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

UNIVERSITE MOHAMED KHIDER BISKRA

FACULTE DES SCIENCES ET TECHNOLOGIE

DEPARTEMENT D'ELECTRONIQUE

THESE

Présentée pour l'obtention du Diplôme de Doctorat en Sciences en Electronique

Par :

OUAFI Abdelkrim

Thème :

Compression d'images avec pertes par codages imbriqués,

Proposition d'une optimisation de l'algorithme EZW

À soutenir devant la commission d'examen :

Pr. M. Khamadja	Président	Prof	Univ Constantine
Dr. Z-E Baarir	Rapporteur	M.C	Univ Biskra
Pr. A. Taleb-Ahmed	Co-Encadreur	Prof	Univ Valenciennes, France
Pr. N. Doghmane	Examinateur	Prof	Univ Annaba
Pr. M. Diaf	Examinateur	Prof	Univ Tizi-Ouzou
Dr. S. Sbaa	Examinateur	M.C	Univ Biskra

2008/2009

Résumé

Dans ce travail, nous étudions les algorithmes de compression par ondelettes dites codages imbriqués. Ces codeurs sont basés sur la notion d'arbre de zéros (zerotree). Le premier codeur en sous-bandes par zerotree à avoir été introduit, est le codeur EZW. Ce codeur offre une représentation progressive de l'image tout en apportant d'excellentes performances de débit-distorsion par rapport à des codeurs bien plus complexes et non progressifs de l'époque.

Dans le cadre de cette thèse, nous proposons une nouvelle approche de compression d'images basée sur le principe de l'algorithme EZW de Shapiro. Notre nouvelle approche notée M-EZW (Modified EZW) consiste à utiliser six symboles au lieu de quatre employés dans l'algorithme de Shapiro, pour minimiser le nombre de symboles redondants d'une part et à optimiser le codage par un regroupement binaire des bits à coder d'autre part. Les résultats obtenus par cette nouvelle approche en termes de PSNR et de taux de compression obtenus par Shapiro sont améliorés 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, SPECK et EBCOT de JPEG 2000.

Mots-clés : *Compression d'images, Algorithme EZW de Shapiro, Algorithme M-EZW, Entropie, Codage, PSNR, Taux de compression, Algorithmes SPIHT, SPECK, EBCOT, JPEG2000.*

Abstract

In this work, we have studied the compression algorithms using wavelets called embedded encoding. These encoders are based on the concept of zero tree. The first sub-bands encoder using zerotree concept to have been introduced is the EZW (Embedded Zero-tree Wavelet) encoder. This encoder provides a progressive image representation while providing excellent performances compared to more complex encoders and non-progressive at the time.

In this thesis, we propose a new approach to image compression based on the principle of Shapiro's EZW algorithm. Our approach, the Modified-EZW (M-EZW), uses six symbols instead of four used in the original Shapiro's algorithm to minimize the redundant symbols, and optimizes the coding by a binary regrouping of the information. This approach produces results that have a significant improvement over the PSNR and compression ratio obtained by Shapiro without affecting the computing time. These results are, also, comparable to those obtained using the SPIHT, SPECK and JPEG2000 algorithms.

Index Terms— Image compression, Shapiro's EZW algorithm, M-EZW algorithm, Entropy, Coding, PSNR, Compression ratio, SPIHT, SPECK and JPEG2000 algorithms.

ملخـــــص

في اطار هذه الرسالة قمنا بدراسة برامج ضغط الصورة المعتمدة على ‹الأمواج القصيرة› (ondelettes) باستعمال التشفيرات المتداخلة. هذه أصفار (zerotree). أول مشفر بالأندلات إعتمد على مبدأ شجرة الأصفار كان البرنامج EZW . هذا البرنامج يمكن من تمثيل الصورة بشكل متدرج و مكن من الحصول على نتائج جيدة من حيث المردود والنوعية بالمقارنة مع مشفرات أخرى أكثر تعقيدا معروفة في نفس الفترة.

في إطار هذا العمل المتواضع, نقترح طريقة جديدة ظغط الصورة تعتمد على نفس مبدأ المشفر EZW ل: Shapiro. الطريقة المقترحة مسماة M-EZW (أي EZW المغيرة) تتمثل في إستعمال ستة رموز بدل الأربعة المستعملة في برنامج Shapiro, وذلك من أجل التقليل من عدد الرموز المتكررة من جهة و تحسين التشفير باستعمال ظم الرموز قبل التشفير من جهة أخرى . النتائج المتحصل عليها بهذه الطريقة الجديدة بمقياس نسبة التشوه PSNR ونسبة الظغط تظهر تحسن بالمقارنة بطريقة Shapiro وخاصة في حالة النسب المتوسطة و المرتفعة دون التغير في الزمن المستغل للعملية. كما أن النتائج التي تحصلنا عليها متقاربة مع تلك المتحصل عليها بالطرق المعروفة في هذا المجال : SPIHT, SPECK, JPEG 2000 .

كلمات المفاتيح: ظغط الصورة, برنامج EZW ل: Shapiro, البرنامج M-EZW, التشفير . PSNR, نسبة الظغط, SPIHT, SPECK, JPEG 2000,

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 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. DHOGHMANE Noureddine pour son aide précieuse et d'avoir accepté de juger ce travail.

Je remercie Pr. KHAMADJA Mohamed qui me fait l'honneur de présider ce jury.

Je remercie Pr. DIAF Moussa et Dr. SBAA Salim d'avoir accepter de juger ce travail.

Je tiens également à remercier mes collègues du département d'Electronique et tous mes amis.

Dédicace

A mes chères parents,

A la mémoire de mon frère Farouk,

A ma femme et mon garçon Mouatez- Billeh,

A toute ma famille,

A tous mes amis,

Abdelkrim

Table Des matières

Table Des matières i

Liste Des Figures iii

Liste Des Tables v

Introduction 1

Chapitre 1 : Etat de l'art des méthodes de compression d'images avec pertes 3

1 INTRODUCTION 4
2 CARACTERISTIQUE D'UN CODEUR D'IMAGES 5
3 COMPRESSION D'IMAGES AVEC PERTES 6
3.1 Norme JPEG 6
3.2 Norme JPEG 2000 7
A. Principe 8
B. Codage EBCOT 8
C. Principaux avantages du codage JPEG 2000 9
3.3 Codages imbriqués 9
A. Algorithme EZW de Shapiro 10
B. Algorithme SPIHT 11
C. Algorithme SPECK 14
D. Algorithme EZBC 16
4 AUTRE PROGRAMMES DE COMPRESSION 16
5 CONCLUSION 16

Chapitre 2 : Codeur EZW de Shapiro. Proposition d'une optimisation M-EZW 16

1 INTRODUCTION 17
2 TRANSFORMATION EN ONDELETTES DISCRETE DWT 17
2.1 Analyse temps-fréquence 18
2.2 Analyse temps-échelle 19
3 ALGORITHME EZW 22
3.1 Principe 22
3.2 Test de signifiance 25
A. Initialisation 25
B. Test de signifiance 26
C. Quantification et Raffinement 28
D. Réitération 29
4 ALGORITHME PROPOSE M-EZW 31
4.1 Test de signifiance 32
4.2 Comparaison quantitative 35
A. Nombre de symboles 36
B. Temps de calcul 36
4.3 Procédure de codage des symboles de signifiance 37
5 CONCLUSION 38

Chapitre 3 : Résultats et Discussions 39

1 INTRODUCTION 40
2 PARAMETRES DE VALIDATION 40
2.1 Quantité d'information et entropie 40

2.2 Taux de compression 41
2.3 Temps de calcul 42
3 IMAGES DE TEST 42
4 ALGORITHME EZW 44
4.1 Principe de codage progressif 46
4.2 Choix de l'ondelette et du niveau de décomposition 46
5 ALGORITHME PROPOSE M-EZW 47
5.1 Test de signifiance 49
5.2 Procédure de codage des symboles de signifiance 53
6 COMPARAISON AVEC LES ALGORITHMES ACTUELS 58
7 CONCLUSION 59
Conclusion générale 70
Annexe 71
Bibliographie 72

Liste Des Figures

Figure 1.1 : Schéma fonctionnel d'une technique de compression générique avec pertes. 6

Figure 1.2 : Schéma fonctionnel de l'algorithme JPEG. 6

Figure 1.3 : Résultats de l'image Lena reconstruite à 0.2 bpp, a) par la norme JPEG avec PSNR =23.4 dB ; b) par la norme JPEG2000, avec PSNR=28.7 dB. 7

Figure 1.4 : Principe du codage EBCOT 9

Figure 1.5: Exemple de transformation en sous-bande de l'image Lena, la ressemblance entres les sous bandes est remarquable. 10

Figure 1.6 : Exemples de descendances parent-fils dans le cas SPIHT, le pixel désigné par (*) n'accepte pas de descendants. 11

Figure 1.7 : Partitionnement de l'image X en deux sous-ensembles : S et le reste $\mathcal{T}$ 14

Figure 1.8 : Principe de partitionnement des ensembles $\mathcal{T}$ 14

Figure. 2.1 Espaces de représentation : domaine de Fourier (a), espace temps-fréquence (b) et espace temps-échelle (c) 19

Figure 2.2 Analyse multi-résolutions en deux dimensions 21

Figure 2.3 Décomposition en ondelettes à trois étapes 22

Figure 2.4 : Exemple de décomposition de l'image Lena par ondelettes en trois résolutions. 23

Figure 2.5 : Relation parent fils. 24

Figure 2.6 : Modèle de dépendances inter-bandes pour l'EZW 2D. 24

Figure 2.7: Ordre de balayage des coefficients des sous-bandes 25

Figure 2.8 : Exemple de décomposition à trois résolutions d'une matrice de taille 8x8. 26

Figure 2.9 : Principe de test de signifiance des coefficients pour l'EZW 27

Figure 2.10 : Principe de l'algorithme EZW pour un cycle de compression 28

Figure 2.11 : Principe de quantification et raffinement 29

Figure 2.12 : Exemple de quantification des coefficients {63, -34, 49 et 47} pour T_0=32. 29

Figure 2.13 : Principe de test de signifiance des coefficients pour l'algorithme M-EZW. 33

Figure 2.14 : 2eme Exemple de décomposition à trois résolutions d'une matrice de taille 8x8, dont seulement la racine est significative par rapport au seuil T_0=32. 35

Figure 2.15 : Exemple de regroupement binaire des éléments de la liste D en 9 bits (a), et des bits de la liste S en 8 bits (b). 37

Figure 3.1 : Images testes utilisées 43

Figure 3.2 : Organigramme du codage EZW 45

Figure 3.3 : Organigramme du codage M-EZW proposé 48

Figure 3.4 : Exemple de regroupement binaire des éléments de la liste C en 9 bits (a), et des bits de la liste S en 8 bits (b). 54

Figure 3.5. Comparaison des différents algorithmes de compression appliqués sur l'image **Lena.** 61

Figure 3.6. Comparaison des différents algorithmes de compression appliqués sur l'image **Barbara** 61

Figure 3.7. Comparaison des différents algorithmes de compression appliqués sur l'image **Goldhill** 62

Figure 3.8. Comparaison des différents algorithmes de compression appliqués sur l'image **Peppers** 62

Figure 3.9. Comparaison des différents algorithmes de compression appliqués sur l'image **Fruits** 63

Figure 3.10. Comparaison des différents algorithmes de compression appliqués sur l'image **Airplane** 63

Figure 3.11. Comparaison des différents algorithmes de compression appliqués sur l'image **Einstein** 64

Figure 3.12. Comparaison des différents algorithmes de compression appliqués sur l'image **Cameraman** 64

Figure 3.13. Comparaison des différents algorithmes de compression appliqués sur l'image Mri 65

Figure 3.14. Comparaison des différents algorithmes de compression appliqués sur l'image **Abdo** 65

Figure 3.15. Comparaison des différents algorithmes de compression appliqués sur l'image **Angio** 66

Figure 3.16. Comparaison des différents algorithmes de compression appliqués sur l'image **Chest** 66

Figure 3.17 : Résultats de l'image **Lena** reconstruite par les algorithmes : a) EZW, b) M-EZW, c) SPIHT et d) JPEG 2000 pour RC= 0.50 bpp 67

Figure 3.18 : Résultats de l'image **Barbara** reconstruite par a) EZW, b) M-EZW, c) SPIHT et d) JPEG 2000 pour RC= 0. 50 bpp 68

Figure 3.19 : Résultats de l'image **Goldhill** reconstruite par a) EZW, b) M-EZW, c) SPIHT et d) JPEG 2000 pour RC= 0.50 bpp 69

Liste Des Tables

Table 2.1 : Différentes étapes de l'algorithme EZW pour une itération (T_0=32) 30

Table 2.2 : Différentes étapes de l'algorithme M-EZW pour une itération (T_0=32), ... 34

Table 2.3 : Résultats des algorithmes EZW et M-EZW appliqués aux matrices de test de la figure 2.8 (A) et figure 2.14 (B). 35

Table 3.1 : Nombre total des symboles et fréquences d'apparition de chaque symbole obtenus par les deux algorithmes EZW et M-EZW appliqués sur l'image **Lena** 512*512 pour différents seuils T. 50

Table 3.2 : Nombre total des symboles et fréquences d'apparition de chaque symbole obtenus par les deux algorithmes EZW et M-EZW appliqués sur l'image **Barbara** 512*512 pour différents seuils T. 50

Table 3.3 : Nombre total des symboles et fréquences d'apparition de chaque symbole obtenus par les deux algorithmes EZW et M-EZW appliqués sur l'image **Goldhill** 512*512 pour différents seuils T. 50

Table 3.4 : Probabilités des différents symboles obtenus par les deux algorithmes EZW et M-EZW appliqués sur l'image **Lena** 512*512 pour différents seuils T. 51

Table 3.5 : Probabilités des différents symboles obtenus par les deux algorithmes EZW et M-EZW appliqués sur l'image **Barbara** 512*512 pour différents seuils T. 51

Table 3.6 : Probabilités des différents symboles obtenus par les deux algorithmes EZW et M-EZW appliqués sur l'image **Goldhill** 512*512 pour différents seuils T. 51

Table 3.7 : Entropie des différents symboles obtenus par les deux algorithmes EZW et M-EZW appliqués sur l'image **Lena** 512*512 pour différents seuils T. 52

Table 3.8 : Entropie des différents symboles obtenus par les deux algorithmes EZW et M-EZW appliqués sur l'image **Barbara** 512*512 pour différents seuils T. 52

Table 3.9 : Entropie des différents symboles obtenus par les deux algorithmes EZW et M-EZW appliqués sur l'image **Goldhill** 512*512 pour différents seuils T. 52

Table 3.10 : Codage arithmétique de la liste **C** de l'image **Lena** pour différents regroupements de N symboles 55

Table 3.11 : Codage arithmétique de la liste **C** de l'image **Barbara** pour différents regroupements de N symboles 55

Table 3.12 : Codage arithmétique de la liste C de l'image Goldhill pour différents regroupements de N symboles . 55

Table 3.13 : Codage arithmétique de la liste **S** de l'image **Lena** pour différents regroupements de N bits 56

Table 3.14 : Codage arithmétique de la liste **S** de l'image **Barbara** pour différents regroupements de N bits. 56

Table 3.15 : Codage arithmétique de la liste **S** de l'image **Goldhill** pour différents regroupements de N bits 56

Table 3.16 : Résultats des différents algorithmes appliqués sur les douze images de test. 60

Introduction

L'utilisation massive des images numériques engendre aujourd'hui des volumes de données de plus en plus importants. La compression de ces images numériques devient alors une nécessité afin d'assurer leur archivage d'une part et de faciliter leur transmission d'autre part.

Parmi les nombreuses méthodes de compression proposées dans la littérature, certaines visent une reconstruction parfaite des données originales. Ces techniques de compression sans pertes conduisent cependant à des taux de compression relativement faibles et sont dédiées aux applications sensibles telles que la compression des images médicales et des images satellitaires, ainsi que la compression des fichiers informatiques.

Lorsque l'application nécessite des débits limités, on s'oriente vers les méthodes tolérant une perte contrôlée de l'information (perte souvent invisible à l'œil). Ces méthodes dites avec pertes (lossy) permettent d'atteindre des taux de compression élevés pour une qualité visuelle acceptable.

De nombreux schémas de compression ont été proposés et standardisés tels que les normes JPEG pour les images fixes, les normes MPEG pour les images vidéo. L'ensemble de ces standards reposent sur la Transformée Discrète en Cosinus (DCT).

Depuis quelques années, 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.

Plusieurs algorithmes de compression utilisant les ondelettes ont été proposés. L'une des applications les plus connues en compression d'image est la norme JPEG 2000 qui donne généralement des résultats plus satisfaisants que ceux obtenus par la norme JPEG.

D'autres algorithmes de compression par ondelettes dites codages imbriqués ont été proposés. Ces codeurs sont basés sur la notion d'arbre de zéros (zerotree) proposée par Shapiro en 1993. Le premier codeur en sous-bandes par zerotree à avoir été introduit est le codeur EZW (Embedded Zerotree Wavelet). Ce codeur offre une représentation progressive de l’image tout en apportant d’excellentes performances de débit-distorsion par rapport à des codeurs bien plus complexes et non progressifs de l’époque. Les codeurs successifs comme SPIHT (Set-Partitioning in Hierarchical Trees) et SPECK (Set-Partitioning Embedded Block) ont amélioré le principe de l’EZW en proposant un codage plus efficace de l’information de signifiance qui code la position des coefficients énergétiques. Comme cette information représente une grande part du débit de l’image, son codage est primordial. Ce même principe est aussi utilisé dans le codeur EBCOT de JPEG 2000, offrant des performances supérieures, en termes de débit distorsion par rapport à SPIHT du fait de l’utilisation de contextes de codages de plus grande dimension, au prix d’une complexité accrue.

Notre travail dans le cadre de cette thèse consiste en l’étude de ces codages imbriqués (EZW, SPIHT et SPECK) et la proposition d’une optimisation de l’algorithme EZW nommée M-EZW (Modified-EZW). Notre méthode consiste à utiliser six symboles au lieu de quatre employés dans l'algorithme EZW de Shapiro, pour minimiser le nombre de symboles redondants d’une part et à optimiser le codage par un regroupement binaire des bits à coder d’autre part. Les résultats obtenus par cette 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, SPECK et EBCOT de JPEG 2000.

Notre thèse est organisée selon les chapitres suivants :

Le premier chapitre est dédié à un état de l’art sur la compression d’images, notamment la compression sans pertes. Un rappel sur les codages imbriqués est aussi proposé.

Le deuxième chapitre commence par une présentation détaillée de l’algorithme EZW de Shapiro avec des exemples d’utilisations. Il se poursuit par notre proposition d’une optimisation de ce dernier (algorithme MEZW).

Le troisième chapitre présente les résultats obtenus sur différentes images testes. Une analyse des résultats obtenus ainsi qu’une étude comparative de ces résultats est proposée.

Nous terminons cette thèse par une conclusion générale et les perspectives envisagées concernant notre travail.

Chapitre 1 : Etat de l'art des méthodes de compression d'images avec pertes

Résumé

Nous nous limiterons, dans ce chapitre à la présentation des principales méthodes de compression sans pertes proposées dans la littérature. On commence par décrire les caractéristiques principales d'un codeur d'images. Ensuite un bref rappel sur les méthodes de compression avec pertes les plus utilisées est souligné. La grande partie de ce chapitre sera consacrée aux méthodes de compression basées sur la transformation en ondelettes, plus spécifiquement les codages imbriqués.

1 Introduction

Les images numériques sont présentes dans une très grande proportion d'applications spécialisées et d'utilisation grand public. Cet essor ne s'est pas atténué depuis les années 1970, et il est porté par les évolutions des méthodes d'acquisition, des capacités de stockage et des performances de calcul.

Malgré l'augmentation continue des capacités des supports de stockage et les débits des réseaux, on a toujours besoin de compresser l'image vue l'évolution de l'utilisation des images et des vidéos et l'augmentation des capacités d'acquisition des capteurs numériques. L'opération de compression d'images est utile soit pour l'archivage sur des supports de mémoires ou pour la transmission.

La compression d'image peut être effectuée avec pertes de données ou sans pertes. Les méthodes de compression sans pertes sont dédiées aux applications sensibles telles que la compression des images médicales et les images satellitaires, ainsi que la compression des fichiers informatiques, où la perte d'un seul bit peut mener à des résultats catastrophiques. Ces méthodes conduisent à des taux de compression relativement faibles [3].

Les méthodes avec pertes sont particulièrement appropriées aux images naturelles telles que des photos dans les applications où une perte mineure de fidélité (parfois imperceptible) est acceptable pour réaliser une réduction substantielle du débit binaire [3].

2 Caractéristique d’un codeur d’images

En fonction de l’application recherchée, différentes qualités sont demandées à un algorithme de compression. Les caractéristiques principales d’un codeur d’image sont [8]:

- Rapidité de la compression/décompression :

  Cela diffère selon l’application (transmission ou archivage). Dans une application de transmission, le temps passé par l’opération de compression/décompression doit être inférieur au temps gagné par la réduction de la taille des données à transmettre. Dans le cas de l’archivage, cette condition n’est pas essentielle.
- Robustesse de l’algorithme :

  C’est la capacité d’un codeur à maintenir ses performances, malgré des changements de types d’images et dans le cas de présence d’erreurs de transmission.
- Taux de compression et qualité de l’image après un cycle de compression/ décompression :

  Il existe des algorithmes de compression sans pertes mais les taux de compression sont limités. Ces algorithmes sont préférés pour des applications sensibles telles que l’imagerie médicale ou les fichiers informatiques. Les algorithmes de compression avec pertes d’information permettent d’obtenir des meilleurs taux de compression mais en jouant sur la dégradation que l’on tolère. Selon l’application visée, on choisit entre qualité meilleure et taux de compression relativement faible, ou un taux de compression élevé avec une qualité plus au moins acceptable.
- Complexité algorithmique des codeurs/décodeurs et la possibilité d’implémentation de ces algorithmes dans des cartes d’acquisition en tems réel.

3 Compression d'images avec pertes

Les techniques de compression avec pertes se décomposent généralement en deux ou trois phases (cf. figure 1.1) : une phase de décorrélation du signal originale (élimination d'une part de l'information redondante), éventuellement une phase de quantification (élimination d'une part d'information non redondante mais jugée non indispensable) et enfin une phase de compression proprement dite (application d'un codage entropique sur le signal décorrélé et quantifié).

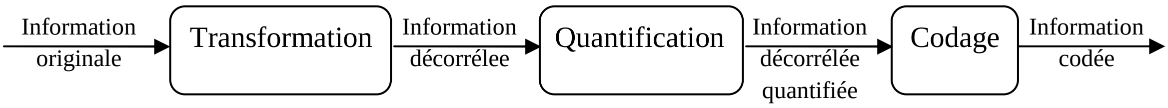


Figure 1.1 : Schéma fonctionnel d'une technique de compression générique avec pertes.

Le domaine de la compression d'images avec pertes est un domaine de recherche très actif depuis de nombreuses années. Les travaux ont notamment conduit aux normes JPEG [10] et JPEG2000 [16]. Cette dernière met la barre très haute en termes de débit/distorsion.

