS**	**DECOMPOSITION: 8 BITS**	**DECOMPOSITION: 2 BITS**	**DECOMPOSITION: 4 BITS**	**DECOMPOSITION: 8 BITS**
seuil/8	8272	8280	9648	7872	7952	9000
seuil/16	14272	14536	16184	17272	13584	14832
seuil/32	31032	31240	33168	28648	28856	30496
seuil/64	71704	71880	76192	70680	70368	73840
seuil/128	136392	136104	141632	136288	136192	141280(bits)

Table 4.16: Résultat du codage arithmétique appliqué sur MSPIHT et SPIHT pour l'image IRM-genou 512×512 avec un seuil =164 et avec ondelette bior3.3.

	MSPIHT			**SPIHT**		
Seuil	OUTBIT	LSP	TOTAL INFORMATION	OUTBIT	LSP	TOTAL INFORMATION
seuil/8	32676	2360	35036	32922	2360	35282
seuil/16	44938	5442	50380	46401	5442	51843
seuil/32	76505	13721	90226	80434	13721	94155
seuil/64	133166	31259	164425	139178	31259	170437
seuil/128	194781	56413	251194	199720	56413	256133(bits)

Table 4.17: OUTBIT et LSP de codeurs MSPIHT et SPIHT algorithmes appliqués a l'image IRM-genou 512×512 seuil =164 et ondelette Haar.

	MSPIHT + CODAGE ARITHMETIQUE			**SPIHT +CODAGE ARITHMETIQUE**		
Seuil	**DECOMPOSITION: 2 BITS**	**DECOMPOSITION : 4BITS**	**DECOMPOSITION: 8 BITS**	**DECOMPOSITION: 2 BITS**	**DECOMPOSITION: 4 BITS**	**DECOMPOSITION: 8 BITS**
seuil/8	15736	10774	11720	15136	10184	11336
seuil/16	27616	27848	31008	27544	27456	30512
seuil/32	67800	68184	71608	68768	68560	71912
seuil/64	137744	137416	142736	140760	140688	145464
seuil/128	219360	218872	223880	224216	223624	229192(bits)

Table 4.18: Résultat du codage arithmétique appliqué sur MSPIHT et SPIHT pour l'image IRM-genou 512×512 avec un seuil =164 et avec ondelette Haar.

Contrairement aux précédentes images, on a le nombre de bits qui est plus faible, on a toujours notre méthode qui est plus efficace. Pour le codage arithmétiques il est nécessaire de travailler avec des valeurs de seuil faible afin d'avoir des performances plus élevées pour notre méthode.

9.4 Image earth 248x248 et image earth 2000x2000 couleurs

	MSPIHT			**SPIHT**		
seuil	OUTBIT	LSP	TOTAL INFORMATION	OUTBIT	LSP	TOTAL INFORMATION
seuil/8	30452	3977	19029	31853	3977	35830
seuil/16	54445	9600	64045	56833	9600	66433
seuil/32	101663	22275	123938	106942	22275	129217
seuil/64	193509	48044	241553	201157	48044	249158(bits)

Table 4.19: OUTBIT et LSP de codeurs MSPIHT et SPIHT algorithmes appliqués a l'image earth couleur 248×248 seuil=240 et ondelette db4.

	MSPIHT + CODAGE ARITHMETIQUE			**SPIHT + CODAGE ARITHMETIQUE**		
seuil	**DECOMPOSITION: 2 BITS**	**DECOMPOSITION : 4BITS**	**DECOMPOSITION: 8 BITS**	**DECOMPOSITION: 2 BITS**	**DECOMPOSITION: 4 BITS**	**DECOMPOSITION: 8 BITS**
seuil/8	13856	13776	15440	12624	12664	13304
seuil/16	42016	41360	43672	36480	36272	37608
seuil/32	92808	92392	97408	92936	93960	97960
seuil/64	184528	182888	186336	198712	185248	190040(bits)

Table 4.20: Résultat du codage arithmétique appliqué sur MSPIHT et SPIHT pour l'image earth couleur 248×248avec un seuil =240 et avec ondelette db4.

	MSPIHT			**SPIHT**		
seuil	OUTBIT	LSP	TOTAL INFORMATION	OUTBIT	LSP	TOTAL INFORMATION
seuil/8	26450	3705	30155	26692	3705	30397
seuil/16	41782	7620	49402	43306	7620	50926
seuil/32	80000	17577	97577	84072	17577	101649
seuil/64	164353	41724	206077	170288	41724	212006(bits)

Table 4.21: OUTBIT et LSP de codeurs MSPIHT et SPIHT algorithmes appliqués a l'image earth couleur 248×248 seuil=240 et ondelette bior3.3.

	MSPIHT+ CODAGE ARITHMETIQUE			**SPIHT +CODAGE ARITHMETIQUE**		
seuil	**DECOMPOSITION : 2 BITS**	**DECOMPOSITION :4BITS**	**DECOMPOSITION:8 BIT**	**DECOMPOSITION:2 BITS**	**DECOMPOSITION:4 BITS**	**DECOMPOSITION:8 BITS**
seuil/8	10704	11208	11968	9352	9784	10392
seuil/16	31088	30552	32200	29496	29144	31096
seuil/32	73880	73668	77088	72952	72840	75760
seuil/64	154992	154208	158640	156256	155768	160384(bits)

Table 4.22: Résultat du codage arithmétique appliqué sur MSPIHT et SPIHT pour l'image earth couleur 248×248 avec un seuil =240 et avec ondelette bior3.3.

	MSPIHT			**SPIHT**		
seuil	OUTBIT	LSP	TOTAL INFORMATION	OUTBIT	LSP	TOTAL INFORMATION
seuil/8	36852	5067	41919	38717	5067	43784
seuil/16	63375	12119	75494	65975	12119	78094
seuil/32	111383	25588	136971	116902	25588	142490
seuil/64	199790	51186	250976	207218	51186	258404(bits)

Table 4.23: OUTBIT et LSP de codeurs MSPIHT et SPIHT algorithmes appliqués a l'image earth couleur 248×248 seuil =240 et ondelette haar.

	MSPIHT + CODAGE ARITHMETIQUE			**SPIHT +CODAGE ARITHMETIQUE**		
seuil	**DECOMPOSITION: 2 BITS**	**DECOMPOSITION : 4BITS**	**DECOMPOSITION: 8 BITS**	**DECOMPOSITION: 2 BITS**	**DECOMPOSITION: 4 BITS**	**DECOMPOSITION: 8 BITS**
seuil/8	20336	19896	21960	20032	20024	21712
seuil/16	51256	50248	51624	50472	50720	52168
seuil/32	102696	101776	107584	102720	103232	108952
seuil/64	191400	189952	193704	193408	192392	197920(bits)

Table 4.24: Résultat du codage arithmétique appliqué sur MSPIHT et SPIHT pour l'image earth couleur 248×248 avec un seuil =240 et avec ondelette Haar.

	MSPIHT			**SPIHT**		
seuil	OUTBIT	LSP	TOTAL INFORMATION	OUTBIT	LSP	TOTAL INFORMATION
seuil/64	3491433	716617	4208050	3574759	716617	4291376
seuil/128	4532367	1171922	5704289	4605311	1171922	5777233(bits)

Table 4.25: OUTBIT et LSP de codeurs MSPIHT et SPIHT algorithmes appliqués a l'image earth couleur 2000×2000 seuil =240 et ondelette Haar.

	MSPIHT + CODAGE ARITHMETIQUE			**SPIHT + CODAGE ARITHMETIQUE**		
seuil	**DECOMPOSITION: 2 BITS**	**DECOMPOSITION : 4BITS**	**DECOMPOSITION: 8 BITS**	**DECOMPOSITION: 2 BITS**	**DECOMPOSITION: 4 BITS**	**DECOMPOSITION: 8 BITS**
seuil/64	2243816	2210904	2203368	2287952	2261520	2265000
seuil/128	3196216	3152944	312664	3268744	3239072	3227624(bits)

Table 4.26: Résultat du codage arithmétique appliqué sur MSPIHT et SPIHT pour l'image earth couleur 2000×2000 avec un seuil =240 et avec ondelette Haar.

Nous avons affiche les résultats directement en somme des composantes y,c et b .

Le maximum de composant est réalise sur y, car sur les composants c et b il ya très peut d'information ceci est d'autant plus vrai, lorsque l'on utilise le codage arithmétique, on constate alors que les composantes c et b n'interviennent très peu.

Si on utilise des images couleur de grandes taille (2000x2000),(tab 4.25 ,4.26), on voit alors l'intérêt de notre méthode qui permet d'obtenir de tres bon taux. Ce qui nous amène à penser et à affirmer l'intérêt de la compression sur les grandes images satellitaires.

9.5 Influence des filtres

Rc(bpp)	0.05	0.1	0.2	0.5	0.75	1.0	1.5
Psnr(db)	25.73	27.05	31.25	34.23	37.16	39.26	42.65

Table 4.27: PSNR et RC de codeur MSPIHT et codage arithmétique décomposition 4bit appliqué sur l'image Lena 512×2512 seuil =184 et ondelette Haar.