3.1 Norme JPEG

Après une longue période de recherche et de développement, avec la participation d'une très grande communauté scientifique, le JPEG est devenu un standard international en 1993. Le schéma fonctionnel de l'algorithme JPEG est illustré en figure 1.2.

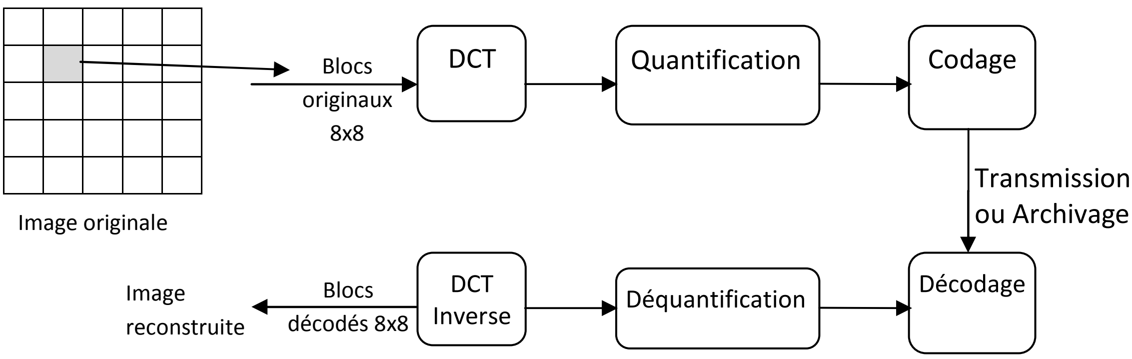


Figure 1.2 : Schéma fonctionnel de l'algorithme JPEG.

L'image est subdivisée en blocs de taille 8x8. Chaque bloc est traité indépendamment en trois étapes [4, 7]:

- Une transformation DCT [12] de chaque bloc : il s'agit d'une analyse spectrale locale par des fonctions cosinusoidales, on obtient 8x8 coefficients fréquentiels, où l'information est concentrée sur peu de coefficients.
- Une étape de quantification : la quantification est adaptée au contenu des blocs DCT afin de garder tous les coefficients significatifs quels que soient leur position.
- Etape de codage : chaque plan spectral quantifié est formé de quelques coefficients non nuls et une majorité de coefficients nuls qui ont été éliminés par l'étape de quantification. Un balayage zigzag est ensuite appliqué sur chaque matrice 8x8 suivie par un codage entropique spécifique à la norme JPEG.

La norme JPEG est actuellement très utilisée pour coder les images numériques que l'on trouve dans notre vie quotidienne (internet, appareils photos numériques,… etc.). Elle est bien adaptée pour les images naturelles et pour des taux de compression ne dépassant pas 8-10 (0.8 à 1 bit par pixel) [4]. Au-delà, des artefacts (effet des blocs) vont apparaitre et seront très gênants visuellement (figure 3.a). La norme JPEG2000 permet de résoudre ce problème en assurant une grande qualité visuelle à des forts taux de compression. La figure 3 montre un exemple d'une image reconstruite par les normes JPEG et JPEG2000 pour le même taux de compression ; la différence dans la qualité visuelle est nette.

Figure 1.3 : Résultats de l'image Lena reconstruite à 0.2 bpp, a) par la norme JPEG avec PSNR =23.4 dB ; b) par la norme JPEG2000, avec PSNR=28.7 dB.

3.2 Norme JPEG 2000

Des applications relativement peu développées lors de la définition de la norme JPEG occupent désormais une place prépondérante dans le monde de l'imagerie numérique : internet, médecine, appareils photos numériques, transmission d'images sur des réseaux mobiles… etc. Les besoins identifiés pour ces applications peuvent parfois être difficilement satisfaits par la norme JPEG, ce qui a conduit à créer une nouvelle version de la norme ; le JPEG 2000, pour intégrer des techniques plus performantes et répondre à ces nouveaux besoins.

A. Principe

La norme JPEG2000 est basée sur la transformée en ondelettes discrètes DWT (cf. chapitre 2) au lieu de la DCT utilisée par la norme JPEG. La DWT permet une analyse plus localisée donc plus fine de l'information ce qui n'est pas possible avec la DCT.

Le but de ces transformées en général, est de décorréler les données brutes de l'image représentées par ses pixels. Cette décorrélation n'est cependant pas parfaite et les coefficients obtenus après transformation restent dépendants statistiquement. Deux classes de méthodes de compression basées sur les ondelettes font référence aujourd'hui : les algorithmes de codage imbriqués {EZW (Embedded Zerotree Wavelet), SPIHT (Set Partitionning In Hierarchical Trees) et SPECK (Set Partitioned Embedded bloCK)} et l'algorithme EBCOT (Embedded Block Coding with Optimized Truncation).

Ces méthodes ce distinguent par la procédure de sélection des coefficients d'ondelettes avant l'étape de quantification. Deux principales approches permettent d'éliminer les coefficients non significatifs:

- La sélection par sous-bande qui consiste à mettre à zéro tous les coefficients non significatifs inférieurs à un seuil dans chacune des sous-bandes séparément (technique intra-bande). La corrélation inter-bande n'est pas donc prise en compte. La sélection des coefficients est effectuée par l'algorithme EBCOT (sélection par contexte). Cette approche est en grande partie intégrée dans la nouvelle norme JPEG 2000.
- La sélection multi-résolutions qui consiste à tenir compte des seuillages effectués à travers les sous-bandes d'une même direction (technique inter-bande). Ces techniques exploitent les ressemblances structurelles dans les sous-bandes associées à une direction et des résolutions successives. Ces techniques sont dites basées arbre de zéros (cf. section 4.3).

B. Codage EBCOT

Le codeur EBCOT [16] n'appartient pas à la famille des schémas de compression à arbre de zéros. Son principe est résumé en figure 1.4.

L'image peut être traitée toute entière ou divisée en 'tuiles' rectangulaires. Ces tuiles (de tailles 16x16, 32x32 ou 64x64) seront ensuite codées de façon indépendante. Les tuiles (ou l'image) sont ensuite décomposées en sous-bandes sur plusieurs niveaux. Une quantification scalaire est appliquée sur les différentes sous-bandes afin de réduire la précision des coefficients supposés non significatifs. Chaque sous-bande est ensuite découpée en petits blocs indépendants appelés 'code-blocs'. Ces derniers sont alors codés de façon progressive, par plan de bits et au moyen d'un codeur arithmétique contextuel (cf. figure 1.4).

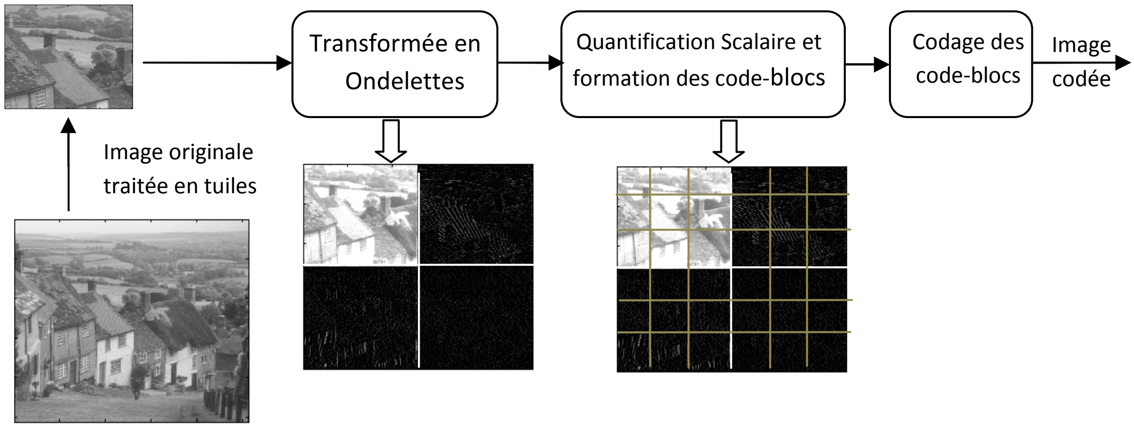


Figure 1.4 : Principe du codage EBCOT

C. Principaux avantages du codage JPEG 2000

Parmi les nombreux avantages de JPEG 2000, on peut citer [3][16]:

- Un rapport débit/distorsion supérieur par rapport à JPEG surtout dans les fort taux de compression.
- Une organisation progressive du train binaire permettant de reconstruire l'image à n'importe quel instant (transmission progressive).
- La possibilité de sélections, des régions d'intérêt en vue de les compresser avec différents taux.
- La possibilité de codage avec au sans pertes.

3.3 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 1.5). Le principe de l'arbre de zéro proposé par Shapiro en 1993, permet d'exploiter cette ressemblance (section 3.3.A).

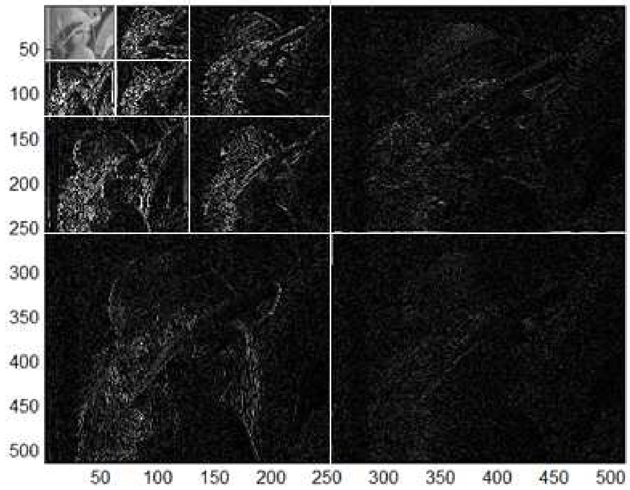


Figure 1.5: Exemple de transformation en sous-bande de l'image Lena, la ressemblance entres les sous bandes est remarquable.

A. Algorithme EZW de Shapiro

C'est le premier codeur en sous-bande par '*zerotree*' à avoir été introduit [17]. 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. Ce codeur est détaillé dans le chapitre 2.

L'algorithme EZW à été ensuite amélioré par l'algorithme SPIHT et SPECK.

B. Algorithme SPIHT

L'algorithme SPIHT [27], proposé par Said et Pearlman est une amélioration du schéma de codage par l'algorithme EZW. Il repose sur les mêmes concepts : codage progressif par plan de bits et utilisation des dépendances hiérarchiques entre les coefficients des différents sous-bandes. Cependant, un nouveau protocole de dépendance entre les coefficients est défini. Les coefficients de la sous-bande de plus basse fréquence (A2 en figure 1.6) sont regroupés par quatre, et pour chaque groupe, la descendance est comme suit :

Un des quatre coefficients (marqué par '*' en figure 1.6) n'admet pas des descendants, alors que les trois autres (en gris en figure 1.6) ont quatre descendants chacun. Pour les autres sous-bandes, comme dans le cas de l'EZW, chaque coefficient admet quatre descendants.

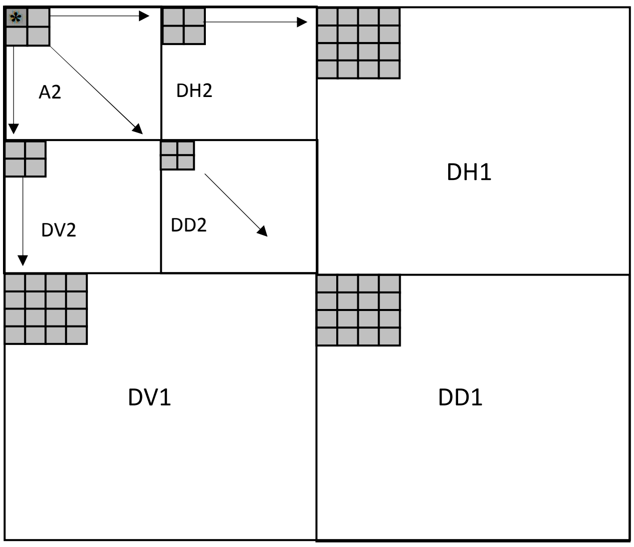


Figure 1.6 : Exemples de descendances parent-fils dans le cas SPIHT, le pixel désigné par (*) n'accepte pas de descendants.

Trois ensembles de coordonnées sont utilisés pour présenter la nouvelle méthode de codage :

D(i,j) : Ensemble de tous les descendants du pixel *(i,j)*.

O(i,j) : Ensemble de coordonnées des descendants directs du pixel *(i,j)*.

L(i,j) : Ensemble défini comme étant *D(i,j)-O(i,j)*.

Algorithme

L'algorithme est cependant plus sophistiqué : contrairement à l'algorithme EZW qui n'utilise qu'un seul ensemble décrivant la signifiance des coefficients, le SPIHT utilise trois listes :

- LIS : Liste des ensembles insignifiants.
- LIP : Liste des coefficients insignifiants.
- LSP : Liste des coefficients signifiants.

Les entrées de chacune des listes sont les coordonnées (i,j) dans la matrice image, qui représentent des pixels individuels dans le cas des listes LIP et LSP et l'ensemble de descendants D(i,j) ou L(i,j) dans le cas de la liste LIS.

Pour préciser la relation entre la comparaison des amplitudes à un seuil *n* et le message en bits envoyé, on utilise la fonction :

$$S_n(\Gamma) = \begin{cases} 1, si\ max_{(i,j)\in\Gamma}\{|c_{i,j}|\} \geq 2^n \\ 0, \quad sinon, \end{cases} \qquad (1.1)$$

Avec $c_{i,j}$ qui représente le coefficient de coordonnées (i, j).
$S_n(\Gamma)$ indique la signifiance de l'ensemble de coordonnées Γ.

Même principe que l'EZW, on utilise une passe pour la description des coefficients significatifs et une passe de raffinement. L'algorithme SPIHT se déroule comme suit [27] :

1. **Initialisation :**

   Calcul du seuil $n = \lfloor log_2(max_{(i,j)}\{|C_{i,j}|\}) \rfloor$

   $LSP = \phi$.

   $LIP =$ Les coordonnées de tous les coefficients de la sous-bande de plus basse fréquence.

   $LIS =$ Les coordonnées de sous-bandes de plus basse fréquence admettant des descendants.

2. **Passe de test de signifiance :**

   a. Pour chaque entrée (i, j) dans LIP faire :

   - Calculer $S_n(i,j)$.
   - Si $S_n(i,j)$=1 alors, déplacer (i,j) vers LSP et envoyer le signe de $c_{i,j}$.

   b. Pour chaque entrée dan LIS faire :

   b.1 si l'entrée est de type A alors,

   - Envoyer $S_n(D(i,j))$.
   - Si $S_n(D(i,j))=1$ alors
     - Pour chaque $(k,l) \in O(i,j)$ faire :
       - Envoyer $S_n(k,l)$.
       - *Si $S_n(k,l)=1$ alors : ajouter (k,l) à LSP et envoyer le signe de $c_{k,l}$.*
       - *Si $S_n(k,l)=0$ alors : ajouter (k,l) à la fin de LIP.*
     - Si L(i,j)=ϕ, alors déplacer (i,j) à la fin de LIS comme une entrée de type B, ensuite aller à l'étape b.2. Sinon retirer l'entrée (i,j) de LIS.

   b.2 Si l'entrée est de type B alors,

   - Envoyer $S_n(L(i,j))$.
   - Si $S_n(L(i,j))=1$ alors
     - Ajouter chaque *$(k,l) \in O(i,j)$ à la fin de LIS comme entrée de type A.*
     - Retirer (i,j) de LIS.

3. **Passe de raffinement :**

   Pour chaque entrée (i,j) dans la liste LIP, à l'exception de ceux trouvés par la dernière passe de test de signifiance, envoyer le n^{ieme} bit le plus signifiant de $|c_{i,j}|$.

4. **Réitération :** Décrémenter n par 1 et passer à l'étape 2.

Un exemple d'application de cet algorithme sur une matrice de test est présenté en **Annexe A.**

L'algorithme SPIHT permet de générer directement un flux binaire à la place des symboles de signifiance utilisés dans l'EZW. Cet algorithme, même dans sa version non codée entropiquement, dépasse les performances obtenues par l'EZW.

C. Algorithme SPECK

Offrant des performances comparables à l’algorithme SPIHT, l'algorithme SPECK [28] exploite des structures d'ensembles de coefficients non significatifs en blocs plutôt qu'en arbres. Ces structures de blocs permettent de s'affranchir efficacement de la non stationnarité (d’ordre 1) des coefficients en adaptant localement la statistique utilisée pour le codage.

Algorithme

Les coefficients sont initialement séparés en deux ensembles, l'un noté S contenant les coefficients de basses fréquences et l'autre, noté $\mathcal{T}$ contenant le reste des coefficients (cf. figure 1.7). 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).

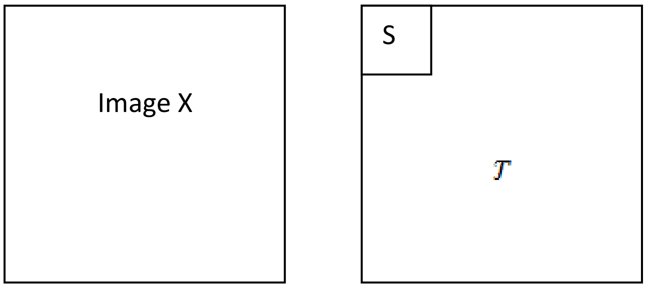


Figure 1.7 : Partitionnement de l’image X en deux sous-ensembles : S et le reste $\mathcal{T}$

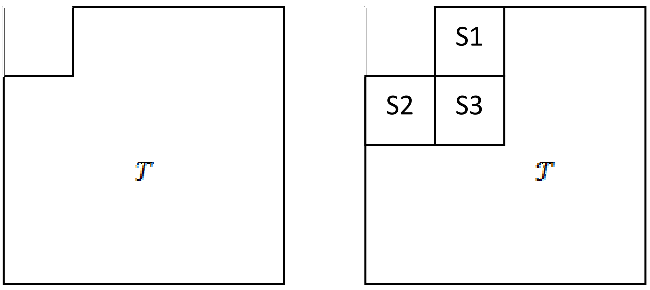


Figure 1.8 : Principe de partitionnement des ensembles $\mathcal{T}$

L'algorithme SPECK peut être résumé comme suit [28]:

1. **Initialisation :**
   - Partitionner l'image X en deux sous-ensembles : $S \equiv Racine$, et $\mathcal{T} = X - S$ (cf. figure 1.7).
   - Envoyer $n_{max} = \lfloor log_2(max_{(i,j)\in X}\{|C_{i,j}|\}) \rfloor$.
   - 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 $\in$ 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 $\mathcal{T}$:
   - Si $\mathcal{T}$ est signifiant, alors,
     Le bloc $\mathcal{T}$ est séparé en trois blocs correspondants au sous-bandes de plus basse-fréquences et un ensemble $\mathcal{T}$ 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 $\mathcal{T}$ 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 $n^{ième}$ 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. Un exemple d'application de cet algorithme pour une matrice de test est présenté dans l'annexe A.

D. Algorithme EZBC

Le principe de codage par l’algorithme EZBC (Embedded ZeroBlocks coding based on Context modeling) [52] 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 [29].

Ce codeur offre des performances comparables à EBCOT, et a également été adapté au codage vidéo sous le nom de MC-EZBC [53].

4 Autre programmes de compression

D’autres codeurs en sous bandes sont proposés dans la littérature. Citons, par exemple les algorithmes EQ (Estimation-Quantization) [54], SFQ (Space-Frequency Quantization) [55]. Ces codeurs ont l'avantage d'être simples et de complexité réduite, offrant des performances équivalentes ou légèrement supérieures à EBCOT et EZBC [29]. Ils ont par contre l'inconvénient de ne pas permettre un décodage progressif de l'image.

En plus de ces algorithmes, on trouve dans la littérature plusieurs versions (modifications) des algorithmes EZW, SPIHT et SPECK.

Enfin, il ya d’autres codeurs d’images qui ne sont pas basés sur la transformation en ondelettes permettant de donner des résultats satisfaisants, tels que la méthode LAR (Locally adaptative Resolution) [63] et les fractales [14].

5 Conclusion

Selon l’application recherchée, on peut choisir entre compression sans pertes d’information ou avec pertes. Plusieurs algorithmes pour les deux cas, sont proposés dans la littérature. Cependant, les algorithmes basés sur la transformation en ondelettes ont gagné un intérêt considérable ces dernières années. Cette dernière offre une plus grande finesse au niveau de l’analyse du signal et permet de mieux s’adapter aux propriétés locales de l’image.

Les codeurs d’images basés sur la transformation en ondelettes (JPEG2000, EZW, SPIHT, SPECK …etc.), permettent d’avoir des résultats meilleurs par rapport à ceux qui utilisent la DCT en plus de la propriété de la transmission progressive de l’image.

Chapitre 2 : Codeur EZW de Shapiro. Proposition d'une optimisation M-EZW

Résumé

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

- L'ajout de symboles dans l'étape de test de signifiance : Cette modification nous a permis de minimiser le nombre de symboles redondant d'une part et d'avoir une meilleure redistribution de l'entropie par rapport à l'algorithme EZW d'autre part. A cet effet, nous avons obtenu une quantité d'information inférieure par rapport à l'algorithme EZW.
- L'optimisation du codage des éléments des listes C et S par un codage en blocs : ceci nous a permis d'avoir un codage entropique des symboles plus efficace que dans le cas initial (codage de chaque symbole à part).

Dans ce chapitre, nous avons détaillé le principe de l'algorithme EZW ainsi que notre approche M-EZW. Une étude comparative entre les deux algorithmes est aussi présentée.

1 Introduction

Dans les images numériques, on trouve fréquemment des zones homogènes (fond d'objets) à forte corrélation spatiale et des anomalies (irrégularités), tels que les contours et les bords des objets. Ces derniers prennent une importance considérable dans la perception de l'image bien que leur contribution énergétique est plus faible que les zones homogènes [17].

Les codeurs traditionnels, tel que ceux utilisant la DCT [12], décomposent les images selon une grille qui a effectivement la même résolution temporelle et fréquentielle pour tous les coefficients dans la représentation (cf. section 2). L'information du bord est donc dispersée ; à cet effet, de nombreux coefficients non nuls sont nécessaires pour représenter les bords avec une bonne fidélité [17]. Toutefois, comme les bords représentent une énergie relativement insignifiante par rapport à l'image entière, les codeurs traditionnels, tels que ceux qui utilisent les DCT, ont une bonne efficacité aux moyens et hauts débits.

Cependant, à très faibles débits binaires, les techniques traditionnelles de codage, tels que le JPEG [10], tend à attribuer plus de bits aux zones homogènes et peu de bits pour représenter les anomalies. En conséquence, l'effet des blocs (artefacts) apparaissent souvent.

Les techniques basées sur la transformation en ondelettes ont montré des bons résultats à très faibles débits puisque les zones homogènes et les anomalies sont disponibles à chaque échelle. La difficulté majeure est dans l'exploitation et la gestion de la position des coefficients dus aux anomalies.

Le codeur mis en place par Shapiro appelé codeur EZW Embedded Zerotree Wavelet, permet aux codeurs d'utiliser efficacement la puissance de la représentation multi-résolutions de la transformation en ondelettes (cf. section 2) par la représentation efficace des positions des coefficients d'ondelettes représentant des anomalies.