Rc(bpp)	0.05	0.1	0.2	0.5	0.75	1.0	1.5
Psnr(db)	28 .2	30.80	33.01	37.2	39.9	41.35	44.3

Table 4.28: PSNR et RC de codeur MSPIHT et codage arithmétique décomposition 4bit appliqué sur l'image lena 512×2512 seuil =184 et ondelette bior3.3.

Rc(bpp)	0.05	0.1	0.2	0.5	0.75	1.0	1.5
Psnr(db)	26.42	29.05	32.70	36.32	38.45	40.25	43.15

Table 4.29: PSNR et RC de codeur MSPIHT et codage arithmétique décomposition 4bit appliqué sur l'image Lena 512×2512 seuil =184 et ondelette db4.

Rc(bpp)	0.05	0.1	0.2	0.5	0.75	1.0	1.5
Psnr(db)	12.75	14.60	15.80	20.90	22.75	23.60	29.10

Table 4.30: PSNR et RC de codeurs MSPIHT et codage entropique décomposition 4bit appliqué sur l'image Texture 512×2512 seuil =252 et ondelette Haar.

Rc(bpp)	0.05	0.1	0.2	0.5	0.75	1.0	1.5
Psnr(db)	13.75	15.85	17.60	22.25	23.90	25.35	31.30

Table 4.31: PSNR et RC de codeurs MSPIHT et codage entropique décomposition 4bit appliqué sur l'image Texture 512×2512 seuil =252 et ondelette bior3.3.

Rc(bpp)	0.05	0.1	0.2	0.5	0.75	1.0	1.5
Psnr(db)	13.20	15.10	16.90	21.20	23.20	24.20	29.80

Table 4.32 : PSNR et RC de codeurs MSPIHT et codage entropique décomposition 4bit appliqué sur l'image Texture 512×2512 seuil =252 et ondelette db4.

Rc(bpp)	0.05	0.1	0.2	0.5	0.75	1.0	1.5
Psnr(db)	30. 34	31.39	34.23	39.94	43.20	47.65	52.65

Table 4.33: PSNR et RC de codeurs MSPIHT et codage entropique décomposition 4bit appliqué sur l'image IRM-genou 512×2512 seuil =164 et ondelette Haar.

Rc(bpp)	0.05	0.1	0.2	0.5	0.75	1.0	1.5
Psnr(db)	32.00	35.00	37.50	42.30	45.80	51.66	55.30

Table 4.34: PSNR et RC de codeurs MSPIHT et codage entropique décomposition 4bit appliqué sur l'image IRM-genou 512×2512 seuil =164 et ondelette bior3.3.

Rc(bpp)	0.05	0.1	0.2	0.5	0.75	1.0	1.5
Psnr(db)	31.99	34.05	35.70	41.99	44.45	49.26	53.15

Table 4.35: PSNR et RC de codeurs MSPIHT et codage entropique décomposition 4bit appliqué sur l'image IRM-genou 512×2512 seuil =164 et ondelette db4.

Nous avons sur trois images ; Lena, Texture et genou, chercher a analyser l'influence des filtres dans le choix de la transformée pour différente valeur du taux de compression (RC), nous avons calculé le PSNR. On constate que quelque soit l'image c'est le filtre bior3.3 qui donne les meilleurs résultats pour notre codeur.

9.6 Comparaison entre la transformée en ondelettes classiques et transformée en bandelettes pour le codeur MSPIHT

Dans les deux cas, le **MSPIHT** avec la transformée en ondelettes classique et le **MSPIHT** avec la transformée bandelettes, utilisent la transformée en ondelette de Haar.

	MSPIHT avec transformée ondelettes			**MSPIHT avec transformée bandelettes**		
Seuil	OUTBIT	LSP	TOTAL INFORMATION	OUTBIT	LSP	TOTAL INFORMATION
seuil/8	35923	4717	40640	34662	4989	39651
seuil/16	54435	8697	63132	43342	50862	33976
seuil/32	95774	19879	115653	67634	13973	81607
seuil/64	177890	43326	221216	110645	26399	137044
seuil/128	335390	90760	426150	182368	47909	230277

Table 4.36: OUTBIT et LSP de codeurs MSPIHT la avec transformée ondelettes classique et MSPIHT avec la transformée en bandelettes appliqués à l'image Lena 512×512.

	MSPIHT avec transformée ondelettes + codage arithmétique			**MSPIHT avec transformée bandelettes +codage arithmétique**		
seuil	**DECOMPOSITION: 2 BITS**	**DECOMPOSITION : 4BITS**	**DECOMPOSITION: 8 BITS**	**DECOMPOSITION: 2 BITS**	**DECOMPOSITION: 4 BITS**	**DECOMPOSITION: 8 BITS**
seuil/8	16712	14200	15320	17768	15672	17568
seuil/16	43672	41696	45112	33976	32040	34824
seuil/32	98752	96760	101168	67368	66200	70488
seuil/64	200184	198968	202256	123088	121960	126232
seuil/128	388672	386704	386832	210760	210328	213448

Table 4.37: Résultats du codage arithmétique appliqué sur le MSPIHT avec la transformée en ondelettes classique et le MSPIHT avec la transformée en bandelettes pour l'image Lena 512×512.

	MSPIHT avec transformée ondelettes			**MSPIHT avec transformée bandelettes**		
Seuil	OUTBIT	LSP	TOTAL INFORMATION	OUTBIT	LSP	TOTAL INFORMATION
seuil/8	172454	33760	206214	99356	21479	120835
seuil/16	299418	77309	376727	209705	49493	259198
seuil/32	412762	129512	542274	329025	90651	419676
seuil/64	496643	178565	675208	430183	136486	566669
seuil/128	543793	213329	757122	502509	179057	681566

Table 4.38: OUTBIT et LSP de codeurs MSPIHT la avec transformée ondelettes classique et MSPIHT avec la transformée en bandelettes appliqués à l'image Texture 512×512.

	MSPIHT avec transformée ondelettes + codage arithmétique			**MSPIHT avec transformée bandelettes +codage arithmétique**		
seuil	**DECOMPOSITION: 2 BITS**	**DECOMPOSITION : 4BITS**	**DECOMPOSITION: 8 BITS**	**DECOMPOSITION: 2 BITS**	**DECOMPOSITION: 4 BITS**	**DECOMPOSITION: 8 BITS**
seuil/8	175776	174568	175520	94872	94414	96840
seuil/16	340888	338896	340264	223016	220920	223272
seuil/32	497048	493984	494096	377336	374232	375808
seuil/64	611088	6092241	605720	517416	513888	514056
seuil/128	684176	666344	660856	621680	616360	615164

Table 4.39: Résultats du codage arithmétique appliqué sur le MSPIHT avec la transformée en ondelettes classique et le MSPIHT avec la transformée en bandelettes pour l'image Texture 512×512.

	MSPIHT avec transformée ondelettes			**MSPIHT avec transformée bandelettes**		
Seuil	OUTBIT	LSP	TOTAL INFORMATION	OUTBIT	LSP	TOTAL INFORMATION
seuil/8	32676	2360	35036	32531	2602	35133
seuil/16	44938	5442	50380	39115	4513	43628
seuil/32	76505	13721	90226	54444	8509	62953
seuil/64	133166	31259	164425	90171	17591	107762
seuil/128	194781	56413	251194	143984	34558	178542

Table 4.40: OUTBIT et LSP de codeurs MSPIHT la avec transformée ondelettes classique et MSPIHT avec la transformée en bandelettes appliqués à l'image IRM-genou 512×512.

	MSPIHT avec transformée ondelettes + codage arithmétique			**MSPIHT avec transformée bandelettes +codage arithmétique**		
seuil	**DECOMPOSITION: 2 BITS**	**DECOMPOSITION : 4BITS**	**DECOMPOSITION: 8 BITS**	**DECOMPOSITION: 2 BITS**	**DECOMPOSITION: 4 BITS**	**DECOMPOSITION: 8 BITS**
seuil/8	15736	10774	11720	10960	11120	12360
seuil/16	27616	27848	31008	20552	20792	22440
seuil/32	67800	68184	71608	40232	40208	42864
seuil/64	137744	137416	142736	81584	81400	85320
seuil/128	219360	218872	223880	148424	148136	153472

Table 4.41: Résultats du codage arithmétique appliqué sur le MSPIHT avec la transformée en ondelettes classique et le MSPIHT avec la transformée en bandelettes pour l'image IRM-genou 512×512.

On remarque que pour notre codeur, la transformée en bandelettes donne de meilleurs résultats que la transformée en ondelettes classiques, parce que dans la première on a plusieurs étapes supplémentaires qui nous permetent de rejeter les détails significatifs soit dans l'image approximée au dans les détails de la structure pyramidale ; ces détails significatifs ne dégradent plus l'image reconstruite.

9.7 Temps de calcul pour SPIHT et MSPIHT

Seuil	**Codeur MSPIHT**	**Codeur SPIHT**
seuil/16	23.80 sec	30.04 sec
seuil /32	59.50 sec	1.07 min
seuil /64	1.43 min	1.55 min
seuil /128	2.28 min	2.38 min

Table 4.42 : Temps de calcule pour les deux codeurs MSPIHT et SPIHT sur l'image earth couleur 248x248

On remarque que le temps du codeur MSPIHT et un peut plus petit que le temps du codeur SPIHT.

On peut conclu aussi que l'lorsque on augmente la taille de l'image le temps du codeur SPIHT et MSPIHT augmente aussi, alors pour notre cas (l'utilisation de la programmation MATLAB et la fréquence du CPU de mon pc dual core 1,8 ghz) le codeur MSPIHT est bien pour l'archivage que la transmission.

7. Comparaison entre notre codeur avec SPIHT , JPEG2000 et EZW

Image	algorithme De codage	PSNR (db)						
		0.05 bpp	0.1 bpp	0.2 bpp	0.5bpp	0.75 bpp	1bpp	1.5 bpp
texture (512×512)	MSPIHT	**13.75**	**15.85**	**17.60**	**22.25**	**23.90**	**25.35**	**31.30**
	SPIHT	12.60	14.00	15.90	19.70	21.70	23.40	26.90
	JPG2	12.60	14.20	16.30	20.00	21.90	23.80	27.30
	EZW	12.20	13.90	15.50	19.10	21.40	22.40	26.50
Lena (512×512)	MSPIHT	**28.2**	**30.80**	**33.01**	**37.2**	**39.9**	**41.35**	**44.3**
	SPIHT	28.3	30.9	33.90	38.00	39.80	41.20	43.7
	JPG2	28.00	30.7	33.70	37.90	39.70	41.10	43.40
	EZW	23.90	27.8	31.20	35.20	37.20	39.30	41.40
IR-genou (512×512)	MSPIHT	**32.00**	**35.00**	**37.50**	**42.30**	**45.80**	**51.66**	**55.30**
	SPIHT	33.00	35.50	37.70	42.20	45.60	48.50	54.50
	JPG2	32.60	35.30	37.70	42.10	45.40	48.30	52.60
	EZW	27.20	31.50	35.10	39.80	42.10	44.10	47.80

Table 4.43 : Résultats des différents algorithmes appliqués sur trois images (Texture 512×512, lena512×512, IRM-genou).

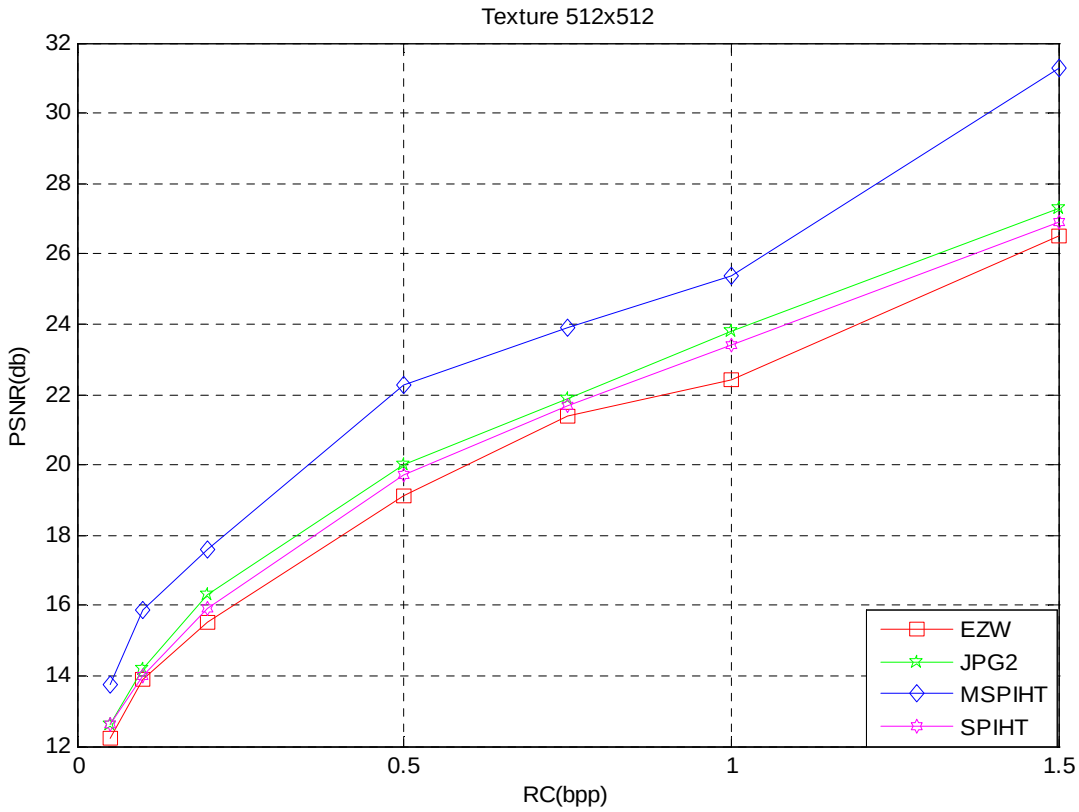


Figure 4.7 : Comparaison des différents algorithmes de compression appliqués sur l'image Texture 512×512.

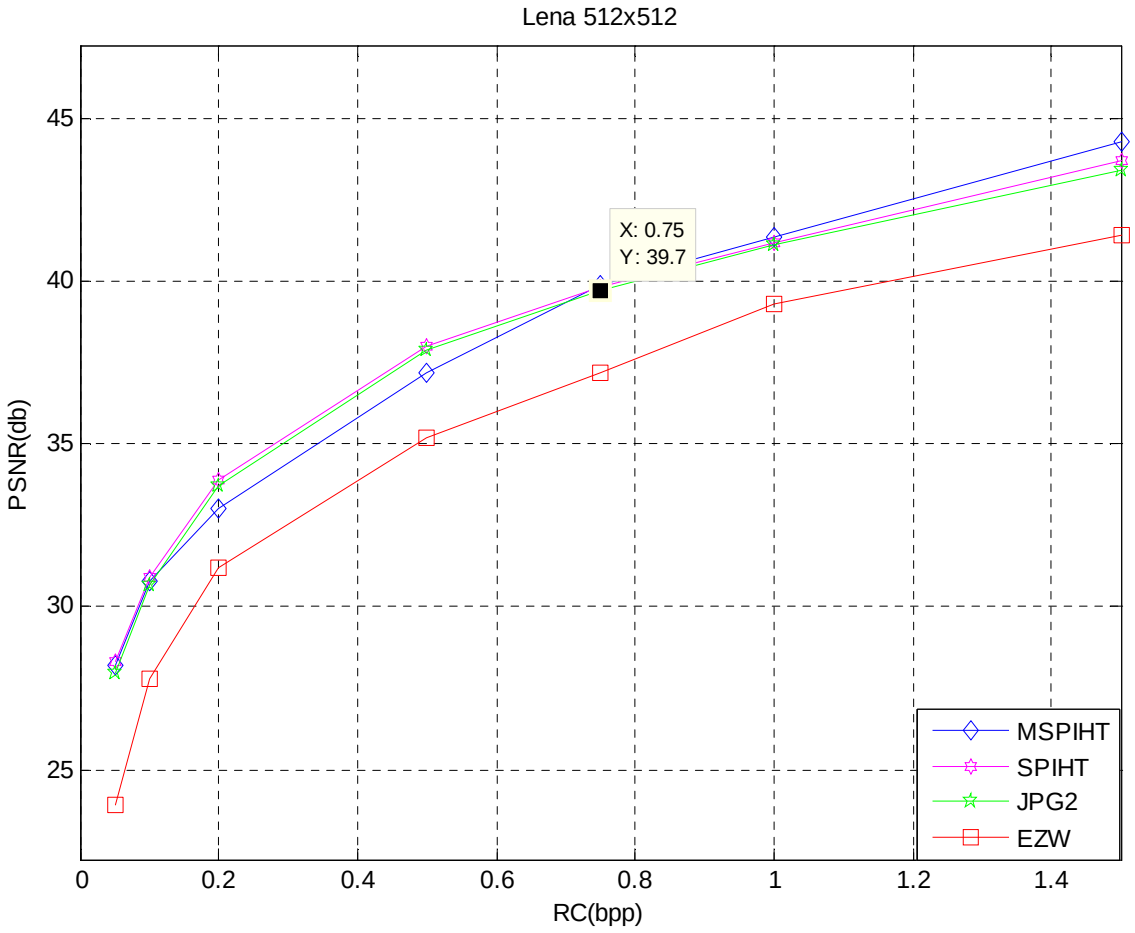


Figure 4.8 : Comparaison des différents algorithmes de compression appliqués sur l'image Lena512×512.