2 Transformation en ondelettes discrète DWT

Les ondelettes, famille de fonctions déduites d'une même fonction appelée ondelette mère par opération de translation et dilatation, ont trouvé de par la puissance de leur théorie, des applications dans de nombreux domaines aussi variés que les mathématiques, le traitement du signal, la physique…etc.

La transformée en ondelettes est un outil d'analyse puissant permettant la représentation des fonctions dans une base bien localisée en espace et en fréquence offrant les avantages de l'analyse de Fourier et s'affranchissant des inconvénients du manque de localisation de cette dernière.

La transformée ou la décomposition en ondelettes, se présente comme une analyse temps-échelle et elle peut être envisagée comme une variation de l'analyse temps-fréquence.

2.1 Analyse temps-fréquence

La transformée de Fourier ou la représentation fréquentielle, décrit le signal comme une somme d'ondes expj2πft monochromatiques et de durée infinie [69] :

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

Le coefficient $X(f)$ mesure l'interaction, le degré de ressemblance, entre le signal et la fonction d'analyse expj2πft et indique donc la présence ou non dans le signal de la fréquence pure associée.

Le principal inconvénient de la transformée de Fourier est qu'elle ne permet pas une analyse du comportement local d'une fonction. De plus, cette transformation ne permet pas d'avoir une localisation temporelle du contenu fréquentiel d'un signal (cf. Figure 2.1.a).

Pour remédier à cette lacune, une solution intuitive consiste à restreindre l'existence du signal autour d'une date t, en le multipliant par une fenêtre g(t-τ) centrée sur cette date et d'extension temporelle limitée, avant d'en calculer la transformée de Fourier. Cette opération donne une transformée à deux composants, appelée 'transformée de Fourier à court terme' STFT (Short Time Fourier Transform), ou aussi transformée de Fourier à fenêtre glissante, approche proposée dans les années 40 par D. Gabor, [66][67] :

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

Avec $g\,(t\text{-}\tau)\, e^{j2\pi ft} = G_{f,\tau}(t)$ appelé *atome d'analyse*.

Les coefficients $STFT_x(f,t)$ indiquent alors la valeur de la fréquence f du signal x, à l'instant t. Ces coefficients ne peuvent cependant pas décrire le contenu du signal strictement à l'instant t et à la fréquence f, car l'atome d'analyse $G_{f,\tau}$ est caractérisé par ses extensions conjointes temporelles Δt et fréquentielle Δf, et mélange donc l'information contenue dans le signal, entre les dates $t \pm \Delta t/2$, dans la bande de fréquence $f \pm \Delta f/2$.

Une localisation temps-fréquence idéale, infiniment précise ($\Delta t=0$ et $\Delta f=0$) est interdite par le principe d'incertitude de Gabor-Heisenberg [65] :

$$\Delta t.\Delta f \geq 1/4\pi \qquad (2.3)$$

L'espace de représentation, le plan temps-fréquence, est alors pavé de cellules élémentaires, dont la forme ne varie ni avec le temps ni avec la fréquence (figure 2.1.b). La résolution fréquentielle de la STFT reste donc constante quelle que soit la fréquence analysée.

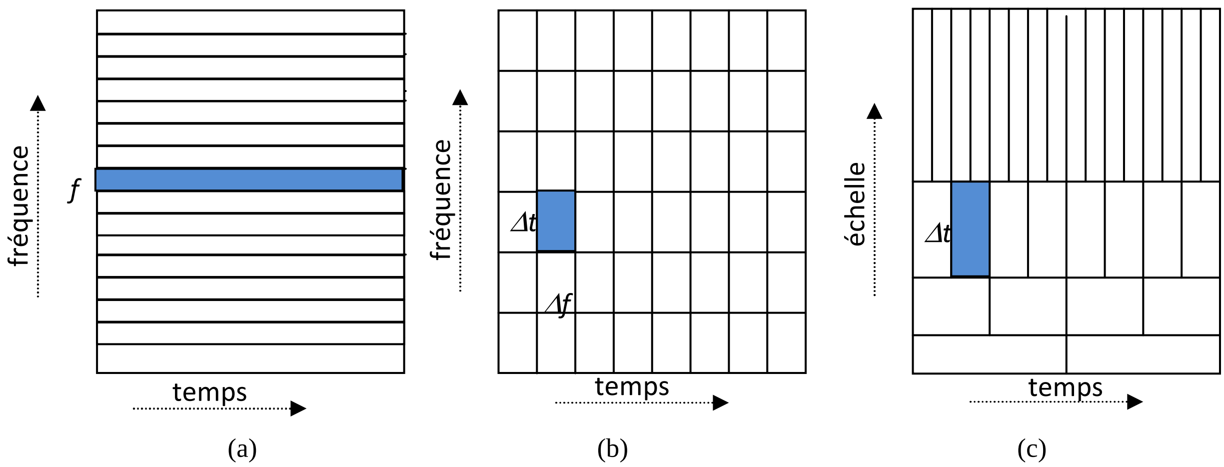


Figure. 2.1 Espaces de représentation : domaine de Fourier (a), espace temps-fréquence (b) et espace temps-échelle (c)

2.2 Analyse temps-échelle

L'analyse temps–échelle ou la décomposition en ondelettes est un nouvel outil qui permet également l'étude des signaux ou processus à comportement non-stationnaires. Les coefficients de cette nouvelle représentation sont encore obtenus en comparent (au moyen de produit scalaire), le signal à analyser avec une famille de fonctions $\psi_{a,t}$. Cette famille de fonctions est également obtenue à partir d'un même motif élémentaire appelée *ondelette-mère* ψ, par opération de translation et de dilatation [65] :

$$\psi_{a,t}(u) = \frac{1}{\sqrt{a}}\psi(\frac{u-t}{a}). \qquad (2.4)$$

Ces fonctions définissent l'espace bidimentionnel dans lequel l'information du signal x est représentée comme le plan temps-échelle. Le support temporel des ondelettes croit avec l'échelle d'analyse, tandis que le support fréquentiel décroît inversement [8] :

$$\Delta t_{\psi_a} = a\Delta t_{\psi} \quad \text{et} \quad \Delta f_{\psi_a} = \frac{\Delta f_{\psi}}{a} \qquad (2.5)$$

L'aire des logons qui couvrent le plan temps-échelle est ainsi préservée. Mais ceux-ci se déforment et s'allongent le long de l'axe des temps quand l'échelle d'analyse ***a*** augmente, cela produit le pavage représenté sur la figure 2.1.c.

Ce mécanisme de déformation des cellules contient l'ensemble de la richesse de l'analyse temps-échelle. Il permet, lorsqu'on sélectionne une petite échelle ***a***, d'effectuer une analyse du signal très localisée (très fine) en temps (obtenir une vue des détails) et à la fois, pour une grande échelle a, de réaliser une analyse sur un horizon beaucoup plus large (donner une vue d'ensemble).

L'analyse des hautes fréquences est réalisée avec une mauvaise résolution fréquentielle mais permet une localisation temporelle très précise. Au contraire, les basses fréquences sont scrutées avec une très forte résolution fréquentielle, mais, leurs instants d'occurrence sont très précis, [65].

La transformée en ondelettes dans le domaine continu peut être définie par [68]:

$$CWT_x(a,t) = \int x(t).\psi^*_{a,t}(a,t)du = \langle x, \psi_{a,t} \rangle. \qquad (2.6)$$

Dans le domaine discret, la transformation en ondelettes discrète décompose le signal en une pyramide de sous-signaux faisant apparaitre des signaux basses fréquences appelés approximés et des signaux hautes fréquences appelés détails, dont la définition de l'analyse multi-résolutions (AMR) [69]. Ces signaux sont obtenus par les relations :

$$A_j x = < x, \varphi_k^j >, \qquad (2.7)$$

$$D_j x = < f, \psi_k^j >. \qquad (2.8)$$

Avec :

$A_j x$, $D_j x$ représentent le signal approximé et le signal détail à l'échelle j, respectivement.
ψ_k^j représente la fonction d'ondelettes à l'échelle j.
φ_k^j représente la fonction d'échelle [65] à l'échelle j.

L'approximation du signal x(n) à la résolution r_j peut être calculée par la convolution de l'approximation à la résolution r_{j-1} avec un filtre discret h dont on connaît la réponse impulsionnelle :

$$A_j x = \sum_{n=-\infty}^{+\infty} \bar{h}(2k-n) A_{j-1} x \qquad (2.9)$$

Avec : h(n)=$< \varphi_0^{-1}, \varphi_n^0 >$

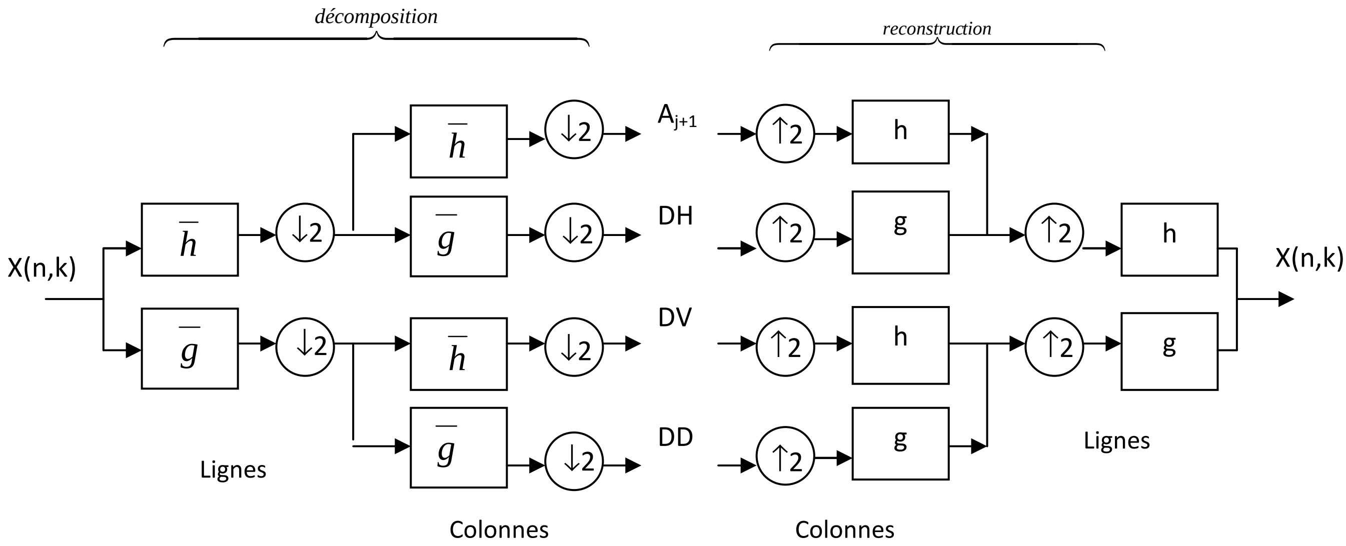
Les détails qui sont l''information perdue entre deux approximations successives peuvent s'écrire sous la forme :

$$D_j x = \sum_{n=-\infty}^{+\infty} \bar{g}(2k-n) A_{j-1} x \qquad (2.10)$$

Avec : $g(n) = (-1)^{1-n}h(1-n)$.

La transformée en ondelettes discrètes est donc une opération de convolution par les filtres discrets $\bar{h}(n)$ de type passe-bas (pour le calcul de l'approximation) et un filtre $\bar{g}(n)$ passe-haut (pour le calcul des détails), suivie d'une opération de décimation d'un coefficient sur deux.

Dans les applications de traitement d'image, l'AMR en deux dimensions peut être résumé par le schéma de la figure 2.2.



Avec :

↓2 : Décimation d'une ligne (colonne) sur deux

h : Convolution avec le filtre h (ou g)

↑2 : Interpolation des lignes (colonnes), c'est-à-dire mettre une colonne (ligne) de zéros entre deux.

Figure 2.2 Analyse multi-résolutions en deux dimensions

Les lignes de l'image originale (X) passent à travers deux filtres d'ondelettes, passe-bas et passe-haut (h et g), la sortie de chaque filtre subit une décimation (↓2) de un coefficient parmi deux. Ensuite, le même principe est répété sur les colonnes. Nous obtenons ainsi une image passe-bas, appelée image approximée (A_1), et trois autres sous-images, appelées Détail Horizontal (DH), Détail Vertical (DV) et Détail Diagonal (DD). La dimension de chacune de ces quatre images est le demi de la dimension de l'image originale (en raison de l'opération de décimation), dans les deux directions (horizontale et verticale). Ce processus de décomposition peut être répété successivement sur les images approximées A_j jusqu'à l'obtention de la plus petite image. La figure 2.3 illustre un exemple de décomposition d'une image en ondelettes à trois résolutions.

La reconstruction se fait à l'aide des Filtres Miroirs en Quadrature (QMF) ; cette opération est l'inverse de l'opération de décomposition, en remplaçant les filtres passe-bas et passe-haut par leurs filtres miroirs associés, et l'opération de décimation est remplacée par l'opération d'interpolation (↑2), qui consiste à ajouter un zéro entre chaque paire de coefficients.

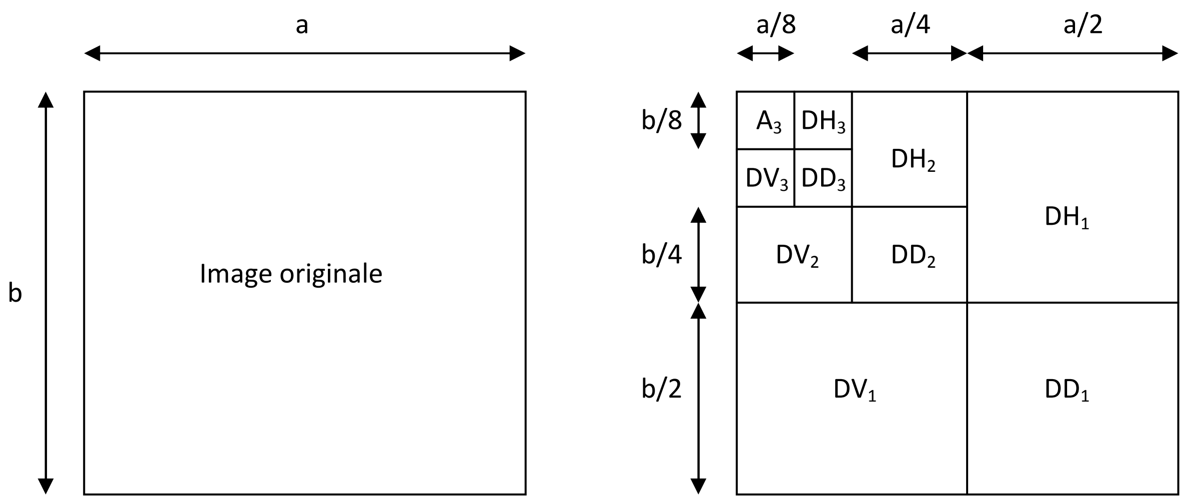


Figure 2.3 Décomposition en ondelettes à trois étapes

3 Algorithme EZW

3.1 Principe

Suite à la transformation par ondelettes discrètes à plusieurs niveaux, l'image est décomposée en sous-bandes (cf. figure 2.4). L'information est concentrée itération après itération dans l'image de plus faible fréquence (A3 pour l'exemple de la figure 2.4). Cette partie regroupe les coefficients d'approximations, issus du filtrage passe-bas de l'image. Les sous-images hautes fréquences (DH1-DH3, DV1-DV3 et DD1-DD3) issues du filtrage passe-haut, dont l'entropie est relativement faible occupent la majeure partie de l'image. De plus, la transformation en ondelettes offrant naturellement une représentation progressive de l'image, il est intéressant de conserver cette propriété lors du codage des sous-bandes.

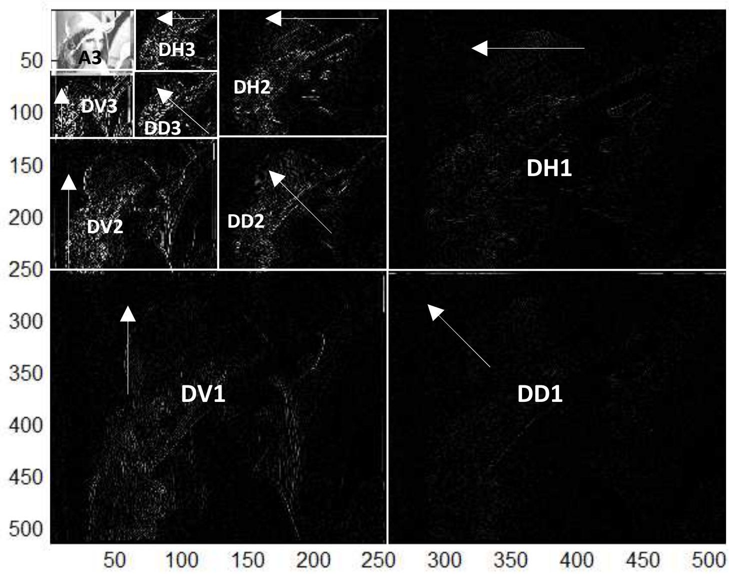


Figure 2.4 : Exemple de décomposition de l'image Lena par ondelettes en trois résolutions.

L'objectif du codeur de Shapiro consiste à exploiter un éventuel protocole de dépendance entre les coefficients des différentes sous-bandes afin d'aboutir à une notion d'arbres de zéros (zerotree).

La création d'un arbre de zéros part du principe que si un coefficient d'ondelette à une échelle plus grossière n'est pas significatif par rapport à un seuil T, alors il est fortement probable que tous les coefficients d'ondelettes aux échelles plus fines qui ont la même orientation (horizontale, verticale ou diagonale) et la même localisation spatiale soient aussi non significatifs par rapport à ce même seuil T. En pratique, la probabilité que ce phénomène survienne est très élevée [17].

Plus spécifiquement, dans un système à sous-bandes hiérarchique, à l'exception de la sous-bande de plus basse fréquence (A3 sur la figure 2.4), chaque coefficient dans une échelle donnée peut être relié avec un ensemble de coefficients dans l'échelle plus fine suivant qui à la même orientation spatiale (cf. figure 2.6). Le coefficient dans l'échelle grossière est appelé '*parent*' et tous les coefficients ayant la même orientation spatiale dans l'échelle plus fine suivant sont appelés '*fils*'.

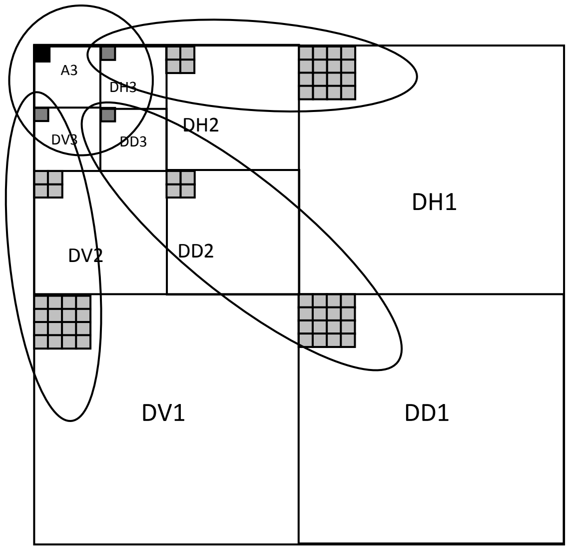


Figure 2.5 : Relation parent fils.

Pour un parent donné, l'ensemble des coefficients aux échelles plus fines qui ont la même orientation spatiale sont appelés '*descendants*'. De même, pour chaque fils à une échelle donnée, l'ensemble des coefficients aux échelles plus grossières qui ont la même orientation spatiale sont appelés 'ancêtres' [17].

Les relations parent-fils dans une structure pyramidale sont représentées en figure 2.5. A l'exception de la sous-bande de plus basse fréquence, chaque parent 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 (A3), chaque coefficient admet trois fils dans chacune des trois sous-bandes correspondant à la même résolution (cf. Figure 2.6).

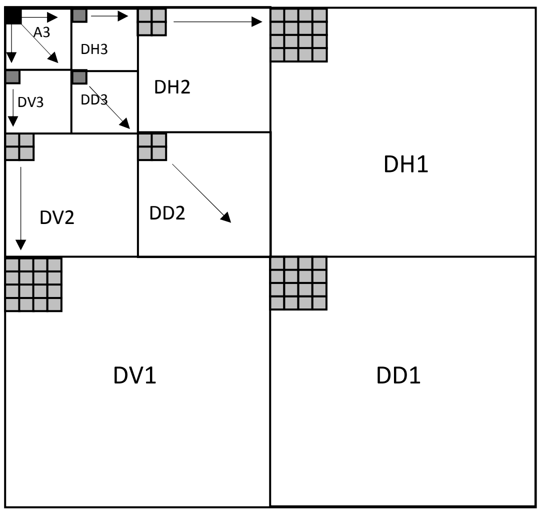


Figure 2.6 : Modèle de dépendances inter-bandes pour l'EZW 2D.

Un balayage des coefficients est défini tel qu'aucun fils n'est testé avant son parent. Ainsi, pour une transformation à N échelles, le balayage commence par la sous-bande de plus basse fréquence A_N, ensuite les autres sous bandes de la même échelle N (DH_N, DV_N et DD_N), et ainsi de suite. Ceci donne lieu au balayage en Z de la figure 2.7.

Si un ancêtre et tous ses descendants sont négligeables par rapport à un seuil T, alors, nous attribuons à l'ancêtre le code "zerotree" et les descendants ne seront pas codés. La structure des "zerotree" permet donc de coder la carte des coefficients significatifs par un très faible nombre de bits. La section suivante explique en détail le codage des coefficients d'ondelettes et la carte de signifiance obtenue.

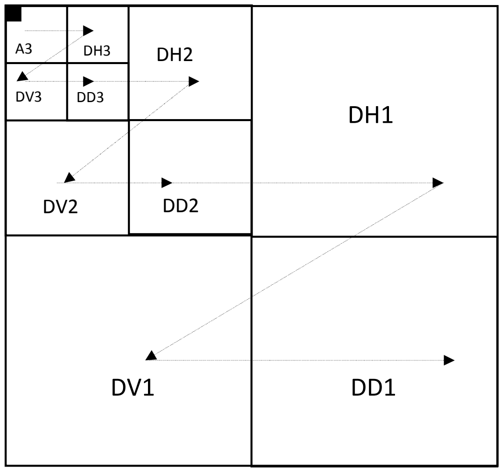


Figure 2.7: Ordre de balayage des coefficients des sous-bandes

3.2 Test de signifiance

Le codage des coefficients d'ondelettes passe par la détermination de deux listes de coefficients :

- Une liste contenant les coordonnées des coefficients qui ne sont pas significatifs par rapport au seuil actuel Tj. C'est la liste dominante D (cf. Table 2.1).
- Une liste contenant les valeurs des amplitudes des coefficients déjà trouvés significatifs. C'est la liste subordonnée S (cf. Table 2.1).

Considérons la matrice test de la figure 2.8, nous allons reprendre les différentes étapes de la méthode de Shapiro comme suit :

A. Initialisation

Après le calcul de la transformation en ondelettes discrètes de l'image, nous commençons par la définition d'un seuil relatif à la première passe de seuillage [35] :

$$T_0 = 2^{\lfloor \log_2(|C_{\max}|) \rfloor} \qquad (2.1)$$

Avec : $C_{max}=max(W_{i,j})$, $W_{i,j}$ étant le coefficient d'ondelette de coordonnées (i, j).

Si on prend le cas de la matrice test de la figure 2.8, le max des coefficients est égal à 63, on trouve T_0=32.

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

Figure 2.8 : Exemple de décomposition à trois résolutions d'une matrice de taille 8x8.

B. Test de signifiance

Nous parcourons les coefficients d'ondelettes selon l'ordre des coordonnées indiqué dans la liste *D* (chemin représenté sur la figure 2.7), de sorte que le codage zerotree soit le plus efficace possible. Pour chaque coefficient $W_{i,j}$, nous affectons un des quatre symboles *{P, N, Zt ou Zi}* en le comparant au seuil courant T_j ($T_j=T_0/2^j$) avec j le nombre d'itérations. Ces symboles serons placés dans une troisième liste *C* appelée *carte de signifiance* :

- Si $W_{i,j} \geq T_j$ (significatif et positif), le symbole 'P' est placé dans C. C'est le cas des coefficients {63, 49 et 47} de la matrice teste (cf. figure 2.8).
- Si $W_{i,j} \leq -T_j$ (significatif et négatif), le symbole 'N'est placé dans C. C'est le cas du coefficient {-34} de la matrice test (cf. figure 2.8).
- Si la valeur absolue du coefficient $W_{i,j}$ est inférieure au seuil T_j et ne posséde que des descendants négligeables, le symbole 'Zt' (zerotree) est placé dans C. Comme le cas du coefficient {23} de la matrice test de la figure 2.8. Les descendants de ce type de coefficients ne seront pas codés.
- Si la valeur absolue du coefficient $W_{i,j}$ est inférieure au seuil T_j et possède un ou plusieurs descendants non-négligeables devant T_j, le symbole 'Zi' (zéro isolé) est placé dans C. C'est le cas des coefficients {-31 et 14} (cf. figure 2.8).

Les coefficients non significatifs des dernières sous bandes, n'acceptant pas des descendants, et qui ne sont pas des fils d'un arbre de zéros, sont considérés aussi zerotree, c'est le cas des coefficients {7, -13, 3, 4, -1, -3 et 2}.

Le principe de test de signifiance est résumé dans l'organigramme de la figure 2.9.

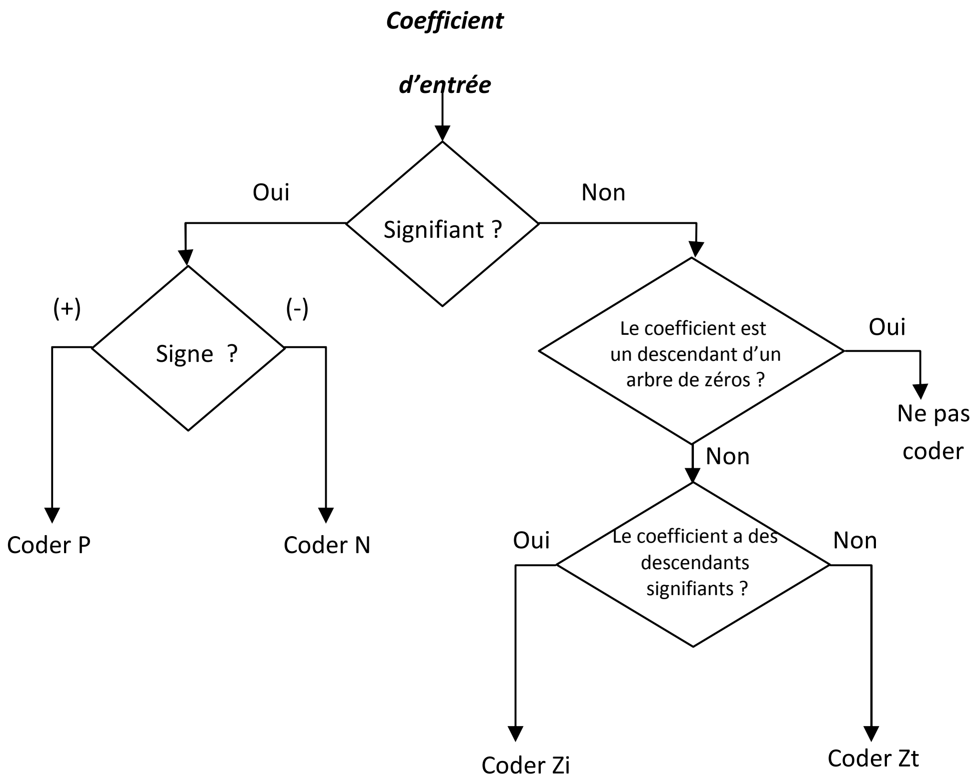


Figure 2.9 : Principe de test de signifiance des coefficients pour l'EZW

Lorsque les coefficients sont significatifs (deux premiers cas), leurs amplitudes sont placées dans la liste S et leurs coordonnées sont retirées de D. En ce qui concerne leur valeur dans l'image des coefficients, elle est mise à zéro pour ne pas être prise en compte à la passe suivante. Les coefficients de la liste S passent par une étape de quantification et raffinement (cf. section C) ensuite un codage entropique. Concernant les symboles de la carte de signifiance C, ils sont codés par un codage entropique adaptatif (cf. figure 2.10).

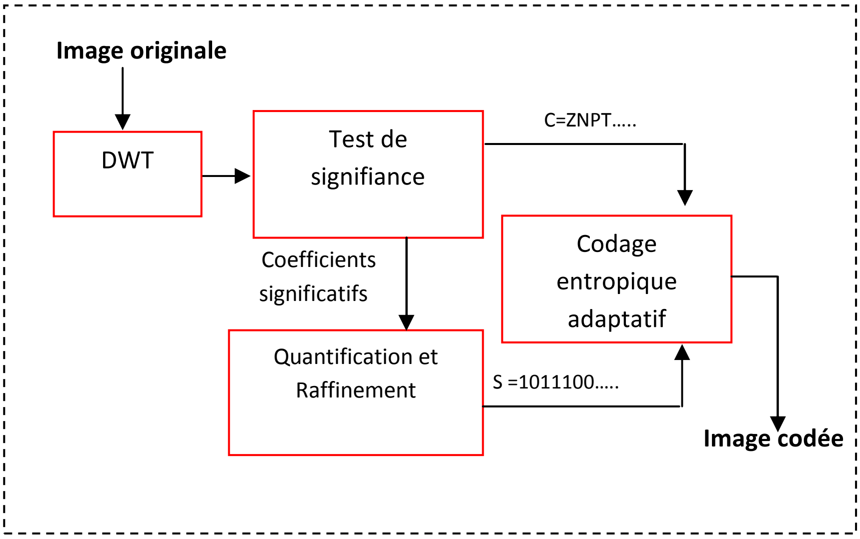


Figure 2.10 : Principe de l'algorithme EZW pour un cycle de compression

C. Quantification et Raffinement

La troisième étape s'attache au codage des éléments de la liste S. C'est dans cette liste que se trouvent les amplitudes des coefficients significatifs par rapport au seuil T_j et il faut les quantifier puis les coder. L'étape de quantification se déroule en deux temps [26] :

- Raffinement des coefficients qui étaient significatifs aux itérations précédentes.
- Quantification des nouveaux arrivants dans la liste S.

Nous utilisons pour cela un quantificateur scalaire uniforme défini par rapport au seuil Tj. La plage de quantification est comprise entre 0 et la valeur max des coefficients $2T_0$. Pour la première itération les valeurs significatives sont comprises dans l'intervalle [T0, 2T0]. La quantification consiste à affecter :

- le bit "0" pour les coefficients appartenant à la première moitié de cette intervalle c-à-d $[T_0, 3T_0/2[$ (cf. Figure 2.11.a). .
- et le bit "1" pour les coefficients appartenant à la deuxième moitié c-à-d $[3T_0/2, 2T_0]$ (cf. Figure 2.11.a).

Pour la deuxième itération, nous réalisons un raffinement des intervalles d'incertitude en les divisant par deux. Ainsi, le bit "0" est affecté aux coefficients appartenant à la première moitié de l'intervalle alors que le bit "1" est affecté à la seconde moitié. Pour les nouveaux arrivants, les coefficient dont l'intervalle est $[T_1, T_0[$, on effectue seulement un quantification par le bit "0" ou "1" selon les deux intervalles $[T_1, 3T_1/2[$ ou $[3T_1/2, T_0[$ successivement (cf.

Figure 2.11.b). Cette opération est répétée pour chaque seuil, en réalisant un raffinement des intervalles d'incertitude en les divisant par deux.

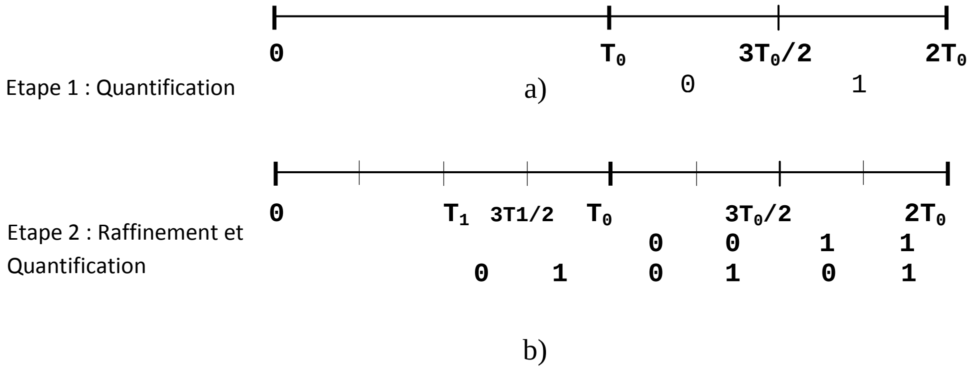


Figure 2.11 : Principe de quantification et raffinement

Dans la matrice de test de notre exemple (cf. figure 2.8), les coefficients trouvés significatifs pour la première itération sont : {63, -34, 49 et 47}. Pour le seuil T_0=32, ces coefficients seront quantifiés comme suit (cf. figure 2.12):

63 → '1', -34 → '0', 49 → '1' et 47 → '0'.

Nous obtenons la séquence de bits "1010"

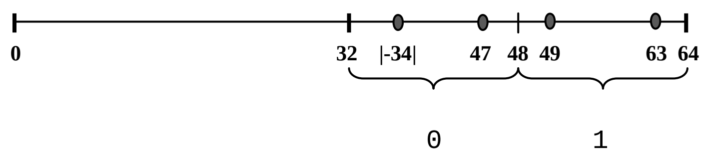


Figure 2.12 : Exemple de quantification des coefficients {63, -34, 49 et 47} pour T_0=32.

D. Réitération

Nous recommençons l'algorithme à partir de l'étape B sur le résidu de l'image en incrémentant *j* de un et en divisant le seuil T_j par deux. Nous itérons le processus jusqu'à l'obtention d'une qualité acceptable de l'image reconstruite (rapport de dégradation PSNR (cf. chapitre 1) acceptable) pour que le nombre de bits transférables demandés soit dépassé (obtention du taux de compression fixé à l'avance).

Considérons la matrice test de la figure 2.8 pour une première itération (T0=32). Le parcours des coefficients de la matrice selon l'algorithme ci-dessus est détaillé dans la Table 2.1.

Coordonnées Du Coefficient	**Symbole obtenu**	**Liste dominante D et liste subordonnée S**
		D = {(0,0)} S = ϕ
(0,0)	P	Ajout des fils de (0,0) dans D : D = {(0,0)F,(0,1),(1,0),(1,1)}. S = {63}
(0,1)	N	Ajout des fils de (0,1) dans D D = {(0,0)F,(0,1)F ,(1,0),(1,1),(0,2),(0,3),(1,2),(1,3)} S = {63,34}
(1,0)	Zi	Ajout des fils de (1,0) dans D : D = {(0,0)F,(0,1)F,(1,0)F,(1,1),(0,2),(0,3),(1,2),(1,3),(2,0),(2,1),(3,0),(3,1)}
(1,1)	Zt	
(0,2)	P	Les fils de (0,2) seront ajoutés à D. D = (0,0)F,(0,1)F,(1,0)F,(1,1),(0,2)F,(0,3),(1,2),(1,3),(2,0),(2,1),(3,0), (3,1),(0,4),(0,5), (1,4), (1,5),} S = {63,34,49)}
(0,3)	Zt	
(1,2)	Zt	
(1,3)	Zt	
(2,0)	Zt	
(2,1)	Zi	Ajout des fils de (2,1) dans D : D = (0,0)F,(0,1)F,(1,0)F,(1,1),(0,2)F,(0,3),(1,2),(1,3),(2,0),(2,1),(3,0), (3,1),(0,4),(0,5), (1,4), (1,5), (4,2),(4,3),(5,2),(5,3)}
(3,0)	Zt	
(3,1)	Zt	
(0,4)	Zt	
(0,5)	Zt	
(1,4)	Zt	
(1,5)	Zt	
(4,2)	Zt	
(4,3)	P	D = (0,0)F,(0,1)F,(1,0)F,(1,1),(0,2)F,(0,3),(1,2),(1,3),(2,0),(2,1),(3,0), (3,1),(0,4),(0,5), (1,4), (1,5), (4,2),(4,3)F,(5,2),(5,3)} S = {63,34,49,47}
(5,2)	Zt	
(5,3)	Zt	
		D = (0,0)F,(0,1)F,(1,0)F,(1,1),(0,2)F,(0,3),(1,2),(1,3),(2,0),(2,1),(3,0), (3,1),(0,4),(0,5), (1,4), (1,5), (4,2),(4,3)F,(5,2),(5,3)} S = {63,34,49,47}

Table 2.1 : Différentes étapes de l'algorithme EZW pour une itération (T_0=32), La notation '**F**' après les coordonnées dans la liste dominante indique que le coefficient est trouvé significatif dans cette itération et que son amplitude sera mise à zéro pour l'itération suivante.

Plusieurs modifications de l'algorithme EZW ont été proposées dans la littérature. Citons les exemples suivants : les auteurs dans [57] utilisent la régularité des ondelettes (régularité de Hôlder) pour obtenir un critère de classification des coefficients d'ondelettes dans l'arbre hiérarchique. Ce critère permet d'affiner la valeur du seuil de l'algorithme EZW. Dans [58], à la place des deux passes utilisées dans la méthode originale EZW, les auteurs utilisent une seule passe pour coder les coordonnées et l'amplitude des coefficients d'ondelettes. [56, 59 et 60] proposent de former des nouveaux symboles zerotree et de signifiance dans l'algorithme afin de minimiser la redondance de l'information et de générer moins de bits de données. Dans [56] la méthode consiste à regrouper plusieurs itérations de l'algorithme original en une seule, en comparant les coefficients d'ondelettes simultanément à plusieurs seuils différents. Au lieu de quatre symboles généré par la passe de signifiance, l'algorithme modifié dans [59] et [60] utilise huit symboles. Ces nouveaux symboles ajoutés réduisent les bits transmis et simplifient ainsi l'algorithme de codage. Dans [61] les auteurs proposent un algorithme de compression sans pertes des coefficients d'ondelettes utilisant la transformation en ondelettes entières. Ils utilisent un module de quantification et de codage différent de celui utilisé dans le codage zerotree classique. Dans [62] les auteurs présentent un codage zerotree adaptative des coefficients d'ondelettes. Ils introduisent des nouveaux symboles différents de ceux utilisés dans l'algorithme EZW de Shapiro, et modifient la méthode de codage de ces symboles. Les auteurs dans [52] proposent un codage emboîté basé sur les zero-blocks et une modulation qui prend en compte le contexte. Le codage zéro-blocks utilisé est basé sur le même principe que les algorithmes SPIHT et SPECK et la modulation du contexte est basée sur un partitionnement en quadtree adaptative.

4 Algorithme proposé M-EZW

Après avoir rappelé les différentes étapes de l'algorithme de Shapiro, nous allons présenter et expliquer notre contribution par rapport à l'algorithme de Shapiro. La différence entre l'algorithme que nous proposons M-EZW (Modified-EZW) et l'algorithme EZW de Shapiro réside dans le processus de test de signifiance des coefficients et du codage des symboles résultant :

- D'une part, nous introduisons de nouveaux symboles dans l'étape de test de signifiance pour minimiser les symboles redondants et permettre une meilleure redistribution de l'entropie (cf. section 5.1).

- D'autre part, Nous optimisons le codage des éléments des listes C et S par un codage en blocs (cf. section 5.2).

4.1 Test de signifiance

Après avoir calculé le seuil, la différence entre notre approche et l'algorithme EZW réside dans le test de signifiance des coefficients dans le cas où ces coefficients sont trouvés significatifs. Si les coefficients ne sont pas significatifs, nous suivons les mêmes étapes que celles pour l'EZW (cf. figure 2.13).

Si un coefficient est trouvé significatif, nous devons tester ensuite ses descendants :

- Si au moins un descendant est significatif, nous codons le coefficient selon l'algorithme de Shapiro, par le symbole 'P' s'il est positif ou 'N' sinon. C'est le cas des coefficients {63, -34 et 47} dans la matrice de test de la figure 2.5.
- Si tous les descendants sont non significatifs (étape colorée (grise) dans la figure 2.13), nous codons le coefficient selon l'algorithme que nous proposons. Soit par le symbole P_t s'il est positif ou par le symbole N_t s'il est négatif. Ces deux nouveaux symboles signifient que le coefficient est une racine d'un arbre dont seulement le premier coefficient (la racine) est significatif. Dans ce cas, il n'est pas utile de coder les descendants de cet arbre.

Le coefficient 49 dans la matrice de test (cf. figure 2.8), est donc codé P_t. Dans ce cas, il n'est plus utile de coder les descendants {7, -13, 3 et 4} de ce coefficient contrairement à l'algorithme EZW, où ils sont codés {Z_t Z_t Z_t Z_t} (cf. Table 2.1).

Cet algorithme de test de signifiance peut être résumé dans l'organigramme de la figure 2.13.

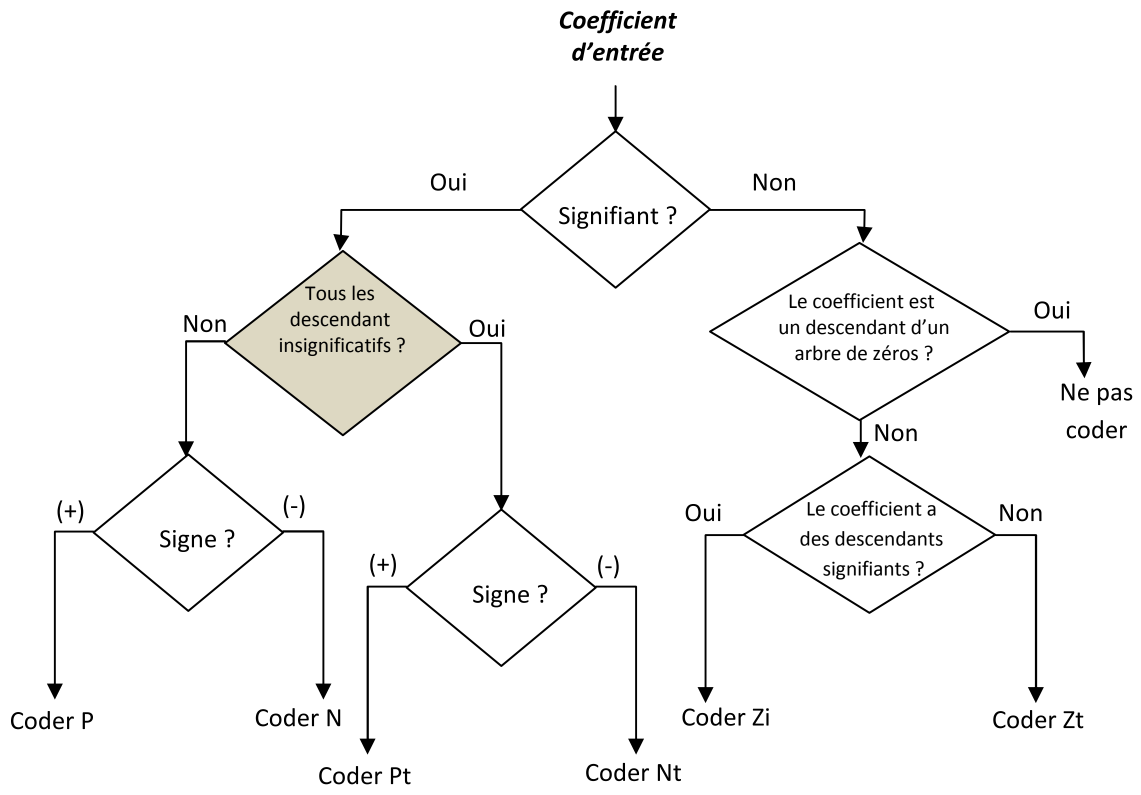


Figure 2.13 : Principe de test de signifiance des coefficients pour l'algorithme M-EZW.

En appliquant notre algorithme, M-EZW, sur la matrice de la figure 2.8, pour une première itération, nous obtenons les résultats présentés en Tables 2.2 et 2.3.

Dans le cas où le coefficient significatif ne se trouve pas à la racine, le symbole P_t (ou N_t) remplace l'ensemble des cinq symboles consécutifs ou non $\{P\ Z_t\ Z_t\ Z_t\ Z_t\}$ (ou $\{N\ Z_t\ Z_t\ Z_t\ Z_t\}$) dans le cas de l'EZW. C'est le cas du coefficient 49 de la matrice teste (cf. figure 2.8).

Si le coefficient significatif se trouve à la racine de la matrice représentant l'ancêtre et tous ses descendants, alors un symbole P_t (ou N_t) remplace l'ensemble des quatre symboles consécutifs $\{P\ Z_t\ Z_t\ Z_t\}$ (ou $\{N\ Z_t\ Z_t\ Z_t\}$) dans le cas de l'EZW. Ce cas n'apparaît pas dans l'exemple de la figure 2.8, mais nous donnons un autre exemple ou il apparaît, c'est le cas de la figure 2.14.

Les descendants de la racine '63' (cf. figure 2.14) sont tous insignifiants par rapport au seuil T0=32, la matrice est codée donc par les quatre symboles {P Zt Zt Zt} dans l'algorithme EZW et par le seul symbole {Pt} dans notre approche M-EZW.