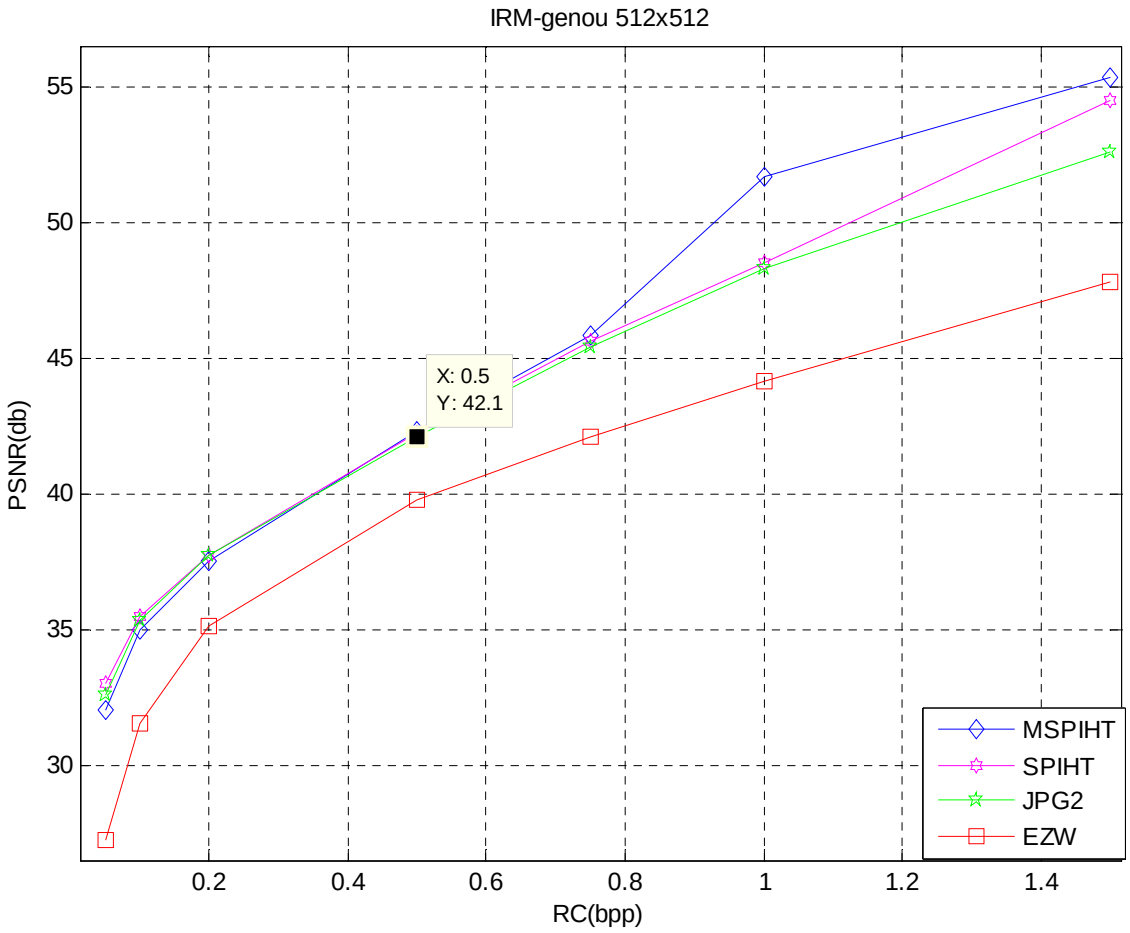


Figure 4.9 : Comparaison des différents algorithmes de compression appliqués sur l'image IRM-genou 512×512.

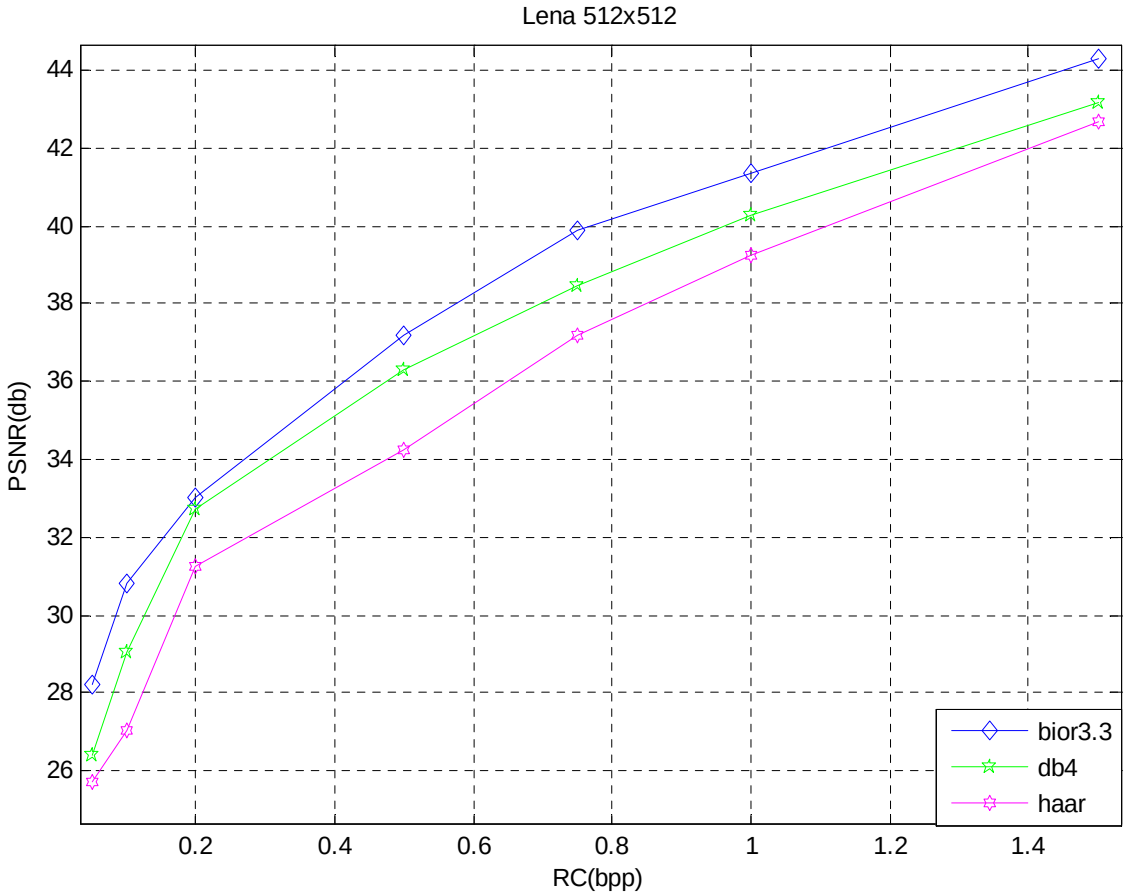


Figure 4.10 : Comparaison des différents filtres appliqués pour la transformation en ondelettes pour l'image Lena 512×512

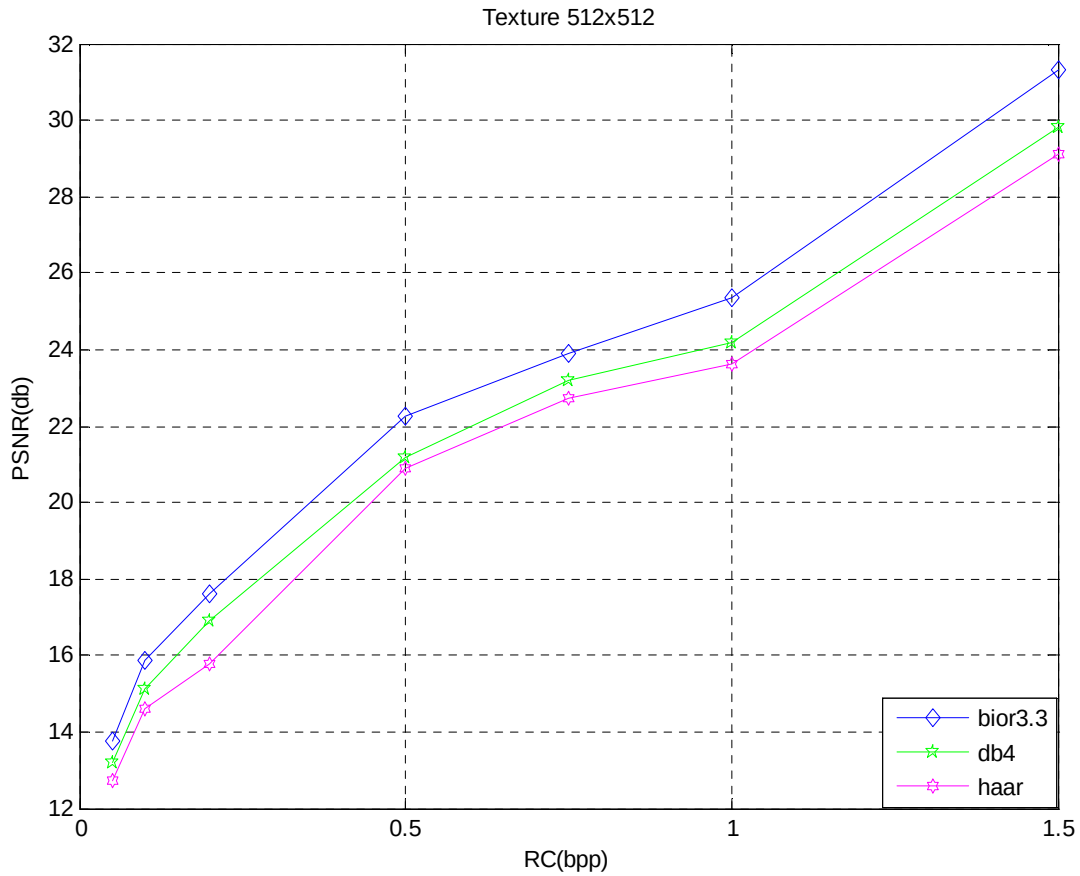


Figure 4.11 : Comparaison des différents filtres appliqués pour la transformation en ondelettes pour l'image Texture 512×512