Coordonnées	Symbole	Liste dominante D et liste subordonnée S
		D = {(0,0)} S = ∅
(0,0)	P	Ajout des fils de (0,0) dans D : D = {**(0,0)**F,(0,1),(1,0),(1,1)}. S = {63}
(0,1)	N	Ajout des fils de (0,1) dans D : D = {(0,0)F,**(0,1)**F ,(1,0),(1,1),(0,2),(0,3),(1,2),(1,3)} S = {63,34}
(1,0)	Zi	Ajout des fils de (1,0) dans D : D = {(0,0)F,(0,1)F,**(1,0)**F,(1,1),(0,2),(0,3),(1,2),(1,3),(2,0),(2,1),(3,0),(3,1)}
(1,1)	Zt	
(0,2)	**Pt**	Ajout des fils de (0,2) dans D avec la notation 'N' pour ne pas êtres testés. D = 0,0)F,(0,1)F,(1,0)F,(1,1),**(0,2)**F,(0,3),(1,2),(1,3),(2,0),(2,1),(3,0),(3,1), (0,4)N,(0,5)N, (1,4)N, (1,5)N,} S = {63,34,49)}
(0,3)	Zt	
(1,2)	Zt	
(1,3)	Zt	
(2,0)	Zt	
(2,1)	Zi	Ajout des fils de (2,1) dans D : D = (0,0)F,(0,1)F,(1,0)F,(1,1),(0,2)F,(0,3),(1,2),(1,3),(2,0),**(2,1)**,(3,0),(3,1), (0,4)N,(0,5)N, (1,4)N, (1,5)N, (4,2),(4,3),(5,2),(5,3)}
(3,0)	Zt	
(3,1)	Zt	
(4,2)	Zt	
(4,3)	P	D = (0,0)F,(0,1)F,(1,0)F,(1,1),(0,2)F,(0,3),(1,2),(1,3),(2,0),(2,1),(3,0),(3,1), (0,4)N,(0,5)N, (1,4)N, (1,5)N, (4,2),**(4,3)F**,(5,2),(5,3)} S = {63,34,49,47}
(5,2)	Zt	
(5,3)	Zt	
		D = (0,0)F,(0,1)F,(1,0)F,(1,1),(0,2)F,(0,3),(1,2),(1,3),(2,0),(2,1),(3,0),(3,1), (0,4)N,(0,5)N, (1,4)N, (1,5)N, (4,2),(4,3)F,(5,2),(5,3)} S = {63,34,49,47}

Table 2.2 : Différentes étapes de l'algorithme M-EZW pour une itération (T_0=32), La notation 'F' après les coordonnées dans la liste dominante indique que le coefficient est trouvé significatif dans cette itération et que son amplitude sera mise à zéro pour l'itération suivante. la notion '**N**', indique que le coefficient ne sera pas testé pour cette itérations.

63	-30	9	10	7	-13	12	7
-31	23	-14	-13	3	4	6	1
15	14	3	-12	5	-7	3	9
-9	-7	-14	8	4	-2	3	9
-5	9	-1	7	4	-6	-2	2
3	0	-3	2	2	-2	0	4
2	-3	6	4	3	6	3	6
5	11	5	6	0	3	-4	4

Figure 2.14 : 2^{eme} Exemple de décomposition à trois résolutions d'une matrice de taille 8x8, dont seulement la racine est significative par rapport au seuil T_0=32.

Les résultats finaux des deux algorithmes EZW et M-EZW appliqués aux matrices de test (cf. figure 2.8 et figure 2.14), sont résumés dans le tableau 3.

(A)	EZW	C	P N Zi Zt **P** Zt Zt Zt Zt Zi **Zt Zt Zt Zt** Zt Zt Zt P Zt Zt
		S	1 0 1 0
	M-EZW	C	P N Zi Zt **Pt** Zt Zt Zt Zt Zi Zt Zt Zt P Zt Zt
		S	1010
(B)	EZW	C	**P Zt Zt Zt**
		S	1
	M-EZW	C	**Pt**
		S	1

Table 2.3 : Résultats des algorithmes EZW et M-EZW appliqués aux matrices de test de la figure 2.8 (A) et figure 2.14 (B).

Dans l'exemple A (cf. Table 2.3), le symbole {Pt} codera le coefficient 49 et ses quatre descendants (en gras sur la figure 2.8) codés {P Zt Zt Zt Zt} dans l'algorithme EZW de Shapiro (cf. Table 2.3). Les descendants de la racine (63) sont négligeables par rapport au seuil T0=32 dans l'exemple B. La matrice de la figure 2.14 est donc codée {P Zt Zt Zt} par l'algorithme EZW et seulement par le symbole {Pt} par notre méthode M-EZW.

4.2 Comparaison quantitative

Nous allons maintenant faire une comparaison quantitative entre les deux algorithmes. Elle sera basée sur deux paramètres, le nombre de symboles obtenus et le nombre global de bits de ces symboles d'une part et le temps de calcul présenté par le nombre d'opérations de test d'autre part.

Ces comparaisons seront menées sur les exemples simples des deux matrices de test (cf. figures 2.8 et 2.14). Les comparaisons avec des images naturelles seront détaillées dans le chapitre 3.

A. Nombre de symboles

Il est clair que le nombre de symboles pour l'algorithme M-EZW est toujours inférieur à celui de l'algorithme EZW dans les deux cas de test (cf. Table 3). Cependant, en appliquant un codage binaire simple, les symboles de l'algorithme EZW seront codés sur 2 bits chacun (4 symboles donc 2 bits) et ceux de l'algorithme M-EZW seront codés sur 3 bits chacun (6 symboles donc 3 bits). Dans ce cas, pour le premier exemple (A) la carte de signifiance C est codée sur 40 bits (20 symboles x 2 bits= 40 bits) pour l'EZW et sur 48 bits (16 symboles x 3 bits = 48 bits) pour le M-EZW. Pour le deuxième exemple, la carte de signifiance est codée sur 8 bits (4 symboles x 2 bits=8 bits) pour l'EZW et sur 3 bits pour le M-EZW.

Donc, on peut constater que le nombre global des bits diffère pour les deux algorithmes selon la disposition des coefficients dans l'arbre hiérarchique. Ainsi, l'apparition des cas du même type que l'exemple (B) dans les images réelles est très probable et dépend du seuil et du type d'image.

Cependant, pour un codage entropique l'efficacité du codage peut être estimée par la quantité d'information contenue dans l'image. Pour cela, une étude détaillée de l'entropie des symboles obtenus par les deux algorithmes EZW et M-EZW appliqués sur des images réelles sera effectuée dans le chapitre 3.

B. Temps de calcul

Considérons le paramètre temps de calcul. Dans nos deux exemples nous constatons que :

- Si le symbole P_t (ou N_t) n'est pas à la racine de la matrice (cf. figure 2.8), il y aura 78 tests effectués par notre algorithme M-EZW conte 76 tests effectués par l'algorithme EZW classique (cf. Table 2.4.A).
- Si ces symboles (P_t ou N_t) se trouvent à la racine (cf. figure 2.14), les deux algorithmes ont les mêmes performances (cf. Table 2.4.B).

	(A)		(B)	
	EZW	**M-EZW**	**EZW**	**M-EZW**
NOMBRE DE SYMBOLES	20	16	4	1
NOMBRE D'OPERATIONS DE TEST	76	78	64	64

Table 2.4 : Nombre d'opérations de calcul de la liste D pour les deux exemples de la figure 2.8 (A) et de la figure 2.14 (B).

Un autre paramètre qu'il faut prendre en compte dans l'opération de codage est le nombre de symboles à coder. Comme dans tous les cas, le nombre total des symboles de notre algorithme M-EZW est inférieur à celui de l'algorithme EZW (cf. Table 2.3 et 2.4) et par conséquent, le temps de codage de ces symboles pour le M-EZW sera plus faible.

En tenant compte du nombre d'opérations de test et du temps de codage, on peut constater que notre approche nécessite des durées de calcul équivalentes à celles de l'algorithme EZW.

4.3 Procédure de codage des symboles de signifiance

Dans l'algorithme EZW de Shapiro, la carte de signifiance C est composée de quatre symboles {P, N, Zi et Zt}. Chacun de ces symboles est codé sur 2 bits binaires, à l'aide d'un codage entropique avant d'être transmis. Même principe pour l'algorithme M-EZW où les six symboles {P, N, Pt, Nt, Zi et Zt} sont codés en binaire sur 3 bits. Cependant, on peut aussi réduire des fois le taux de compression grâce au regroupement de plusieurs symboles [30].

Notre méthode consiste donc à un regroupement binaire des bits à coder (juxtaposition de symboles) des éléments de la carte de signifiance C (cf. figure 2.15.a) avant d'effectuer un codage entropique.

En ce qui concerne la liste signifiante S, un regroupement des bits obtenus par le raffinement des coefficients significatifs est aussi effectué (cf. figure 2.15.b), suivi par un codage entropique.

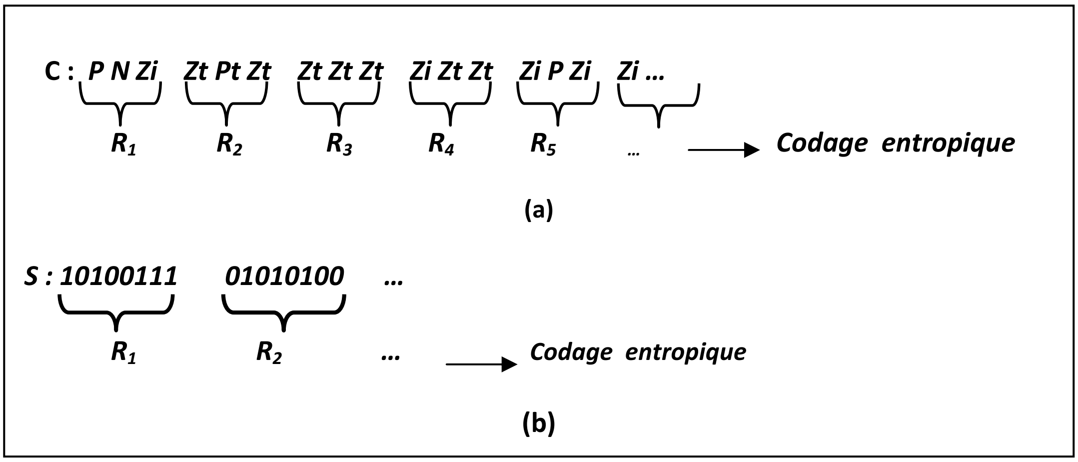


Figure 2.15 : Exemple de regroupement binaire des éléments de la liste D en 9 bits (a), et des bits de la liste S en 8 bits (b).

Le nombre de bits de regroupement des listes C et S ainsi que l'effet de ce regroupement sur la quantité d'information (l'entropie) sera étudié dans le chapitre 3.

5 Conclusion

Dans ce chapitre, nous avons développé un algorithme de compression d'images le M-EZW, basé sur le même principe que l'algorithme EZW de Shapiro. Cet algorithme est basé sur l'utilisation de six symboles au lieu de quatre symboles de signifiance utilisés par l'algorithme EZW pour minimiser le nombre de symboles redondants d'une part, et le regroupement binaire de l'information avant le codage arithmétique pour une meilleure optimisation de la quantité d'information, d'autre part. De plus, notre approche est réalisée avec un temps de calcul comparable à celui de l'EZW original.

Chapitre 3 : Résultats et Discussions

Résumé

On commence ce chapitre par un rappel des différents paramètres de test et de validations utilisés dans notre travail (cf. section 2).

Pour valider notre approche, nous avons d'abord testé l'effet de l'ajout de symboles dans l'étape de test de signifiance en section 5.1. Ensuite, l'effet du codage en blocs des éléments des listes S et D est étudié en section 5.2. Enfin, nous avons vérifié l'amélioration par ces deux propositions utilisées conjointement sur différentes images de test. Une comparaison avec les algorithmes actuels de compression d'images fixes est présentée en section 5.3.

1 Introduction

Après l'étude de l'algorithme EZW et la proposition d'une optimisation de ce dernier, nommée M-EZW (cf. chapitre 2); Nous effectuons dans ce chapitre une étude détaillée des résultats obtenus par les deux algorithmes (EZW er M-EZW) 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 et l'algorithme SPIHT est ainsi 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 (cf. section 2.2), le rapport de dégradation (cf. section 2.3) et la rapidité de compression et de la décompression (section 2.4). Autres paramètres très utilisés dans le domaine de la compression, la quantité d'information et l'entropie (cf. section 2.1). 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.

2.1 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 [33] :

$$Q_x(n_i) = -log(p(n_i)) \qquad (3.1)$$

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

$$H(x) = E\{Q_x\} = E\{-\log(p(x))\} \qquad (3.2)$$

Considérons une source d'information S qui émet des messages constitués de symboles de l'alphabet : $A = \{\lambda_1, \dots, \lambda_n\}$ 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 [6] :

$$H(s) = -\sum_{i=1}^{n} p_i . \log(p_i) \qquad (3.3)$$

Shannon [33] 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.

2.2 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 :

$$RC(\%) = \frac{nombre\ de\ bits\ codés}{nombre\ de\ bits\ de\ l'image\ originale} \times 100 \qquad (3.4)$$

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 :

$$RC(bpp) = \frac{nombre\ de\ bits\ codés}{taille\ de\ l'image(nombre\ de\ pixels)} \qquad (3.5)$$

1.1 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 [7] :

$$EQM = \frac{1}{n \times m} \sum_{i=1}^{n} \sum_{j=1}^{m} \left(f(i,j) - \hat{f}(i,j)\right)^2 \qquad (3.6)$$

$n \times m$: 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] :

$$SNR = 10log_{10}\left(\frac{v}{EQM}\right) \qquad (3.7)$$

Où v 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 :

$$PSNR(dB) = 10log_{10}\left(\frac{\max(f_i)^2}{EQM}\right) \qquad (3.8)$$

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

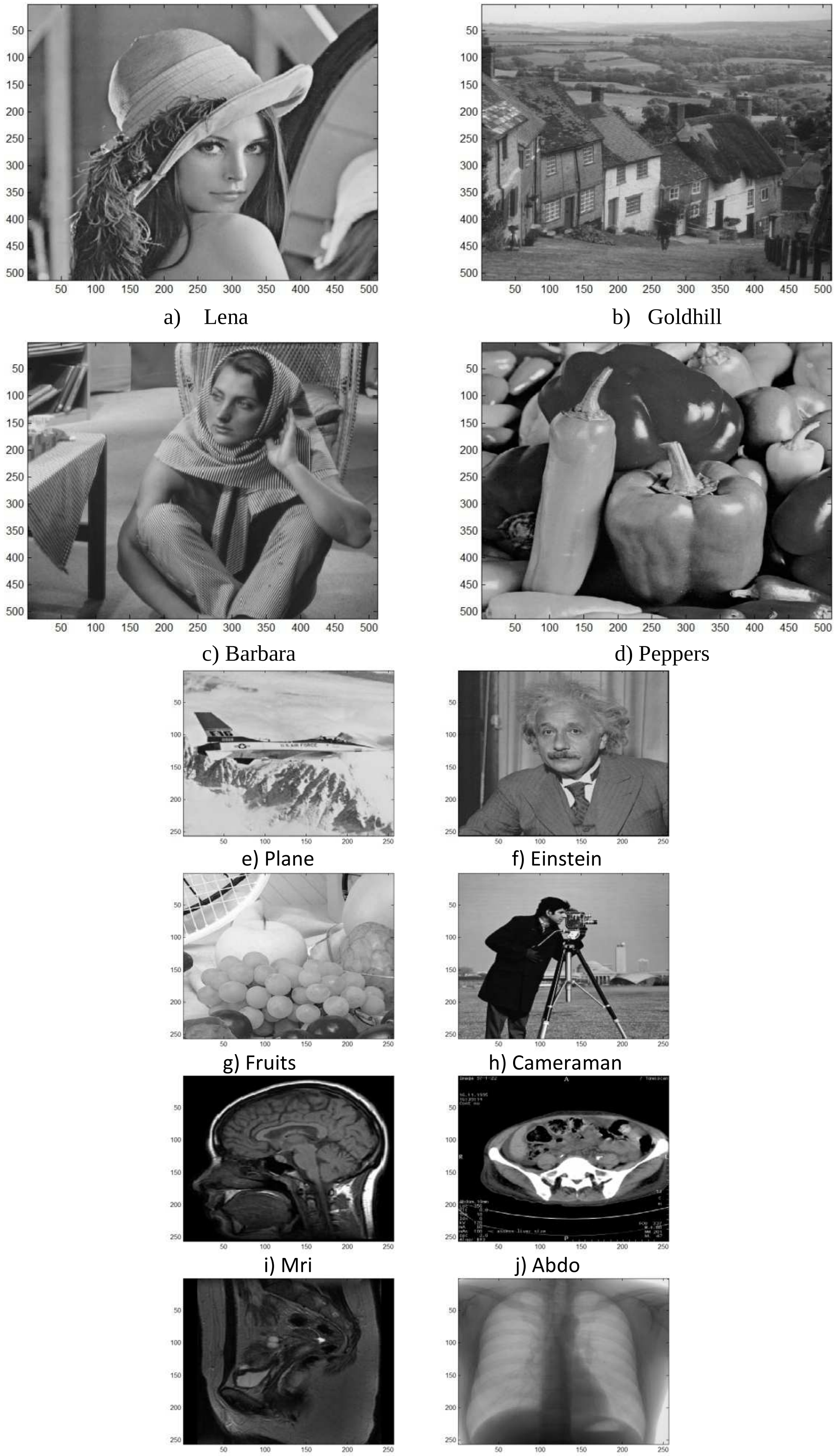
2.3 Temps de calcul

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 [8]. Cette qualité sera cependant moins cruciale dans des applications visant l'archivage de données.

3 Images de test

Pour notre application, nous avons utilisé des images testes en niveaux de gris (codées sur 8 bits) de différents types et de différentes tailles (cf. figure 3.1) :

- Les images Lena (a), Goldhill (b), Barbara (c) et Peppers (d) de taille 512x512 ;
- Les images Plane (e), Einstein (f), Fruits (g), Cameraman (h), Mri (i), Abdo (j), Angio (k) et Chest (l), de taille 256x256 pixels.



a) Lena b) Goldhill

c) Barbara d) Peppers

e) Plane f) Einstein

g) Fruits h) Cameraman

i) Mri j) Abdo

Figure 3.1 : Images testes utilisées

4 Algorithme EZW

L'architecture globale de l'algorithme EZW peut être résumée par l'organigramme de la figure 3.2. L'image originale est décomposée premièrement en ondelettes à plusieurs niveaux selon la taille de l'image et la longueur des filtres d'ondelettes (section 4.2). Les coefficients d'ondelettes obtenus passent ensuite à travers un processus de test de signifiance (cf. chapitre 2). Ce dernier permet d'obtenir deux listes de données. La première liste C contient quatre symboles différents {P, N, Zt, Zi} représentant la signifiance des coefficients d'ondelettes. La deuxième liste S, contient les amplitudes des coefficients significatifs par rapport au seuil courant. Cette dernière passe ensuite à travers un processus de quantification et de raffinement (cf. chapitre 2). Les deux listes sont ensuite codées par un codage arithmétique adaptatif. Le codage est terminé quand le taux de compression désiré est atteint, sinon, on continue le test du reste des coefficients. Si tous les coefficients sont testés on divise le seuil par deux et on recommence l'étape de test de signifiance.

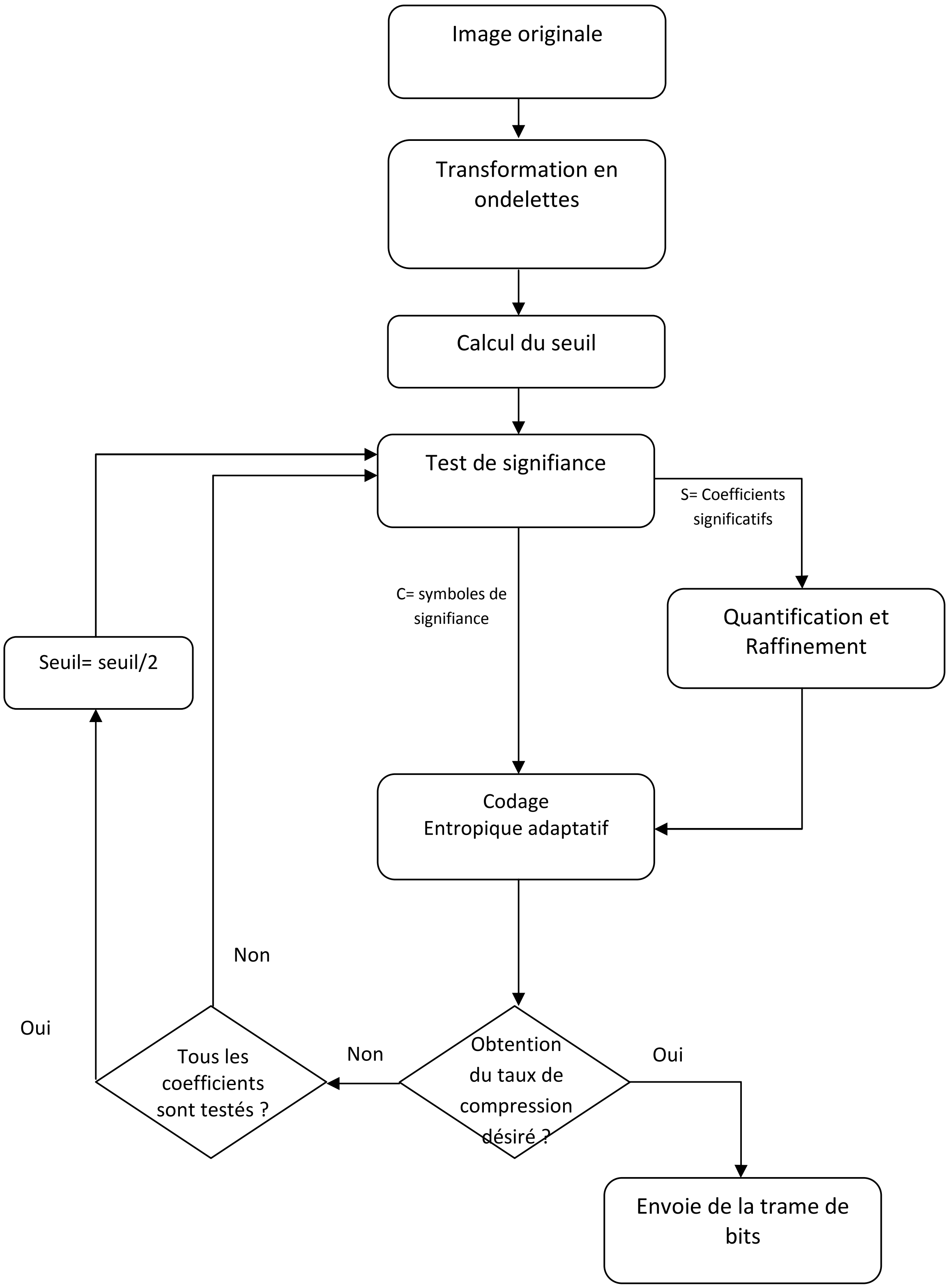


Figure 3.2 : Organigramme du codage EZW

4.1 Principe de codage progressif

Un codage progressif consiste en la possibilité de reconstruire l'image à n'importe quel instant. Pour être progressif, l'algorithme EZW 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 (cf. figure 3.2).

Le récepteur (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 carte de signifiance C par les valeurs quantifiées indiquées par la liste S. Par l'application de la transformée en ondelettes inverse sur la matrice (l'image) obtenue, on peut avoir une image reconstruire à n'importe quel instant.

4.2 Choix de l'ondelette et du niveau de décomposition

La qualité de l'image compressée par ondelettes dépend essentiellement du choix de l'ondelette. En effet, ce choix est basé sur le fait de trouver des bancs de filtres qui compactent le maximum d'énergie, tout en garantissant une bonne qualité de reconstruction de l'image [42]. Dans la littérature, plusieurs études réalisées sur des bases d'ondelettes réelles et entières ont abouti aux conclusions suivantes [43] :

- Pour la compression avec pertes, les transformations réelles [44] donnent des résultats meilleurs que ceux donnés par les transformations entières [45], [46].
- La transformation en ondelettes entières est préférée dans le cas d'une compression sans pertes, alors que pour une compression avec pertes, on utilise plutôt une transformation en ondelettes réelles.
- Pour la compression avec pertes, la base bi-orthogonale de Daubechies [47] notée « 9/7 filter bank » donne généralement les meilleurs résultats.
- Pour la compression sans pertes, des études ont montré que les transformations en ondelettes entières 5/3 et 2/6 demeurent performantes pour la majorité des images et la transformation 5/11-C donne de bons résultats pour des images de paysage [48],[49].

Un autre critère très important dans la compression est le niveau de décomposition de la transformée par ondelettes. Des études ont montré qu'il suffit d'opérer, aussi bien pour les ondelettes orthogonales que les ondelettes bi-orthogonales, cinq niveaux de décomposition pour compacter le maximum d'énergie dans la sous bande d'approximation [50][51].

5 Algorithme proposé M-EZW

La différence entre l'algorithme M-EZW, que nous proposons et celui de Shapiro réside dans le processus de test de signifiance des coefficients et du codage de ces derniers (étapes en gris dans la figure 3.3).

Dans l'étape de test de signifiance, on utilise six symboles au lieu de quatre employés dans l'algorithme de Shapiro (section 5.1). Dans l'étape de codage, on a proposé d'utiliser un regroupement binaire des bits à coder avant l'application du codage entropique (section 5.2).

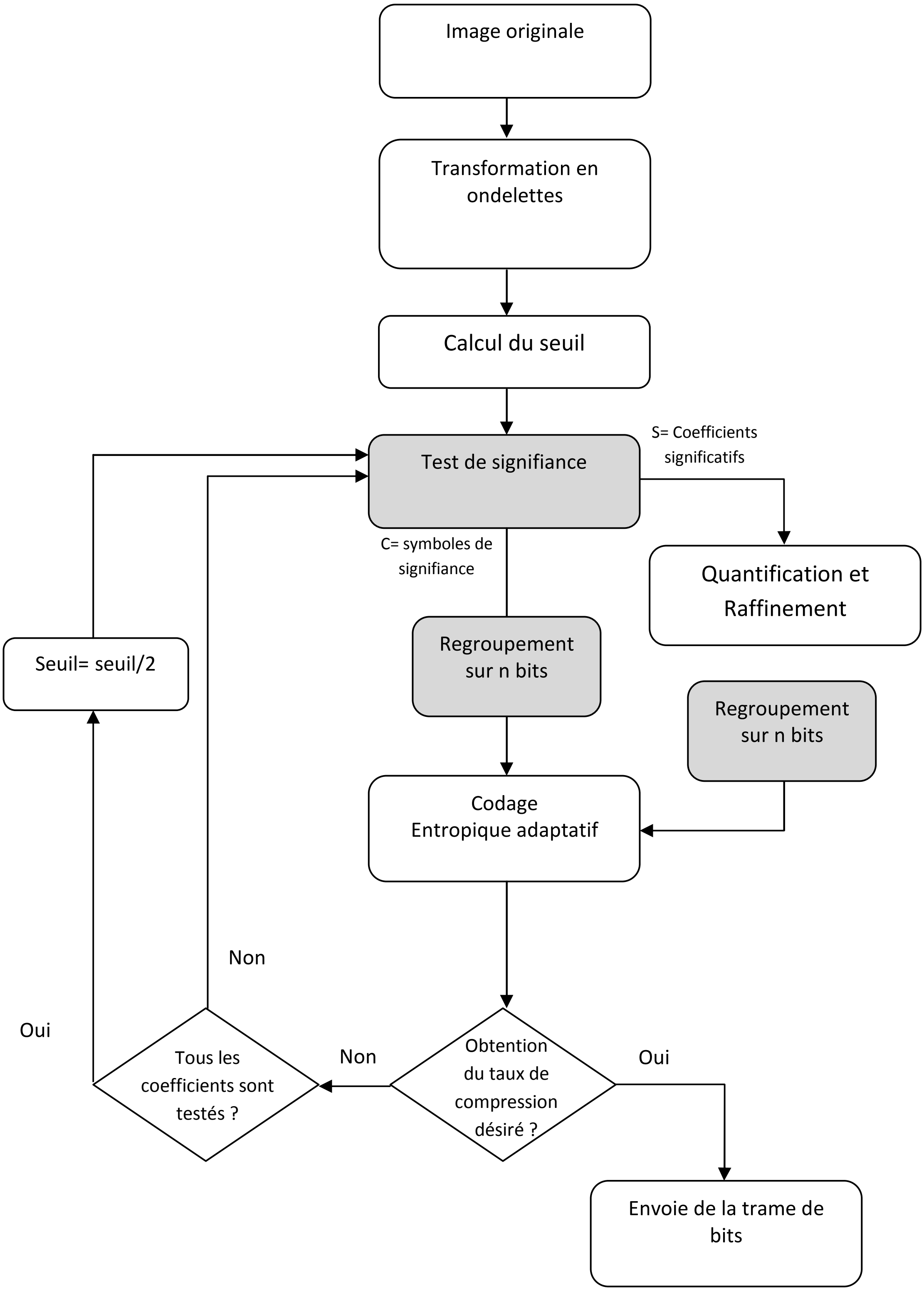


Figure 3.3 : Organigramme du codage M-EZW proposé

5.1 Test de signifiance

Nous avons étudié l'effet de l'ajout des deux symboles ('Pt' et 'Nt') dans l'étape de test de signifiance de différentes manières :

- Premièrement, on a calculé les fréquences d'apparitions des différents symboles de signifiance ainsi que le nombre total de tous ces symboles pour différents seuils T. Ceci nous a permis de calculer le nombre de bits alloué à l'ensemble des symboles. Ce dernier est calculé par une simple multiplication du nombre de symboles par le nombre de bits alloué à chaque symbole (2 bits pour le cas de l'algorithme EZW et 3 bits pour le cas de l'algorithme M-EZW) (cf. Tables 3.1 – 3.3).
- Deuxièmement, la probabilité d'apparition et l'entropie des différents symboles sont calculées pour les deux algorithmes (EZW et M-EZW) et pour différents seuils. Ceci est effectué dans le but d'étudier l'effet de l'ajout des deux symboles supplémentaires sur la table de distribution des probabilités et sur les entropies des symboles (cf. tables 3.4 – 3.6 et 3.7 – 3.9). La redistribution de la table des probabilités et des entropies nous permet de tirer un pré-jugement sur l'efficacité du codage.
- Enfin, pour estimer l'efficacité de notre algorithme d'une façon plus claire, nous avons calculé l'entropie en bit par pixel (bpp) de l'ensemble des symboles de signifiance générés par les deux algorithmes EZW et M-EZW (cf. tables 3.7 – 3.9). Cette entropie est calculée comme suit :

$$E = \frac{\sum Entropie\ de\ chaque\ symbole \times sa\ fréquence\ d'apparition}{Taille\ de\ l'image}$$

Notre étude dans cette section s'est concentrée sur trois images de test {Lena, Barabara et Goldhill} de taille 512*512.

	Image Lena													
	M-EZW								EZW					
Seuil	Nombre total de symboles	Nombre total de bits	Nombre de symboles						Nombre total de symboles	Nombre total de bits	Nombre de symboles			
			Zt	Zi	P	N	Pt	Nt			Zt	Zi	P	N
4096	256	**768**	129	0	0	0	127	0	637	1274	510	0	127	0
2048	512	**1536**	275	0	0	0	237	0	1223	2446	986	0	237	0
1024	804	**2412**	525	11	1	0	261	6	1617	3234	1338	11	262	6
512	1415	**4245**	922	118	5	0	313	57	2640	5280	2147	118	318	57
256	2815	**8445**	1724	415	27	7	448	194	5128	10256	4037	415	475	201
128	6273	**18819**	3615	1175	74	56	777	576	11430	22860	8772	1175	851	632
64	14327	**42981**	7841	2873	235	264	1665	1449	26528	53056	20042	2873	1900	1713
32	30694	**92082**	16372	6335	803	965	3240	2979	55315	110630	40993	6335	4043	3944
16	61583	**184749**	32386	12752	2814	3047	5460	5124	103664	207328	74467	12752	8274	8171
8	123507	**370521**	64891	25665	8380	8653	8064	7854	186924	373848	128308	25665	16444	16507
4	264775	794325	136999	54366	25843	26113	10844	10610	350336	**700672**	222560	54366	36687	36723
2	465255	1395765	232055	94458	57166	56784	12514	12278	564168	**1128336**	330968	94458	69680	69062

Table 3.1 : Nombre total des symboles et fréquences d'apparition de chaque symbole obtenus par les deux algorithmes EZW et M-EZW appliqués sur l'image **Lena** 512*512 pour différents seuils T.

	Image Barbara													
	M-EZW								EZW					
Seuil	Nombre total de symboles	Nombre total de bits	Nombre de symboles						Nombre total de symboles	Nombre total de bits	Nombre de symboles			
			Zt	Zi	P	N	Pt	Nt			Zt	Zi	P	N
4096	256	**768**	173	0	0	0	83	0	505	1010	422	0	83	0
2048	512	**1536**	278	0	0	0	234	0	1214	2428	980	0	234	0
1024	807	**2421**	527	12	1	0	261	6	1621	3242	1341	12	262	6
512	1303	**3909**	879	85	2	2	298	37	2389	4778	1965	85	300	39
256	2508	**7524**	1556	335	21	10	418	168	4598	9196	3646	335	439	178
128	6742	**20226**	3854	1253	132	106	834	563	12076	24152	9188	1253	966	669
64	25487	**76461**	12956	5357	1798	1729	1958	1689	39821	79642	27290	5357	3756	3418
32	60776	182328	30005	12652	5303	5259	3940	3617	90750	**181500**	59979	12652	9243	8876
16	116393	349179	57342	24199	11187	10999	6516	6150	166803	**333606**	107752	24199	17703	17149
8	197657	592971	97780	41111	20003	20175	9496	9092	271755	**543510**	171878	41111	29499	29267
4	324001	972003	162436	67437	34869	35051	12263	11945	420579	**841158**	259014	67437	47132	46996
2	498269	1494807	248402	103945	58662	59286	14176	13798	609911	**1219822**	360044	103945	72838	73084

Table 3.2 : Nombre total des symboles et fréquences d'apparition de chaque symbole obtenus par les deux algorithmes EZW et M-EZW appliqués sur l'image **Barbara** 512*512 pour différents seuils T.

	Image Goldhill													
	M-EZW								EZW					
Seuil	Nombre total de symboles	Nombre total de bits	Nombre de symboles						Nombre total de symboles	Nombre total de bits	Nombre de symboles			
			Zt	Zi	P	N	Pt	Nt			Zt	Zi	P	N
4096	256	**768**	187	0	0	0	69	0	463	926	394	0	69	0
2048	512	**1536**	282	0	0	0	230	0	1202	2404	972	0	230	0
1024	792	**2376**	520	8	0	0	258	6	1592	3184	1320	8	258	6
512	1277	**3831**	865	79	1	0	283	49	2349	4698	1937	79	284	49
256	2357	**7071**	1478	304	14	5	390	166	4325	8650	3446	304	404	171
128	5676	**17028**	3265	1009	67	68	759	508	10488	20976	8077	1009	826	576
64	15793	**47379**	8575	3077	534	347	1775	1485	28577	57154	21359	3077	2309	1832
32	39209	**117627**	20846	7850	1741	1431	3766	3575	68317	136634	49954	7850	5507	5006
16	93725	**281175**	48680	18733	6424	5856	7063	6969	149597	299194	104552	18733	13487	12825
8	204125	612375	104581	40747	19336	18173	10743	10545	289021	**578042**	189477	40747	30079	28718
4	381137	1143411	191176	76868	44790	42708	13001	12594	483261	**966522**	293300	76868	57791	55302
2	576505	1729515	285833	118850	73280	70309	14372	13861	689181	**1378362**	398509	118850	87652	84170

Table 3.3 : Nombre total des symboles et fréquences d'apparition de chaque symbole obtenus par les deux algorithmes EZW et M-EZW appliqués sur l'image **Goldhill** 512*512 pour différents seuils T.

	Probabilité des symboles de l'image Lena									
	MEZW						EZW			
Symboles / Seuil	Zt	Zi	P	N	Pt	Nt	Zt	Zi	P	N
4096	0.5039	0	0	0	0.4961	0	0.8006	0	0.1994	0
2048	0.5371	0	0	0	0.4629	0	0.8062	0	0.1938	0
1024	0.6530	0.0137	0.0012	0	0.3246	0.0075	0.8275	0.0068	0.1620	0.0037
512	0.6516	0.0834	0.0035	0	0.2212	0.0403	0.8133	0.0447	0.1205	0.0216
256	0.6124	0.1474	0.0096	0.0025	0.1591	0.0689	0.7872	0.0809	0.0926	0.0392
128	0.5763	0.1873	0.0118	0.0089	0.1239	0.0918	0.7675	0.1028	0.0745	0.0553
64	0.5473	0.2005	0.0164	0.0184	0.1162	0.1011	0.7555	0.1083	0.0716	0.0646
32	0.5334	0.2064	0.0262	0.0314	0.1056	0.0971	0.7411	0.1145	0.0731	0.0713
16	0.5259	0.2071	0.0457	0.0495	0.0887	0.0832	0.7183	0.1230	0.0798	0.0788
8	0.5254	0.2078	0.0679	0.0701	0.0653	0.0636	0.6864	0.1373	0.0880	0.0883
4	0.5174	0.2053	0.0976	0.0986	0.0410	0.0401	0.6353	0.1552	0.1047	0.1048
2	0.4988	0.2030	0.1229	0.1220	0.0269	0.0264	0.5866	0.1674	0.1235	0.1224

Table 3.4 : Probabilités des différents symboles obtenus par les deux algorithmes EZW et M-EZW appliqués sur l'image **Lena** 512*512 pour différents seuils T.

	Probabilité des symboles de l'image Barbara									
	MEZW						EZW			
Symboles / Seuil	Zt	Zi	P	N	Pt	Nt	Zt	Zi	P	N
4096	0.6758	0	0	0	0.3242	0	0.8356	0	0.1644	0
2048	0.5430	0	0	0	0.4570	0	0.8072	0	0.1928	0
1024	0.6530	0.0149	0.0012	0	0.3234	0.0074	0.8273	0.0074	0.1616	0.0037
512	0.6746	0.0652	0.0015	0.0015	0.2287	0.0284	0.8225	0.0356	0.1256	0.0163
256	0.6204	0.1336	0.0084	0.0040	0.1667	0.0670	0.7930	0.0729	0.0955	0.0387
128	0.5716	0.1858	0.0196	0.0157	0.1237	0.0835	0.7608	0.1038	0.0800	0.0554
64	0.5083	0.2102	0.0705	0.0678	0.0768	0.0663	0.6853	0.1345	0.0943	0.0858
32	0.4937	0.2082	0.0873	0.0865	0.0648	0.0595	0.6609	0.1394	0.1019	0.0978
16	0.4927	0.2079	0.0961	0.0945	0.0560	0.0528	0.6460	0.1451	0.1061	0.1028
8	0.4947	0.2080	0.1012	0.1021	0.0480	0.0460	0.6325	0.1513	0.1085	0.1077
4	0.5013	0.2081	0.1076	0.1082	0.0378	0.0369	0.6159	0.1603	0.1121	0.1117
2	0.4985	0.2086	0.1177	0.1190	0.0285	0.0277	0.5903	0.1704	0.1194	0.1198

Table 3.5 : Probabilités des différents symboles obtenus par les deux algorithmes EZW et M-EZW appliqués sur l'image **Barbara** 512*512 pour différents seuils T.

	Probabilité des symboles de l'image Barbara									
	MEZW						EZW			
Symboles / Seuil	Zt	Zi	P	N	Pt	Nt	Zt	Zi	P	N
4096	0.7305	0	0	0	0.2695	0	0.8510	0	0.1490	0
2048	0.5508	0	0	0	0.4492	0	0.8087	0	0.1913	0
1024	0.6566	0.0101	0	0	0.3258	0.0076	0.8291	0.0050	0.1621	0.0038
512	0.6774	0.0619	0.0008	0	0.2216	0.0384	0.8246	0.0336	0.1209	0.0209
256	0.6271	0.1290	0.0059	0.0021	0.1655	0.0704	0.7968	0.0703	0.0934	0.0395
128	0.5752	0.1778	0.0118	0.0120	0.1337	0.0895	0.7701	0.0962	0.0788	0.0549
64	0.5430	0.1948	0.0338	0.0220	0.1124	0.0940	0.7474	0.1077	0.0808	0.0641
32	0.5317	0.2002	0.0444	0.0365	0.0960	0.0912	0.7312	0.1149	0.0806	0.0733
16	0.5194	0.1999	0.0685	0.0625	0.0754	0.0744	0.6989	0.1252	0.0902	0.0857
8	0.5123	0.1996	0.0947	0.0890	0.0526	0.0517	0.6556	0.1410	0.1041	0.0994
4	0.5016	0.2017	0.1175	0.1121	0.0341	0.0330	0.6069	0.1591	0.1196	0.1144
2	0.4958	0.2062	0.1271	0.1220	0.0249	0.0240	0.5782	0.1725	0.1272	0.1221

Table 3.6 : Probabilités des différents symboles obtenus par les deux algorithmes EZW et M-EZW appliqués sur l'image **Goldhill** 512*512 pour différents seuils T.

	Entropie des symboles de l'image Lena											
	MEZW							EZW				
Seuil \ Symboles	Zt	Zi	P	N	Pt	Nt	Entropie Totale En bpp	Zt	Zi	P	N	Entropie totale En bpp
4096	0.6854	-	-	-	0.7010	-	**0.0010**	0.2224	-	1.6126	-	0.0018
2048	0.6216	-	-	-	0.7703	-	**0.0029**	0.2154	-	1.6410	-	0.0051
1024	0.4262	4.2917	6.6896	-	1.1251	4.8978	**0.0062**	0.1894	4.9904	1.8200	5.5966	0.0096
512	0.4283	2.4842	5.6454	-	1.5087	3.2118	**0.0138**	0.2067	3.1078	2.1165	3.8355	0.0189
256	0.4903	1.9144	4.6469	5.9968	1.8379	2.6749	**0.0311**	0.2392	2.5142	2.3792	3.2392	0.0398
128	0.5512	1.6750	4.4399	4.7187	2.0886	2.3879	**0.0727**	0.2647	2.2750	2.5976	2.8951	0.0895
64	0.6028	1.6068	4.1103	3.9940	2.1523	2.2913	**0.1732**	0.2804	2.2228	2.6363	2.7400	0.2090
32	0.6285	1.5780	3.6435	3.4597	2.2485	2.3325	**0.3977**	0.2996	2.1670	2.6161	2.6408	0.4677
16	0.6427	1.5747	3.0858	3.0062	2.4229	2.4865	**0.8638**	0.3308	2.0955	2.5280	2.5406	0.9797
8	0.6436	1.5712	2.6904	2.6584	2.7289	2.7553	**1.8065**	0.3763	1.9856	2.4307	2.4269	1.9663
4	0.6589	1.5831	2.3268	2.3164	3.1953	3.2171	**3.8193**	0.4537	1.8632	2.2565	2.2555	3.9909
2	0.6956	1.5944	2.0966	2.1033	3.6157	3.6348	-	0.5333	1.7872	2.0914	2.1003	-

Table 3.7 : Entropie des différents symboles obtenus par les deux algorithmes EZW et M-EZW appliqués sur l'image **Lena** 512*512 pour différents seuils T.

	Entropie des symboles de l'image Barbara											
	MEZW							EZW				
Seuil \ Symboles	Zt	Zi	P	N	Pt	Nt	Entropie Totale En bpp	Zt	Zi	P	N	Entropie totale En bpp
4096	0.3919	-	-	-	1.1263	-	**0.0009**	0.1796	-	1.8057	-	0.0012
2048	0.6107	-	-	-	0.7830	-	**0.0028**	0.2141	-	1.6464	-	0.0045
1024	0.4261	4.2084	6.6933	-	1.1288	4.9016	**0.0062**	0.1896	4.9059	1.8225	5.5990	0.0091
512	0.3936	2.7298	6.4793	6.4793	1.4753	3.5615	**0.0126**	0.1954	3.3360	2.0748	4.1151	0.0170
256	0.4774	2.0131	4.7827	5.5247	1.7918	2.7033	**0.0279**	0.2320	2.6192	2.3489	3.2516	0.0354
128	0.5592	1.6828	3.9333	4.1527	2.0899	2.4828	**0.0739**	0.2733	2.2657	2.5258	2.8932	0.0889
64	0.6766	1.5598	2.6515	2.6906	2.5662	2.7140	**0.2729**	0.3779	2.0060	2.3610	2.4553	0.2998
32	0.7058	1.5694	2.4389	2.4473	2.7360	2.8216	**0.7562**	0.4141	1.9703	2.2842	2.3248	0.8034
16	0.7079	1.5707	2.3422	2.3592	2.8827	2.9405	**1.6787**	0.4370	1.9305	2.2431	2.2749	1.7529
8	0.7038	1.5703	2.2907	2.2821	3.0357	3.0791	**3.2310**	0.4581	1.8886	2.2205	2.2284	3.3330
4	0.6905	1.5696	2.2291	2.2239	3.2742	3.3004	-	0.4848	1.8304	2.1887	2.1916	-
2	0.6961	1.5673	2.1393	2.1288	3.5596	3.5866	-	0.5271	1.7695	2.1251	2.1217	-

Table 3.8 : Entropie des différents symboles obtenus par les deux algorithmes EZW et M-EZW appliqués sur l'image **Barbara** 512*512 pour différents seuils T.

	Entropie des symboles de l'image Goldhill											
	MEZW							EZW				
Seuil \ Symboles	Zt	Zi	P	N	Pt	Nt	Entropie Totale En bpp	Zt	Zi	P	N	Entropie totale En bpp
4096	0.3141	Inf	Inf	Inf	1.3111	Inf	**0.0008**	0.1614	Inf	1.9036	Inf	0.0011
2048	0.5964	Inf	Inf	Inf	0.8002	Inf	**0.0028**	0.2124	Inf	1.6537	Inf	0.0043
1024	0.4207	4.5951	Inf	Inf	1.1216	4.8828	**0.0059**	0.1874	5.2933	1.8198	5.5810	0.0087
512	0.3895	2.7828	7.1523	Inf	1.5068	3.2604	**0.0122**	0.1928	3.3923	2.1128	3.8699	0.0165
256	0.4667	2.0481	5.1261	6.1557	1.7990	2.6532	**0.0263**	0.2272	2.6551	2.3708	3.2305	0.0336
128	0.5530	1.7273	4.4393	4.4245	2.0120	2.4135	**0.0643**	0.2612	2.3413	2.5414	2.9019	0.0790
64	0.6107	1.6356	3.3869	3.8180	2.1858	2.3642	**0.1787**	0.2911	2.2286	2.5158	2.7472	0.2106
32	0.6317	1.6084	3.1144	3.3105	2.3429	2.3949	**0.4723**	0.3131	2.1636	2.5181	2.6135	0.5385
16	0.6551	1.6101	2.6803	2.7729	2.5855	2.5989	**1.1981**	0.3583	2.0777	2.4062	2.4565	1.3108
8	0.6688	1.6114	2.3568	2.4188	2.9445	2.9631	**2.7831**	0.4222	1.9591	2.2627	2.3090	2.9299
4	0.6900	1.6011	2.1412	2.1888	3.3781	3.4099	**5.7066**	0.4994	1.8385	2.1237	2.1677	5.8489
2	0.7016	1.5791	2.0627	2.1041	3.6917	3.7279	-	0.5478	1.7576	2.0621	2.1027	-

Table 3.9 : Entropie des différents symboles obtenus par les deux algorithmes EZW et M-EZW appliqués sur l'image **Goldhill** 512*512 pour différents seuils T.