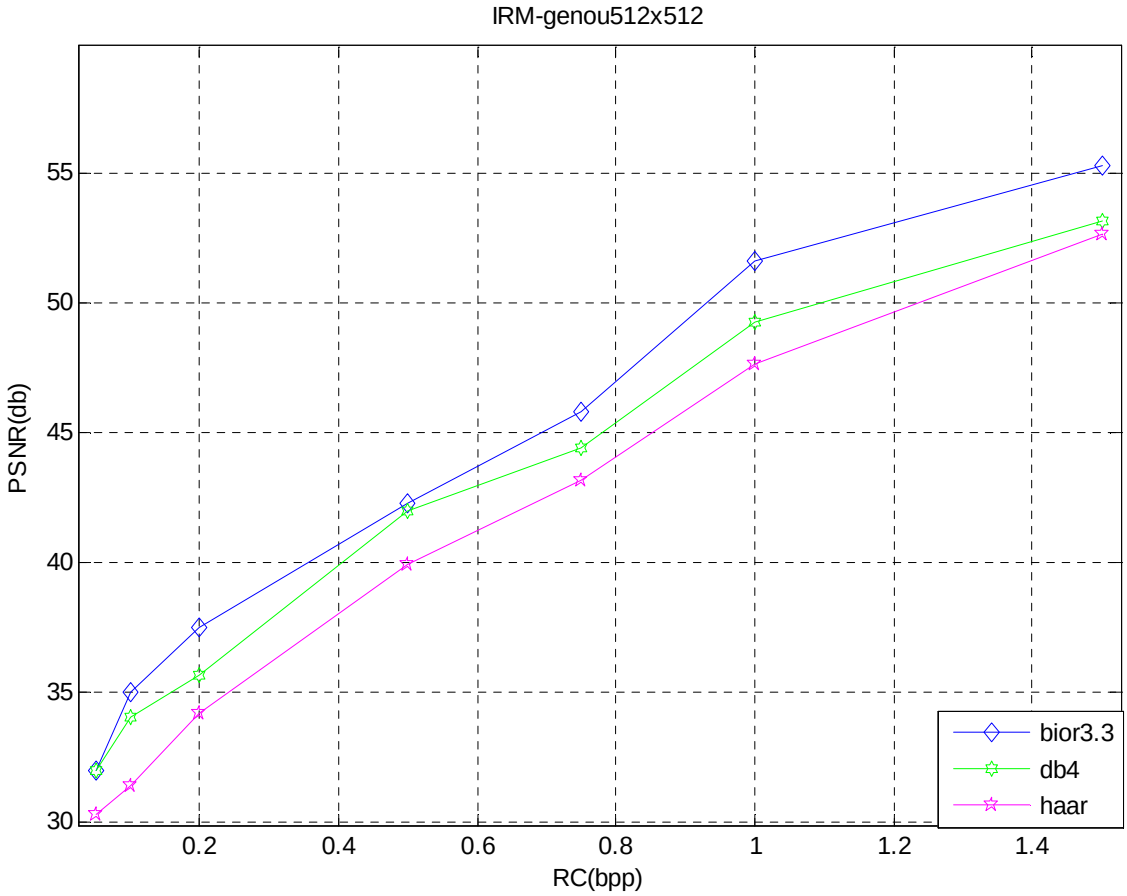


Figure 4.12 : Comparaison des différents filtres appliqués pour la transformation en ondelettes pour l'image IRM-genou 512×512

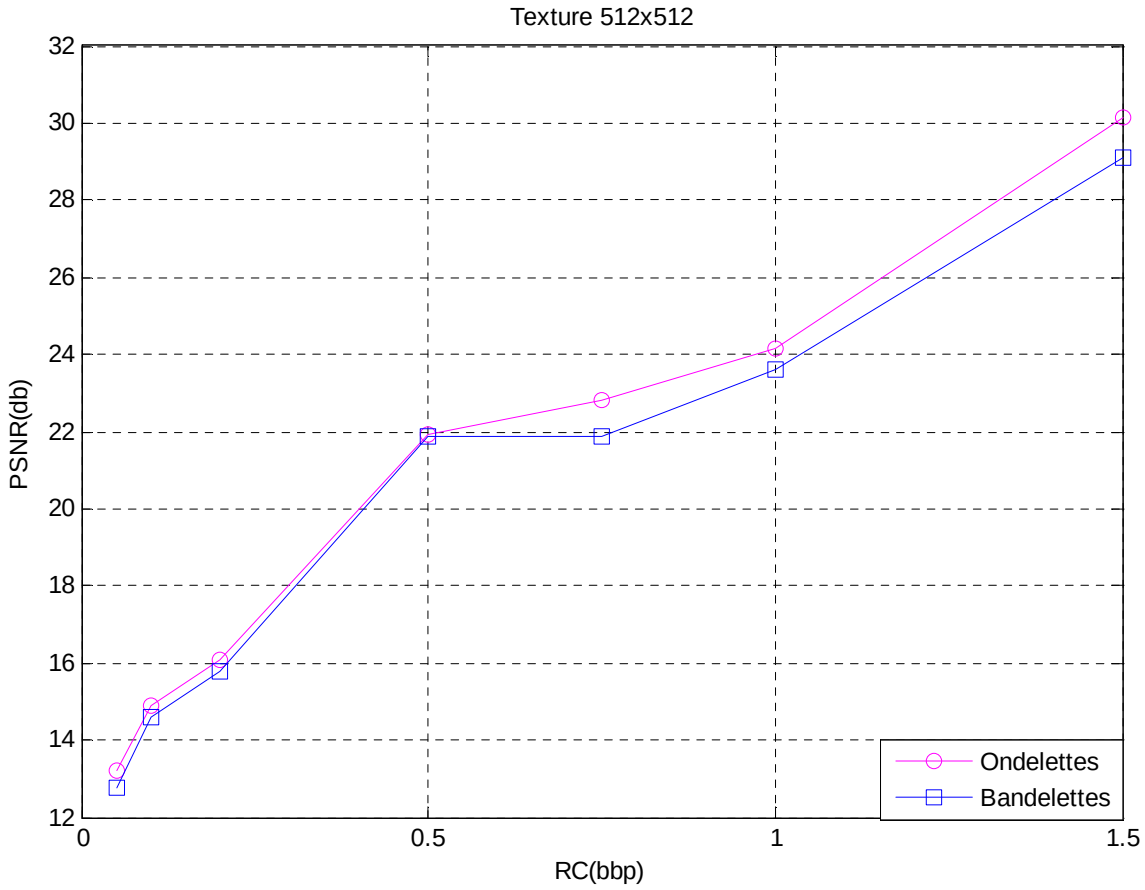


Figure 4.13 : Comparaison entre transformée ondelettes et bandelettes pour le codeur MSIHT sur l'image Texture 512×512