Discussion 1 : Nombre de symboles

Les résultats obtenus montrent que l'utilisation de six symboles au lieu de quatre utilisés dans l'algorithme originale EZW, permet de diminuer le nombre total de symboles (cf. tables 3.1 - 3.3). Cela est évident parce que un symbole Pt ou Nt dans notre approche M-EZW représente quatre ou cinq symboles dans le cas de l'EZW (cf. chapitre 2, section 4.1). Cependant, la taille en bits binaires des symboles des deux algorithmes n'est pas le même. Les symboles dans le cas de l'algorithme EZW sont représentés en binaire sur 2 bits chacun (quatre symboles donc 2 bits), tandis que les symboles dans le cas de l'algorithme M-EZW sont représentés sur 3 bits chacun (6 symboles donc 3 bits). A cet effet, le nombre global des bits attribué aux symboles de chaque algorithme est égal au nombre de symboles multiplié par le nombre de bits binaire attribué à chaque symbole. On remarque que le nombre total de bits (dans le cas du codage binaire) est inférieur dans la plupart des cas pour notre approche M-EZW (cf. tables 3.1 - 3.3).

Discussion 2 : Probabilité et entropie

Pour les probabilités des symboles, on remarque dans le cas de l'EZW que le symbole 'Zt' est dominant (il représente une fréquence d'apparition très élevée par rapport aux autres symboles). La table des probabilités est redistribuée dans le cas du M-EZW, où la domination du symbole 'Zt' est moins grande par rapport à celle de l'algorithme EZW (cf. tables 3.4 – 3.6). Cette redistribution de la table de probabilités peut rendre le codage entropique encore plus efficace [32].

A cet effet, il est important de constater que, pour toutes les images de test utilisées et pour tous les seuils considérés, la quantité totale d'information (entropie en bit par symbole) des symboles est toujours inférieure dans le cas de l'algorithme M-EZW que celle pour l'EZW (tables 3.7 – 3.9).

5.2 Procédure de codage des symboles de signifiance

Dans l'algorithme EZW de Shapiro, la carte de signifiance C est composée de quatre symboles {P, N, Z et T}. Chacun de ces symboles est codé en binaire sur 2 bits. Ces symboles subissent un codage entropique avant d'être transmis. Cependant, on peut réduire quelques fois le taux de compression grâce au regroupement binaire de plusieurs symboles [30]. Notre méthode consiste donc à regrouper plusieurs symboles adjacents pour avoir un mot (code) binaire plus long avant d'effectuer l'opération de codage (cf. figure 3.4).

La figure 3.4 représente un exemple de regroupement des éléments de la liste C sur 9 bits (3 symboles juxtaposés) et des éléments de la liste S sur 8 bits.

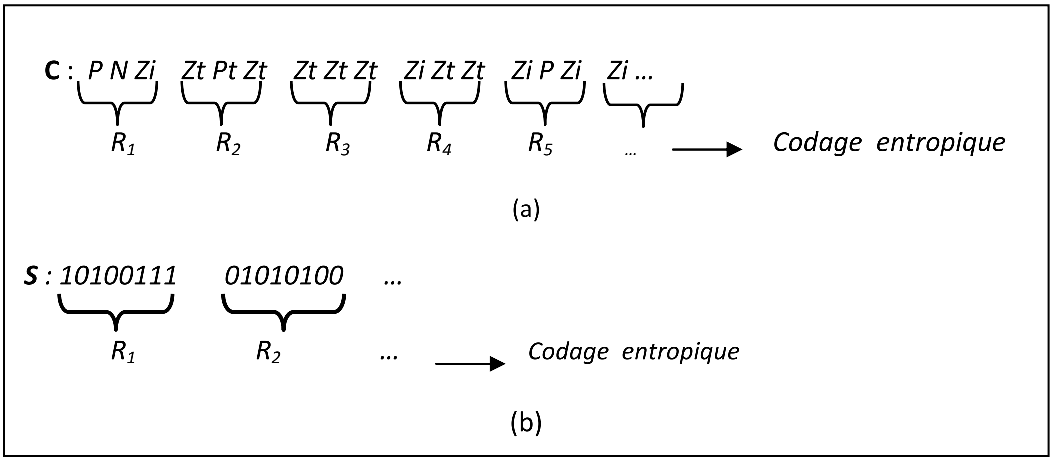


Figure 3.4 : Exemple de regroupement binaire des éléments de la liste C en 9 bits (a), et des bits de la liste S en 8 bits (b).

Nous avons étudié l'influence de ce regroupement sur le nombre bits par pixels obtenu pour la liste C et la liste S après un codage arithmétique utilisant les mêmes images de test de la section 5.1. Les tables 3.10-3.15 représentent les résultats du codage des deux listes C et S par un codage arithmétique avec un regroupement de N symboles pour la liste C et N bits pour la liste S et cela pour différents seuils T.

	Image Lena **Résultats du codage en bpp pour des regroupements de la liste C en N symboles**			
Seuil	N=1	N=2	N=3	N=4
4096	0.0033	0.0018	**0.0015**	0.0018
2048	0.0066	0.0036	**0.0030**	0.0036
1024	0.0105	0.0059	**0.0050**	0.0057
512	0.0185	0.0105	**0.0092**	0.0105
256	0.0365	0.0208	**0.0186**	0.0218
128	0.0806	0.0447	**0.0401**	0.0459
64	0.1829	0.0980	**0.0837**	0.0949
32	0.3903	0.2040	**0.1656**	0.1849
16	0.7817	0.4020	**0.3102**	0.3344
8	1.5657	0.7966	**0.5875**	0.5901
4	3.3538	1.6936	1.2010	**1.1181**

Table 3.10 : Codage arithmétique de la liste **C** de l'image **Lena** pour différents regroupements de N symboles

	Image Barbara Résultats du codage en bpp pour des regroupements de la liste C en N symboles			
Seuil	N=1	N=2	N=3	N=4
4096	0.0033	0.0018	**0.0015**	0.0017
2048	0.0067	0.0036	**0.0030**	0.0034
1024	0.0106	0.0060	**0.0052**	0.0056
512	0.0171	0.0098	**0.0086**	0.0093
256	0.0327	0.0188	**0.0173**	0.0201
128	0.0866	0.0478	**0.0427**	0.0503
64	0.3242	0.1688	**0.1340**	0.1463
32	0.7713	0.3947	**0.2978**	0.3076
16	1.4755	0.7492	0.5486	**0.5437**
8	2.5044	1.2662	0.9077	**0.8674**
4	4.1037	2.0686	1.4582	**1.3442**

Table 3.11 : Codage arithmétique de la liste **C** de l'image **Barbara** pour différents regroupements de N symboles

	Image Goldhill Résultats du codage en bpp pour des regroupements de la liste C en N symboles			
Seuil	N=1	N=2	N=3	N=4
4096	0.0033	0.0018	**0.0014**	0.0015
2048	0.0065	0.0035	**0.0029**	0.0029
1024	0.0102	0.0057	0.0049	**0.0048**
512	0.0165	0.0094	0.0086	**0.0084**
256	0.0305	0.0175	**0.0164**	0.0173
128	0.0728	0.0405	**0.0374**	0.0423
64	0.2012	0.1069	**0.0904**	0.1030
32	0.4980	0.2575	**0.2033**	0.2195
16	1.1883	0.6051	**0.4495**	0.4534
8	2.5858	1.3065	0.9329	**0.8807**
4	4.8264	2.4296	1.6972	**1.5211**

Table 3.12 : Codage arithmétique de la liste C de l'image Goldhill pour différents regroupements de N symboles

	Image Lena								
	Résultats du codage en bpp pour des regroupements de la liste S sur N bits								
Seuil	N=1	N=2	N=3	N=4	N=5	N=6	N=7	N=8	N=9
4096	0.0017	0.0009	0.0007	0.0006	0.0005	0.0006	0.0006	**0.0004**	0.0006
2048	0.0047	0.0025	0.0020	0.0018	0.0019	0.0020	0.0020	**0.0016**	0.0019
1024	0.0082	0.0044	0.0033	0.0031	0.0034	0.0035	0.0036	**0.0032**	0.0034
512	0.0130	0.0068	0.0054	0.0058	0.0054	0.0057	0.0058	**0.0053**	0.0056
256	0.0216	0.0113	0.0082	**0.0075**	0.0081	0.0089	0.0095	0.0093	0.0094
128	0.0405	0.0208	0.0148	0.0128	0.0131	**0.0144**	0.0159	0.0168	0.0166
64	0.0862	0.0438	0.0304	0.0250	0.0237	**0.0249**	0.0284	0.0316	0.0341
32	0.1874	0.0946	0.0644	0.0510	0.0455	**0.0450**	0.0497	0.0564	0.0648
16	0.3956	0.1988	0.1342	0.1039	0.0888	0.**0830**	0.0862	0.0960	0.1146
8	0.8126	0.4075	0.2736	0.2090	0.1740	0.1562	**0.1536**	0.1637	0.1938
4	1.7413	0.8721	0.5837	0.4421	0.3617	0.3151	**0.2946**	0.2964	0.3321

Table 3.13 : Codage arithmétique de la liste **S** de l'image **Lena** pour différents regroupements de N bits

	Image Barbara								
	Résultats du codage en bpp pour des regroupements de la liste S sur N bits								
Seuil	N=1	N=2	N=3	N=4	N=5	N=6	N=7	N=8	N=9
4096	0.0011	0.0006	0.0005	0.0004	0.0004	0.0003	0.0003	**0.0003**	0.0003
2048	0.0041	0.0022	0.0017	0.0016	0.0017	0.0019	0.0016	**0.0016**	0.0017
1024	0.0076	0.0040	0.0033	0.0034	0.0035	0.0033	0.0033	**0.0032**	0.0031
512	0.0119	0.0063	0.0047	**0.0045**	0.0048	0.0054	0.0053	0.0052	0.0051
256	0.0198	0.0103	0.0076	**0.0070**	0.0074	0.0085	0.0084	0.0086	0.0084
128	0.0406	0.0208	0.0148	0.0128	**0.0129**	0.0146	0.0153	0.0166	0.0175
64	0.1313	0.0663	0.0454	0.0363	0.0327	0.0334	0.0353	0.0407	0.0458
32	0.3605	0.1810	0.1222	0.0944	0.0803	**0.0752**	0.0754	0.0848	0.1002
16	0.8017	0.4017	0.2697	0.2056	0.1703	0.1524	**0.1462**	0.1555	0.1821
8	1.5454	0.7737	0.5181	0.3923	0.3209	0.2802	**0.2602**	0.2641	0.2961
4	2.7364	1.3694	0.9155	0.6912	0.5611	0.4826	0.4387	**0.4295**	0.4615

Table 3.14 : Codage arithmétique de la liste **S** de l'image **Barbara** pour différents regroupements de N bits.

	Image Goldhill								
	Résultats du codage en bpp pour des regroupements de la liste S sur N bits								
Seuil	N=1	N=2	N=3	N=4	N=5	N=6	N=7	N=8	N=9
4096	0.0009	0.0005	0.0004	0.0003	0.0003	0.0003	0.0003	**0.0003**	0.0003
2048	0.0038	0.0020	0.0016	0.0015	0.0016	0.0015	0.0014	**0.0014**	0.0015
1024	0.0072	0.0038	0.0029	**0.0028**	0.0030	0.0030	0.0031	0.0030	0.0030
512	0.0115	0.0060	0.0046	**0.0043**	0.0046	0.0051	0.0051	0.0050	0.0048
256	0.0189	0.0098	0.0073	**0.0066**	0.0073	0.0079	0.0082	0.0081	0.0081
128	0.0367	0.0188	0.0135	0.0122	**0.0117**	0.0135	0.0147	0.0155	0.0151
64	0.0892	0.0452	0.0313	0.0256	**0.0242**	0.0254	0.0282	0.0312	0.0341
32	0.2223	0.1119	0.0761	0.0597	0.0525	**0.0510**	0.0540	0.0612	0.0707
16	0.5553	0.2786	0.1876	0.1439	0.1209	0.1101	**0.1094**	0.1184	0.1376
8	1.2993	0.6509	0.4361	0.3309	0.2717	0.2380	0.2280	**0.2235**	0.2535
4	2.7305	1.3667	0.9136	0.6896	0.5599	0.4806	0.4362	**0.4235**	0.4457

Table 3.15 : Codage arithmétique de la liste **S** de l'image **Goldhill** pour différents regroupements de N bits

Discussion : Regroupement des éléments des listes C et S

Selon les résultats obtenus (cf. tables 3.10 - 3.12), on peut dire qu'un regroupement de N symboles de la liste C (N>1), donne toujours des résultats de taux de compression meilleurs par rapport au codage des symboles sans regroupement (N=1).

Nous constatons qu'un regroupement de trois symboles de la liste C donne les meilleurs résultats pour la plupart des cas et pour toutes les images de test utilisées (cf. Tables 3.10–3.12).

Pour la liste S, les résultats du regroupement de 5, 6, 7 et 8 bits donnent des résultats comparables (proches) pour la plupart des itérations (cf. Tables 3.13 - 3.15). Cependant, le regroupement sur 8 bits donne généralement les meilleurs résultats pour les itérations supérieures.

Pour notre application, nous avons utilisé un regroupement binaire sur 9 bits (trois symboles juxtaposés) des éléments de la liste C (cf. figure 3.4.a) et un regroupement sur 8 bits des éléments de la liste S (cf. figure 3.4.b). Les éléments obtenus subissent par la suite un codage arithmétique adaptatif.

6 Comparaison avec les algorithmes actuels

Nous avons comparé les performances, rapport de dégradation PSNR (en dB) et taux de compression TC (en bpp), de notre algorithme M-EZW par rapport à l'algorithme EZW de Shapiro [17], l'algorithme SPIHT [27] ainsi que le standard JPEG 2000 [16] (cf. Table 3.16 et figures 3.5 – 3.16).

Nous avons testé notre algorithme sur différentes images testes (cf. section 3), selon une décomposition en ondelettes à cinq niveaux avec des filtres biorthogonaux 9/7 (cf. section 4.2).

Discussion : Taux de compression/Rapport de dégradation

Les résultats de PSNR obtenus par le M-EZW sont meilleurs par rapport à ceux obtenus par l'algorithme EZW de Shapiro pour toutes les images testes utilisés (cf. Table 3.16 et figures 3.5 – 3.16).

Pour la comparaison avec l'algorithme SPIHT et la norme JPEG 2000, on distingue deux cas :

- Cas des images de taille 512*512 :

On remarque que le graphe du PSNR en fonction de RC peut être divisé en deux zones :

- Zone des faibles débits binaires (RC < 0.5 bpp) : Dans ce cas, il ressort que notre approche donne des résultats inférieurs par rapport aux deux algorithmes SPIHT et SPECK.
- Zone de moyennes et hauts débits binaires (RC > 0.5 bpp) : Dans ce cas, il ressort pour la plupart des images utilisées que notre approche donne des résultats proches (et même supérieurs dans certains cas) par rapport aux deux algorithmes SPIHT et JPEG 2000.

Ces résultats sont dûs à la table de probabilité des symboles qui doivent être transmis avec les données dans l'étape du codage arithmétique pour permettre leur décodage. La taille de cette table est importante par rapport aux bits codés dans le cas des faibles débits, ce qui paralyse le taux de compression. Dans le cas de moyens et hauts débits, la taille de cette table est négligeable par rapport aux bits codés et influe peu sur le taux de compression.

- Cas des images de taille 256*256 :

Les résultats de l’algorithme M-EZW sont meilleurs par rapport à ceux de l’algorithme MEZW pour toutes les images (cf. figure 3.9 – 3.16). Cependant, ces résultats sont faibles par rapport à ceux obtenus par le SPIHT et JPEG 2000 pour tous les taux de compression.

Dans ce cas on peut conclure que le codage de l’arbre de zéro à l’aide de symboles pour les deux algorithmes n’est efficace que pour les images de petites et moyennes tailles.

Les comparaisons des différentes méthodes de compression (EZW, M-EZW, SPIHT et JPEG 2000) appliquées sur toutes les images sont illustrées en figures 3.5 – 3.16.

Les résultats des images Lena, Goldhill et Barbara, reconstruites par les algorithmes M-EZW, SPIHT et JPEG 2000 pour des taux de compression de 0.25, 0.5 et 1 bpp sont présentés en figures 3.17 – 3.19.

7 Conclusion

Dans ce travail, nous avons développé un algorithme de compression d’images le M-EZW, basé sur le même principe que l’algorithme EZW de Shapiro. Cet algorithme nous a permis d’améliorer les performances de l’EZW en raison de l’utilisation de six symboles au lieu de quatre symboles de signifiance pour minimiser le nombre de symboles redondants et optimiser l’entropie des symboles, d’une part, ainsi que le regroupement binaire de l’information avant le codage arithmétique pour une meilleure optimisation de ce dernier, d’autre part. De plus, notre approche est réalisée avec un temps de calcul comparable (cf. chapitre 2) à celui de l'EZW original.

Enfin, notre algorithme a donné des résultats comparables aux résultats obtenus par les algorithmes SPIHT et la norme JPEG 2000 surtout dans le cas des moyens et haut débits, ce qui est très intéressant dans le domaine du codage hiérarchique.

Images	Taux de compression	PSNR (dB) pour chaque algorithme			
		EZW	M-EZW	SPIHT	JPEG2000
Lena (512x512)	**0.25**	30.5	32.0	**33.2**	32.9
	0.5	35.2	36.8	**37.3**	37.2
	1	38.6	**40.6**	40.5	40.4
Barbara (512x512)	**0.25**	24.7	26.7	27.2	**27.7**
	0.5	29.2	32.0	32.1	**32.9**
	1	35.0	37.8	37.5	**38.1**
Goldhill (512x512)	**0.25**	28.2	28.9	**29.9**	29.8
	0.5	31.6	33.0	33.1	**33.2**
	1	34.8	**36.8**	36.6	36.5
Peppers (512x512)	**0.25**	30.3	31.4	**32.7**	32.5
	0.5	35.0	35.6	**35.9**	35.8
	1	37.3	**38.7**	38.6	38.3
Fruits (256x256)	**0.25**	26.8	26.9	**29.3**	28.8
	0.5	31.9	32.0	**34.2**	33.6
	1	36.3	37.2	**39.2**	38.5
Airplane (256x256)	**0.25**	25.4	25.5	**27.0**	26.5
	0.5	29.6	30.4	**31.8**	31.6
	1	34.4	35.9	**37.2**	37.0
Einstein (256x256)	**0.25**	27.8	28.1	29.7	**30.0**
	0.5	32.0	32.5	33.5	**33.6**
	1	35.4	36.2	**36.9**	**36.9**
Cameraman (256x256)	**0.25**	26.9	25.0	**27.0**	26.6
	0.5	28.8	29.8	**31.5**	31.1
	1	34.1	35.3	**36.8**	36.5
Mri (256x256)	**0.25**	27.4	27.4	**29.7**	29.5
	0.5	32.7	33.0	**35.2**	**35.2**
	1	41.6	39.0	40.6	**40.7**
Abdo (256x256)	**0.25**	22.7	22.7	**24.1**	**24.1**
	0.5	27.7	27.8	29.7	**29.8**
	1	33.2	33.9	35.7	**36.0**
Angio (256x256)	**0.25**	31.1	32.0	**33.4**	33.1
	0.5	35.6	35.9	**36.9**	36.7
	1	38.7	39.8	**40.3**	40.1
Chest (256x256)	**0.25**	37.4	38.5	**41.8**	41.4
	0.5	42.6	43.2	**45.2**	44.8
	1	45.9	47.0	**48.6**	47.8

Table 3.16 : Résultats des différents algorithmes appliqués sur les douze images de test.