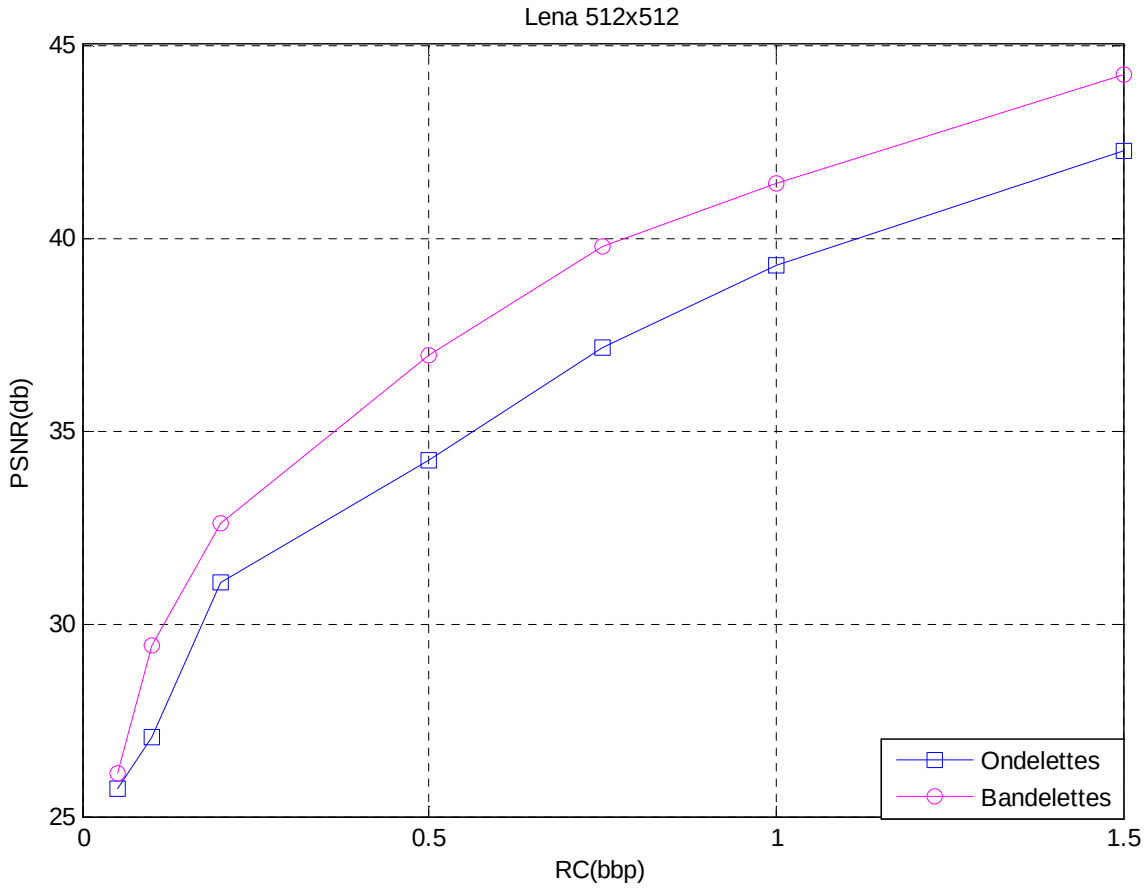


Figure 4.14 : Comparaison entre transformée ondelettes et bandelettes pour le codeur MSIHT sur l'image Lena 512×512

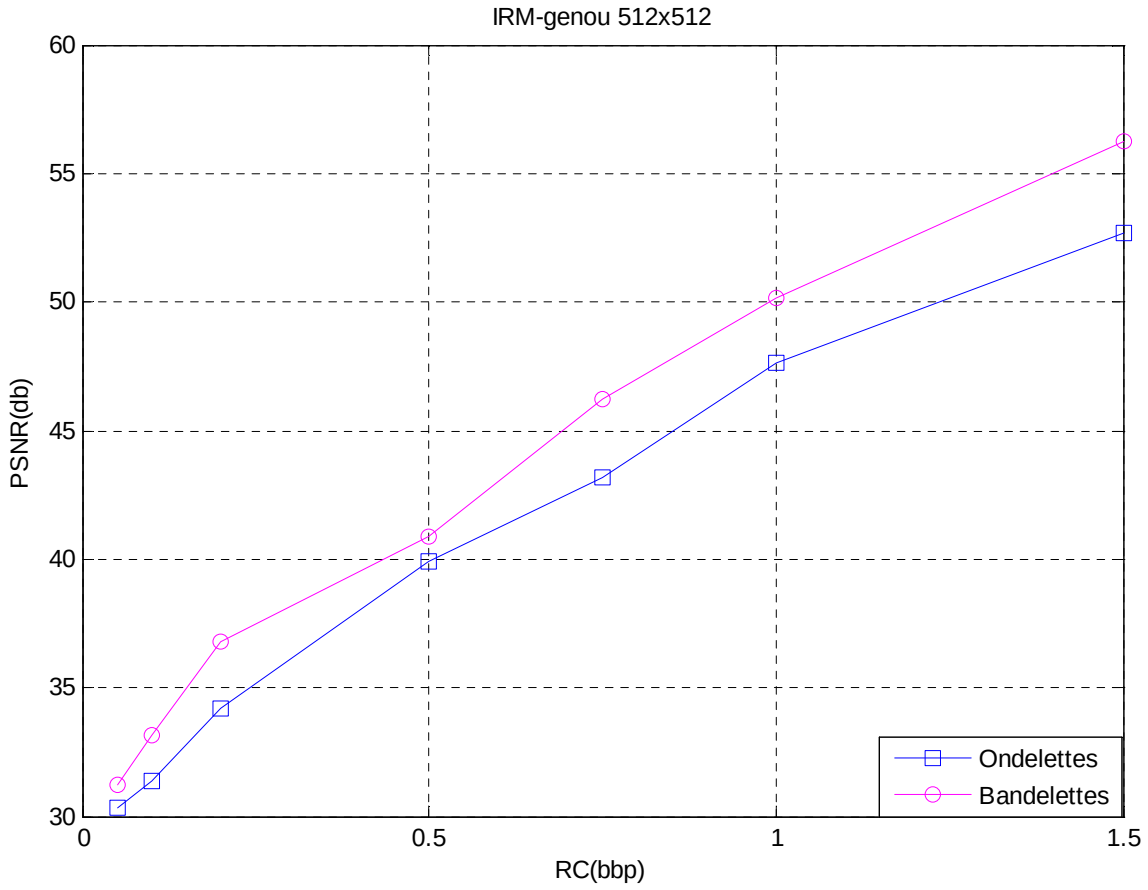
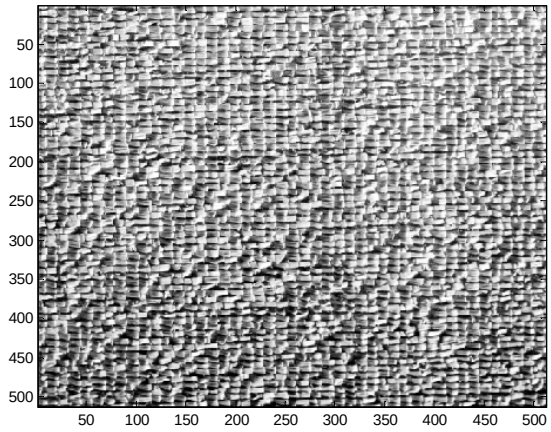
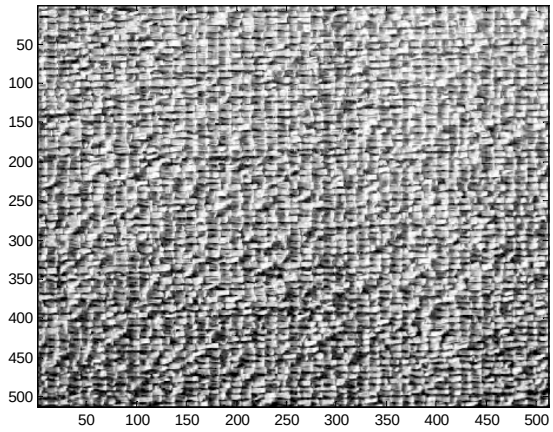


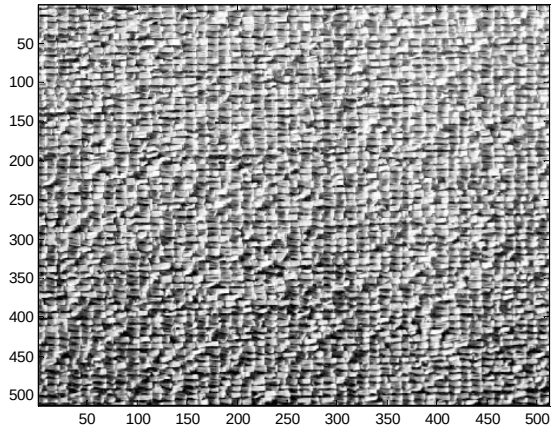
Figure 4.15 : Comparaison entre transformée ondelettes et bandelettes pour le codeur MSIHT sur l'image IRM-genou 512×512



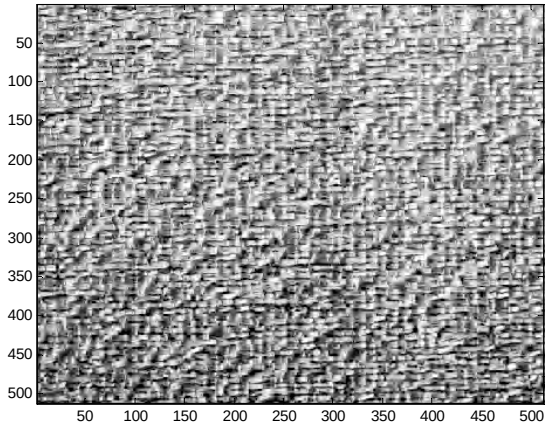
(a) Image original



(c) PSNR=24.2 dB CR=1 bpp



(b) PSNR=29,8 dB CR=1.5 bpp



(d) PSNR=19, 15 dB CR=0.75 bpp

Figure 4.13: Résultats de l'image Texture 512×512 reconstruite par MSPIHT

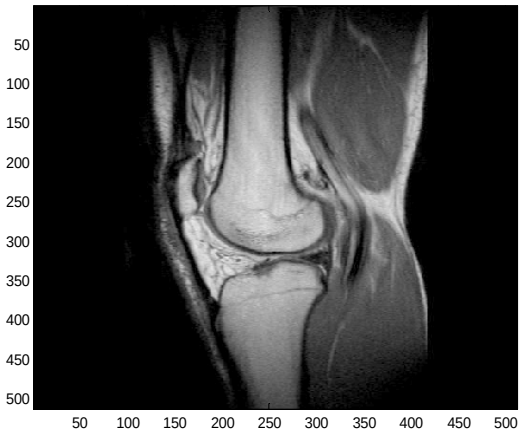
(a) Image original

(c) PSNR=41.35dB CR=1 bpp

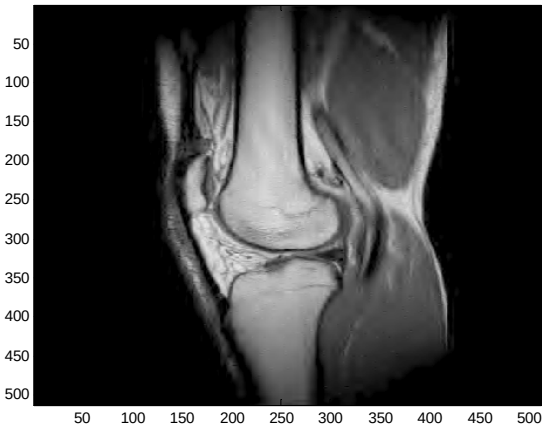
(b) PSNR=43.50 dB CR=1.5 bpp

(d) PSNR=39.24dB CR=0.75 bpp.

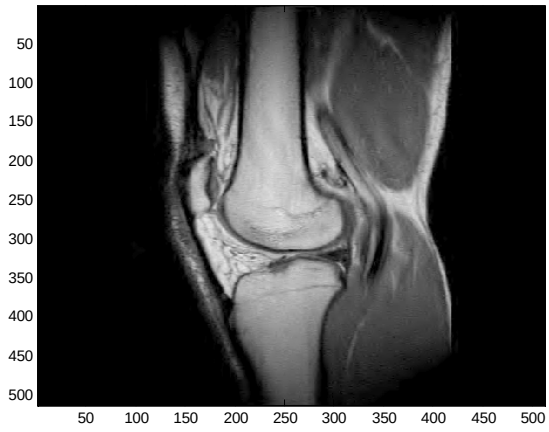
Figure 4.14: Résultats de l'image Lena 512×512 reconstruite par MSPIHT



(a) Image original



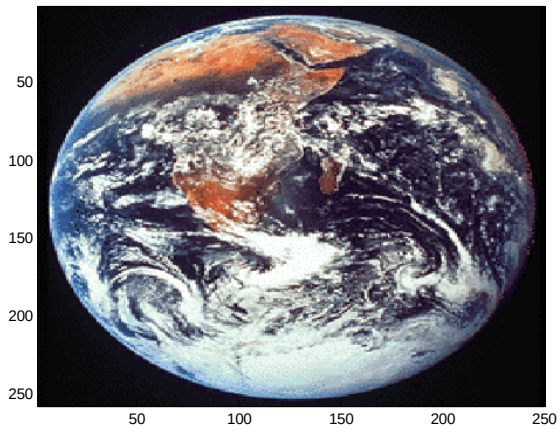
(c) PSNR=39,37 CR=1 bpp



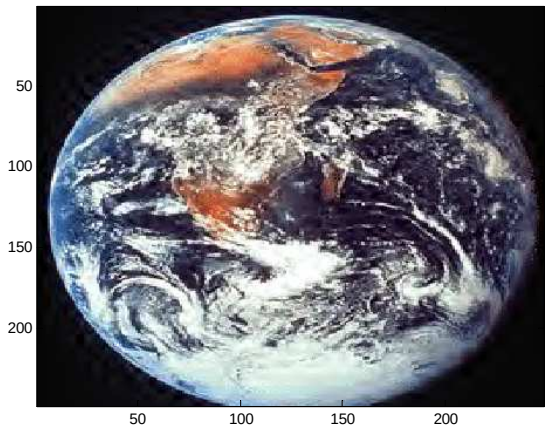
(b) PSNR=43.06 dB CR=1.5 bpp

(d) PSNR=37.22dB CR=0.75 bpp.

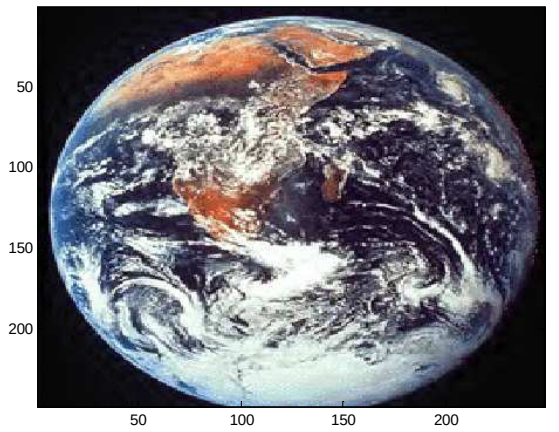
Figure 4.15: Résultats de l'image IRM-genou 512×512 reconstruite par MSPIHT



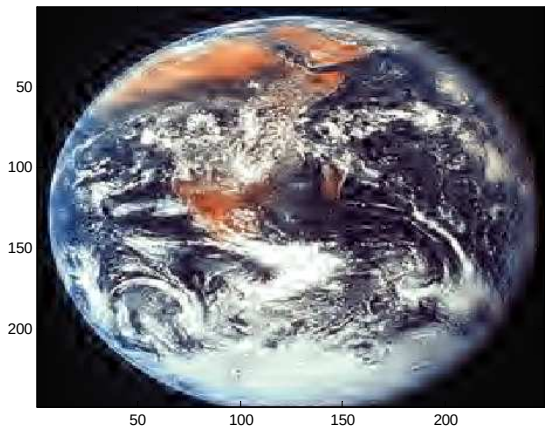
(a) Image original



(c) PSNR= 44,14 CR=1 bpp



(b) PSNR=54.16 dB CR=1.5 bpp



(d) PSNR=39.12dB CR=0.75 bpp.

Figure 4.16: Résultats de l'image earth couleur 248×248 reconstruite par MSPIHT

10.Conclusion

Nous avons à travers ce chapitre proposé un protocole de validation de notre méthode MSPIHT. Dans ce protocole nous avons cherché à analyser l'influence en terme :

- Types d'image.
- Taille d'image choix du filtre de l'ondelette.
- intérêt du codage entropique (arithmétique).
- Transformée bandelettes.

Pour cela nous avons utilisé la métrique suivante :

- PSNR (rapport signal sur bruit).
- RC (taux de compression).
- Temps de calcul (temps de l'opération codage décodage).

Nous avons alors constaté que notre méthode est en général la plus performante.

Conclusion générale

L'objet de cette thèse a été la recherche et l'évaluation de techniques pour améliorer la Prédiction d'images au sein d'un codeur numérique. L'enjeu a consisté à explorer des méthodes qui visent à reproduire et étendre des motifs de complexité variée, présents au sein des images numériques. Nos choix se sont orientés vers des solutions alliant la théorie du signal et des approches connues en codages imbriqués.

Dans ce contexte, nous avons considéré les algorithmes de compression par ondelettes et par codages imbriqués (SPIHT, MSPIHT). Le codeur imbriqué offre la propriété de la transmission progressive de l'image codée tout en apportant d'excellentes performances débit/distorsion.

Dans cette thèse, nous avons proposé un nouvel algorithme pour optimiser le codage SPIHT existant. Notre méthode dite Modified SPIHT (M-SPIHT) a deux spécificités :

- elle emploie un seul symbole au lieu de quatre, employés dans l'algorithme SPIHT original. Cette modification nous a permis de minimiser le nombre de symboles redondants et d'avoir une meilleure redistribution de l'entropie par rapport à l'algorithme SPIHT. A cet effet, nous avons obtenu une quantité d'informations inférieure par rapport à l'algorithme SPIHT.
- elle optimise le codage en utilisant un regroupement binaire des bits à coder. Ceci nous a permis d'avoir un codage entropique des symboles plus efficace que dans le cas de SPIHT.

A travers le protocole de validation que nous avons suivi dans le chapitre 4, nous avons alors montré que l'algorithme MSPIHT est meilleure du point de vue rapport taux de compression/qualité d'image dans la plupart des cas par rapport à l'algorithme SPIHT original et comparable avec celle des algorithmes EZW et JPEG 2000.

En perspectives, nous proposons d'une part l'application du principe du codage MSPIHT dans le cas des codeurs EZBC et EZW. Ce principe pourrait réduire le nombre de symboles (donc le nombre de bits) à coder en utilisant des codes spécifiques représentant les arbres de zéros dont la racine est signifiante et les descendants sont insignifiants.

D’autre part, nous proposons d’associer notre algorithme MSPIHT avec une transformation qui utilise les curvelete ou les contourlete. Ces dernières représentent l’image par un nombre de coefficients significatifs inférieur par rapport à celui obtenu par les ondelettes classiques. L’association de ces nouvelles transformations avec notre algorithme pourrait réduire en plus le nombre de coefficients à coder.

Bibliographie

[1] S. Bres, J-M Jolion et F. Lebourgeois, “Traitement et analyse des images numériques“, Edition Lavoisier, 2003.

[2] E. Incerti, “Compression d’images, algorithmes et standards“, Edition Vuibert, Paris, 2003.

[3] C. Shannon, “A mathematical theory of communication”. Bell System Technical Journal, pp. 379–423, Juillet 1948.

[4] D.A. Huffman, "A method for the construction of minimum-redundancy codes Proceedings of the I.R.E., sept 1952, pp 1098-1102

[5] P. G. Howard and J. S. Vitter, “Arithmetic Coding for Data Compression”, Proceedings of the IEEE, 82(6), pp. 857-865, June 1994.

[6] Lloud (S.P) Least square quantization in PCM . IEEE trans. On Information Theory vol. 28 N° 2 mars 1982 pp 129-137

[7] Analysis of optical near-field images by Karhunen—Loève transformation Daniel Charraut, Daniel Courjon, Claudine Bainier, and Laurent Moulinier, Applied Optics, Vol. 35, Issue 20, pp. 3853-3861 (1996)

[8] W. Chen, C.H. Smith, and S.C. Fralick. A fast computational algorithm for the discrete cosine transform. IEEE Trans. on Communications, COM-25:1004_1009, 1977.

[9] S. Grgic, M. Mrak, M. Grgic, ,”Comparison of JPEG Image Coders”, Proceedings of the 3rd International Symposium on Video Processing and Multimedia Communications, VIPromCom-2001, Zadar, 2001, Croatia, pp. 79-85.

[10] M.D. Adams, “The JPEG-2000 Still Image Compression Standard (Last Revised: 2002-12-25)”, ISO/IEC JTC 1/SC 29/WG1 N 2412, December 2002.

[11] S. Hsiang and J. W. Woods, “Embedded image coding using zeroblocks of subband/wavelet coefficients and context modeling,” in MPEG-4 Workshop and Exhibition, ISCAS 2000, May 2000.

[12] M. Shapiro, “Embedded image coding using zerotres of wavelet coefficients,” IEEE Trans. on Signal Processing, vol. 41, pp. 3445–3462, Dec. 1993.

[13] A. Islam and W. A. Pearlman, “An embedded and efficient low-complexity hierarchical image coder,” Visual Communications and Image Processing ’99, Proceedings of SPIE, vol. 3653, pp. 294–305, Jan. 1999.

[14] J. Mainaibeye, “Codage et compression d’images par ondelettes“, thèse de doctorat, Université de Tunis, ELMANAR, E.N.I.T, Juillet 2002.

[15] V. Chappelie, ”Codage progressif d'images par ondelettes orientées”. Thèse de doctorat, Université de Rennes 1. Déc 2005.

[16] K. Anis K. Nawres H. Kamel, “ Compression de séquences d’images scintigraphiques par régions d’intérêt avec l’algorithme de SPIHT ”, Compression et Représentation des Signaux Audiovisuels, CORESA'04, Lille, France, Mai 2004.

[17] J. Radon., On the determination of functions from their integral values along certain manifolds, in reports _ Saxon Academy of Sciences, Leipzig Math Nat,1917, vol. 69, pp. 262_277.

[18] M. N. Do and M. Vetterli, Orthonormal finite ridgelet transform for image compression, IEEE International Conference on Image Processing, 2000, vol. 2,pp. 367_370.

[19] E. J. Candes and D. L. Donoho, Curvelets _ a surprisingly effective non adaptive representation for objects with edges, pp. 1_10, Vanderbilt University Press, Nashville, TN, 1999.

[20] N. G. Kingsbury, The dual-tree complex wavelet transform : a new efficient tool for image restoration and enhancement,_ in European Signal Processing Conference,
Sept. 1998, pp. 319_322.

[21] M. N. Do and M. Vetterli, The contourlet transform : An efficient directional multiresolution image representation, IEEE Transactions on Image Processing, Oct. 2003.

[22] F. G. Meyer and R. R. Coifman, Brushlets : a tool for directional image analysis and image compression, in Applied and Computational Harmonic Analysis, 1997, vol. 4, pp. 147_187.

[23] D. L. Donoho and X. Huo, _Beamlet pyramids : a new form of multiresolution analysis, suited for extracting lines, curves and objects from very noisy image data, in SPIE Conference on Wavelet Applications in Signal and Image Processing, 2000, vol. 4119, pp. 434_444.

[24] D. L. Donoho, _Wedgelets : Nearly minimax estimation of edges,_ in Annals of Statistics, 1999, vol. 27(3), pp. 859_897.

[25] E. Pennec and S. Mallat, Image compression with geometrical wavelets, in IEEE International Conference on Image Processing, 2000, vol. 1, pp. 661_664.
[26] A. Cohen, «Ondelettes et traitement numérique du signal», Masson, Paris, 1992.
[27] M. Barlaud, T. Gaidon, P. Mathieu, J.C. Fauveau, «Edge detection using recursive biorthogonal wavelet transform», IEEE, vol. CH2977, pp. 2553-2556, 1991.
[28] M. Antonini, M. Barlaud, P. Mathieu, I. Daubechies, «Image coding using wavelet transform»,IEEEtrans.onImageProcessing,vol.1,N2, pp.205-216, april 1992.
[29] I. Daubechies, «Ten Lectures onWavelets», SIAM, Philadelphia, PA, 1992.
[30] S. Mallat, «Multiresolution approximations and wavelet orthonormal bases of L2(R)»,Trans.Am.Math.Soc.,vol.315,N1, pp. 69-87, sep. 1989.
[31] S. Mallat, «Multifrequency channel decomposition of images and wavelet models», IEEETrans.onAccousticSpeechandSignalProc.,vol.37,N12, pp. 2091-2110, dec. 1989.
[32] S. Mallat, S. Zhong, «Characterization of signals from multiscale edges», IEEE trans. on PAMI, vol. 14, n_7, pp. 710-732, july 1992.
[33] Christophe BERNARD, « Ondelettes et problèmes mal posés : la mesure du flot optique et l'interpolation irrégulière », Thèse de doctorat, Paris 2001.
[34] Sébastien ROUX, «Adéquation algorithme-architecture pour le traitement mulitimédia embarque », Thèse de Doctorat, Laboratoire France Télécom, 2002
[35] Ouafi abdelkarim, « Compression d'images avec pertes par codages imbriqués, Proposition d'une optimisation de l'algorithme EZW» Thèse de Doctorat, université Mohamed Khider Biskra, 2010
[36] A. Said and W.A. Pearlman, "A new, fast and efficient image codec based on set partitioning in hierarchical trees," IEEE Trans. Circuits and Systems for Video Technology, vol. 6, pp. 243–250, Juin 1996.
[37] Zitouni Athmane.. Baarir Zine Eddine, Ouafi Abd Elkarim and Taleb-Ahmed Abdelmalik. « A New Application of MSPIHT for Medical Imaging»,Journal of Applied Computer Science & Mathematics, no. 13 (6) /2012, Suceava.
[38] S. Grgic, M. Mrak, M. Grgic, ,"Comparison of JPEG Image Coders", Proceedings of the 3rd International Symposium on Video Processing and Multimedia Communications, VIP romCom-2001, Zadar, 2001, Croatia, pp. 79-85.
[39] H. Guitter, "La compression des images numériques", Hermes, 1995.

[40] G. Burel, “Introduction au traitement d’images, simulation sous matlab“, Edition Lavoisier, 2001.
[41] E. Incerti, “Compression d’images, algorithmes et standards“, Edition Vuibert, Paris, 2003.
[42] D. Lingrand, “Introduction au traitement d’images“, Edition Vuibert, Paris 2004.
[43]J.D. Villasenor, B.Belzer and J.Liano “Wavelet Filter Evaluation for image Compresion”, Proc. In IEEE International conference on Image Processing ICIP’97, Vol.1, pp.624-627,1997.
[44] J.K. Romberg, M. Wakin, and R. Baraniuk. Multiscale wedgelet image analysis : fast decompositions and modeling. In in IEEE Int. Conf. on Image Proc. 2002, volume 3,
pages 585–588, Jun. 2002.
[45] F. Friedrich, L. Demaret, H. Fuhr, and K. Wicker. Efficient moment computation over polygonal domains with an application to rapid wedgelet approximation. SIAM Journal on Scientific Computing, 29(2) :842, 2008.
[46] SERIEF Chahira, « Extraction automatique de points d’interet a base de la transformeé en contourlets non sous echantionnee pour le recalage », Thèse de doctorat, Constantine 2009.
[47] Jonathan TAQUET« Techniques avancées pour la compression d’images médicales », Thèse de doctorat, Institut National de Recherche en Informatique et en Automatique Centre INRIA Rennes - Bretagne Atlantique 2012.
[48] Xavier DELAUNAY «Compression d'images satellite par post-transformées dans le domaine ondelettes», université de Toulouse 2008.