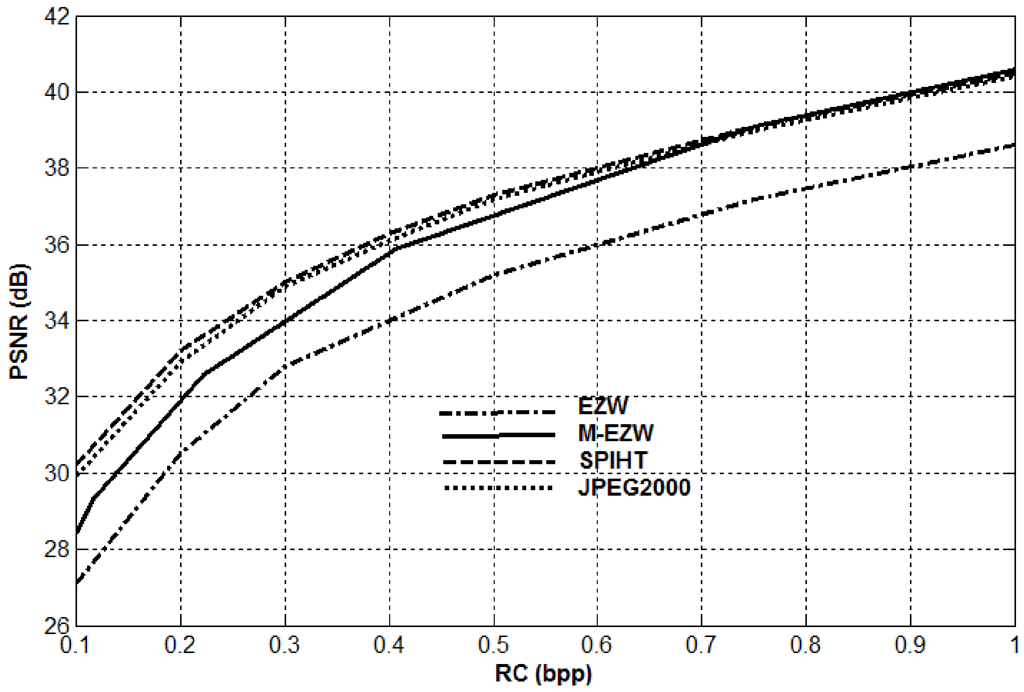


Figure 3.5. Comparaison des différents algorithmes de compression appliqués sur l'image **Lena.**

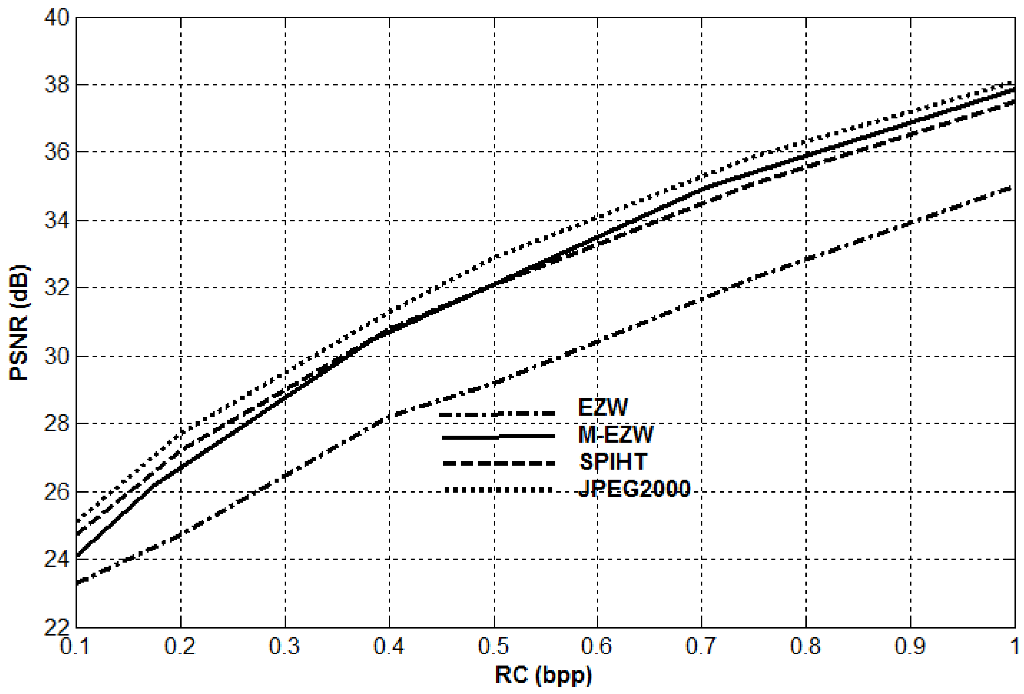


Figure 3.6. Comparaison des différents algorithmes de compression appliqués sur l'image **Barbara**

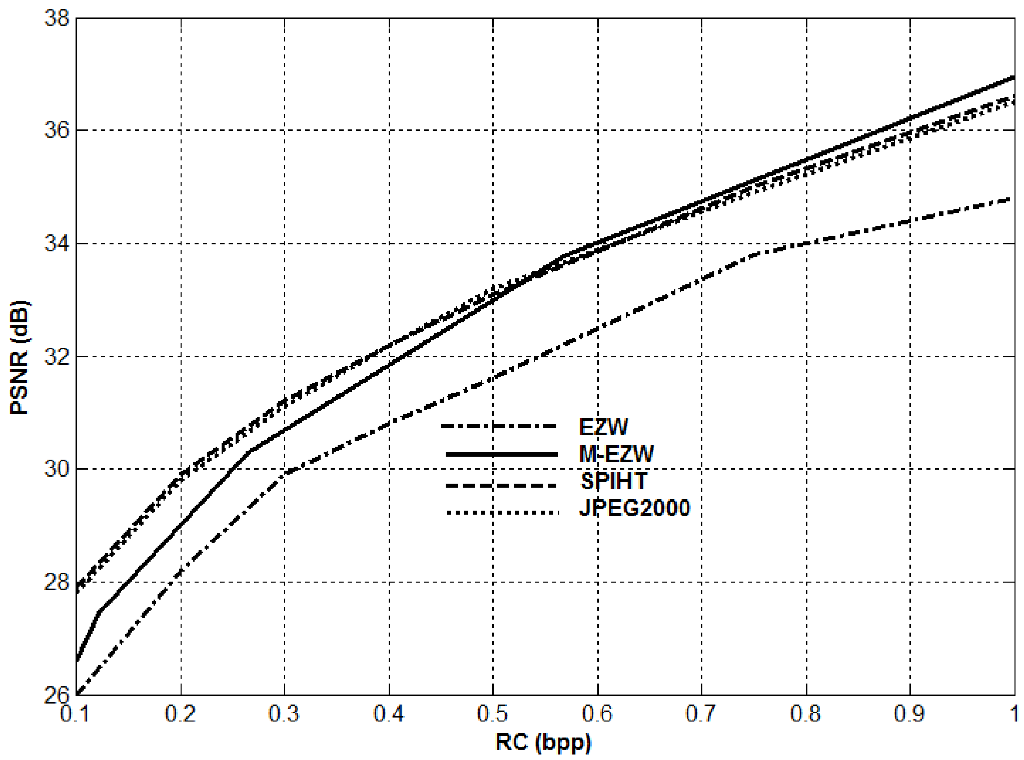


Figure 3.7. Comparaison des différents algorithmes de compression appliqués sur l'image **Goldhill**

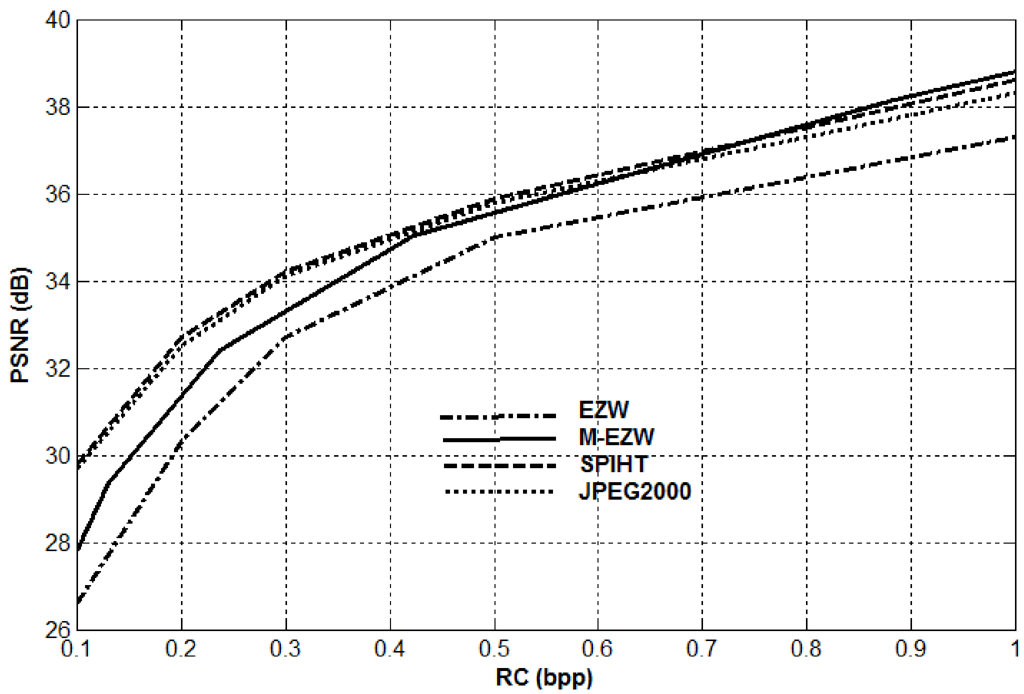


Figure 3.8. Comparaison des différents algorithmes de compression appliqués sur l'image **Peppers**

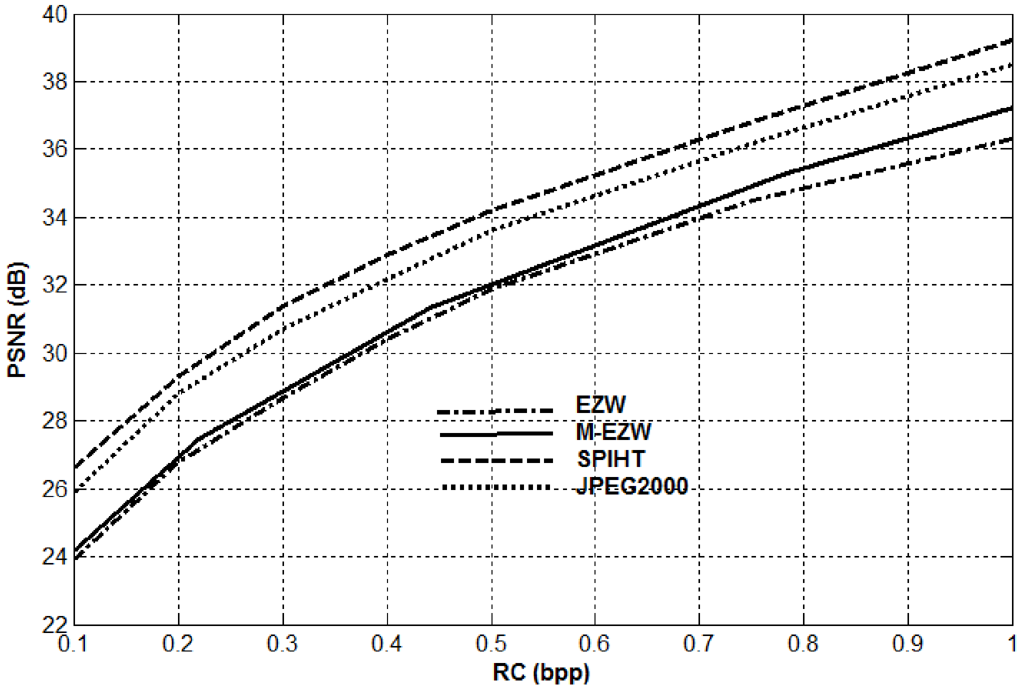


Figure 3.9. Comparaison des différents algorithmes de compression appliqués sur l'image **Fruits**

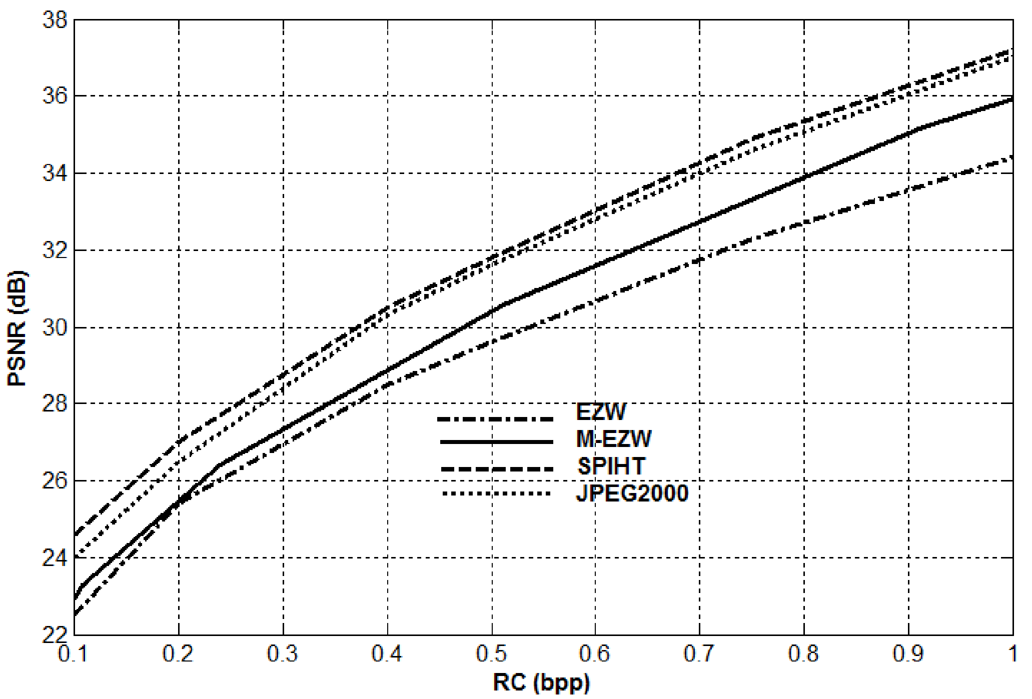


Figure 3.10. Comparaison des différents algorithmes de compression appliqués sur l'image **Airplane**

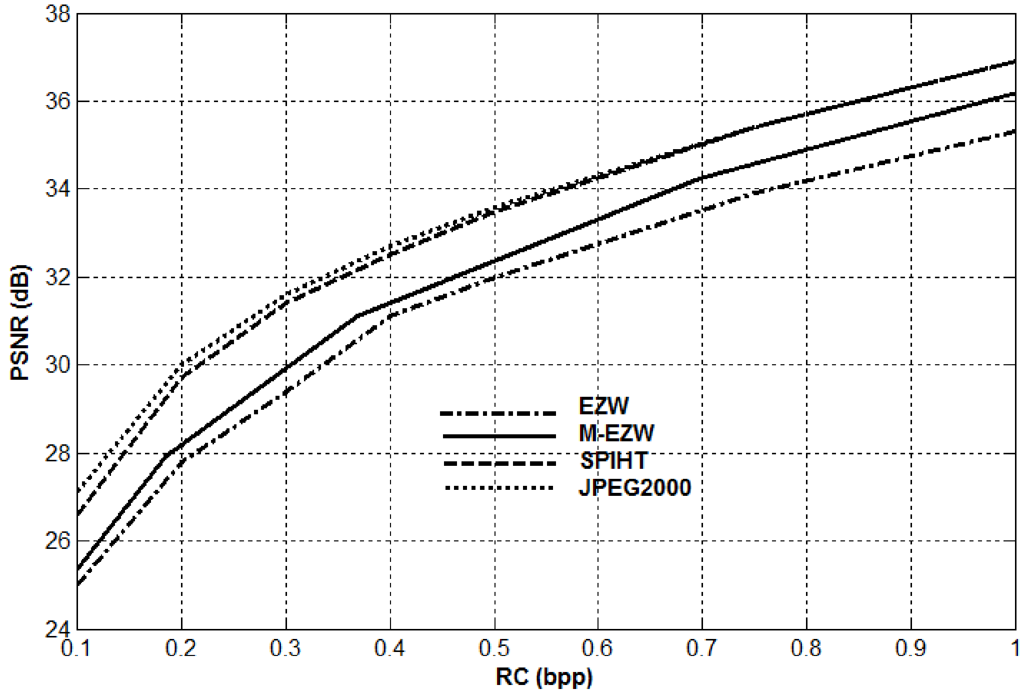


Figure 3.11. Comparaison des différents algorithmes de compression appliqués sur l'image **Einstein**

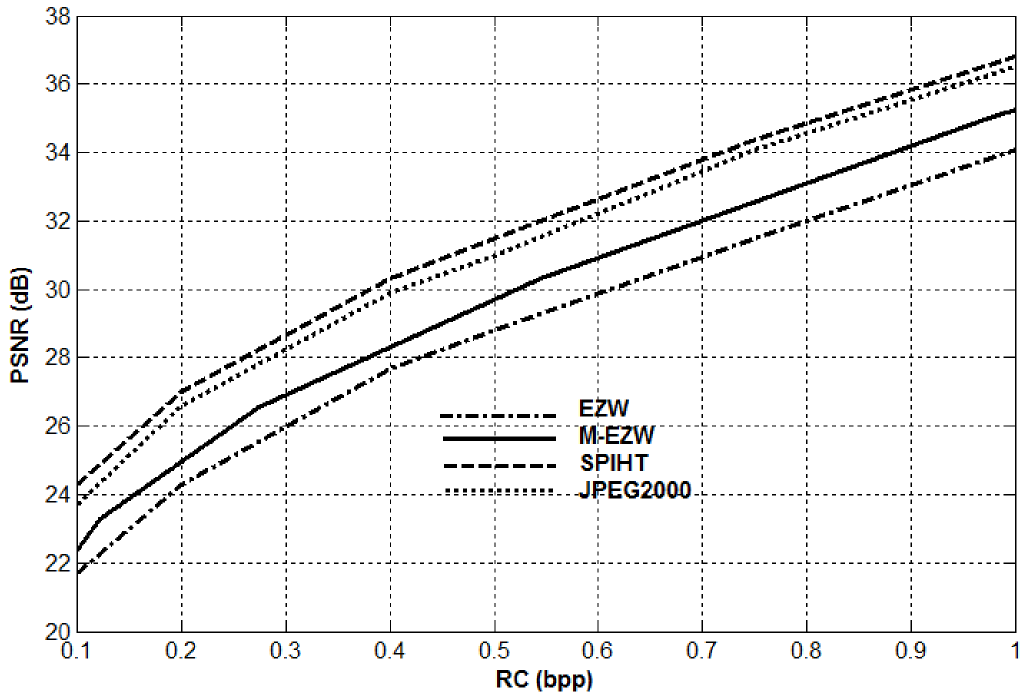


Figure 3.12. Comparaison des différents algorithmes de compression appliqués sur l'image **Cameraman**

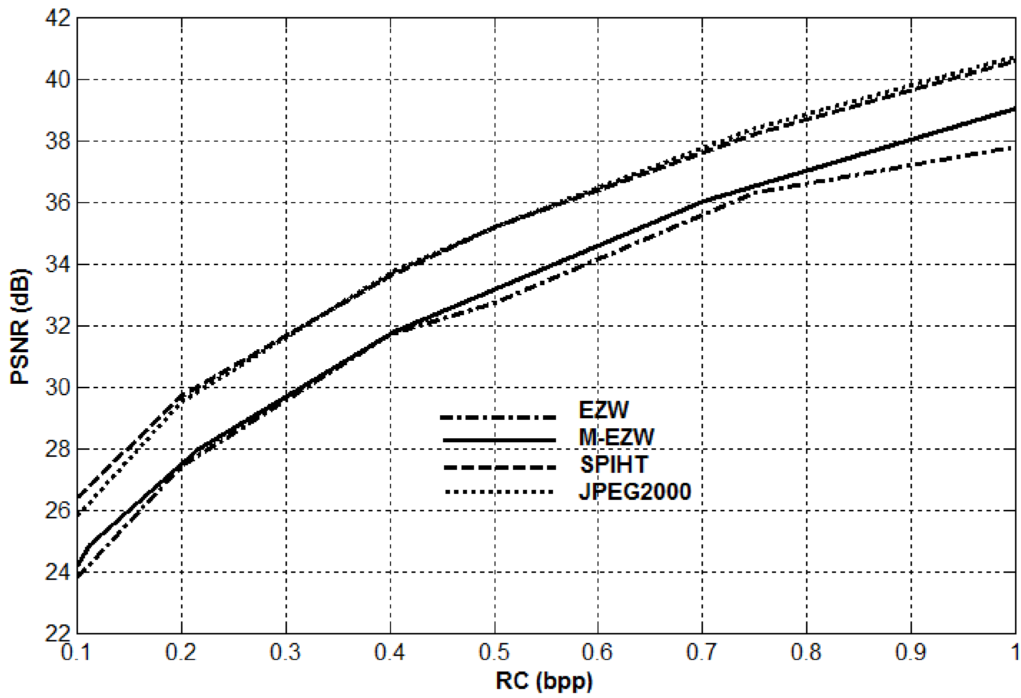


Figure 3.13. Comparaison des différents algorithmes de compression appliqués sur l'image Mri

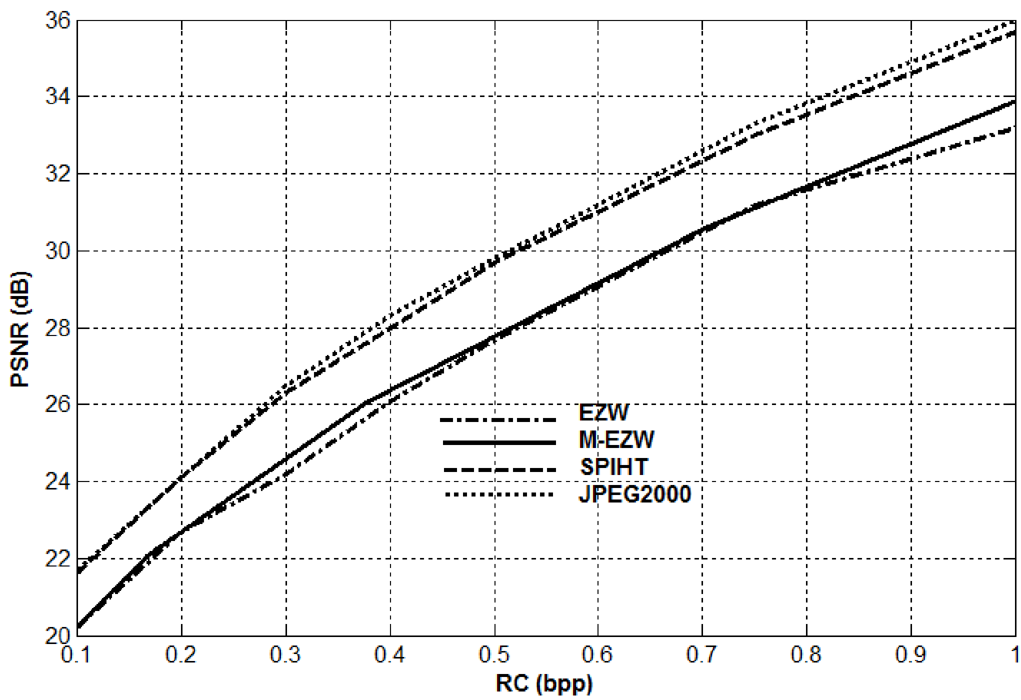


Figure 3.14. Comparaison des différents algorithmes de compression appliqués sur l'image **Abdo**

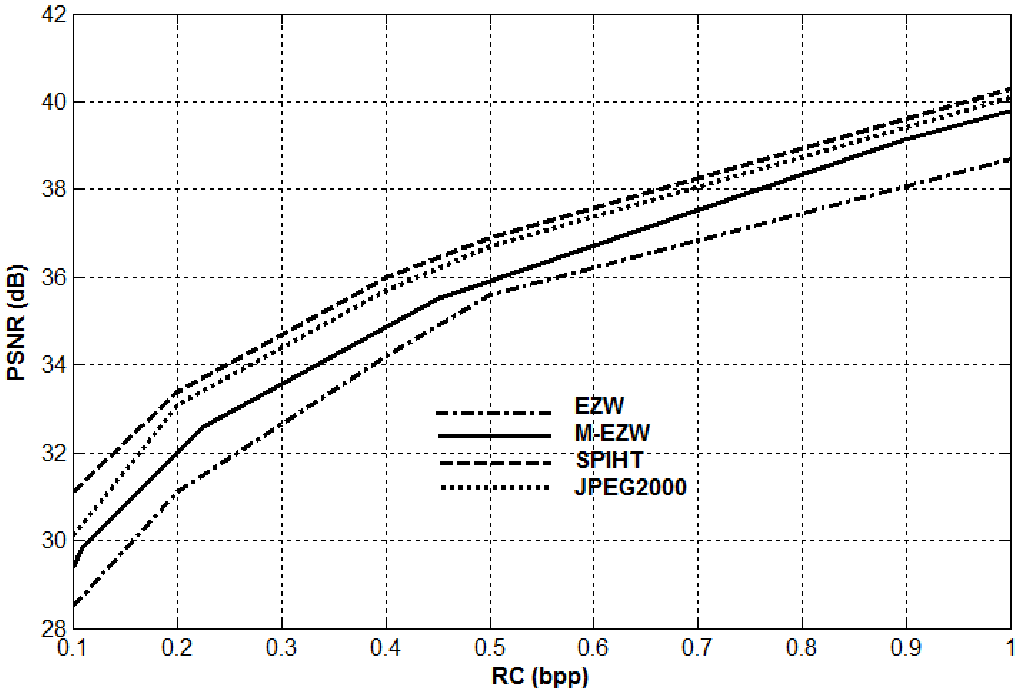


Figure 3.15. Comparaison des différents algorithmes de compression appliqués sur l'image **Angio**

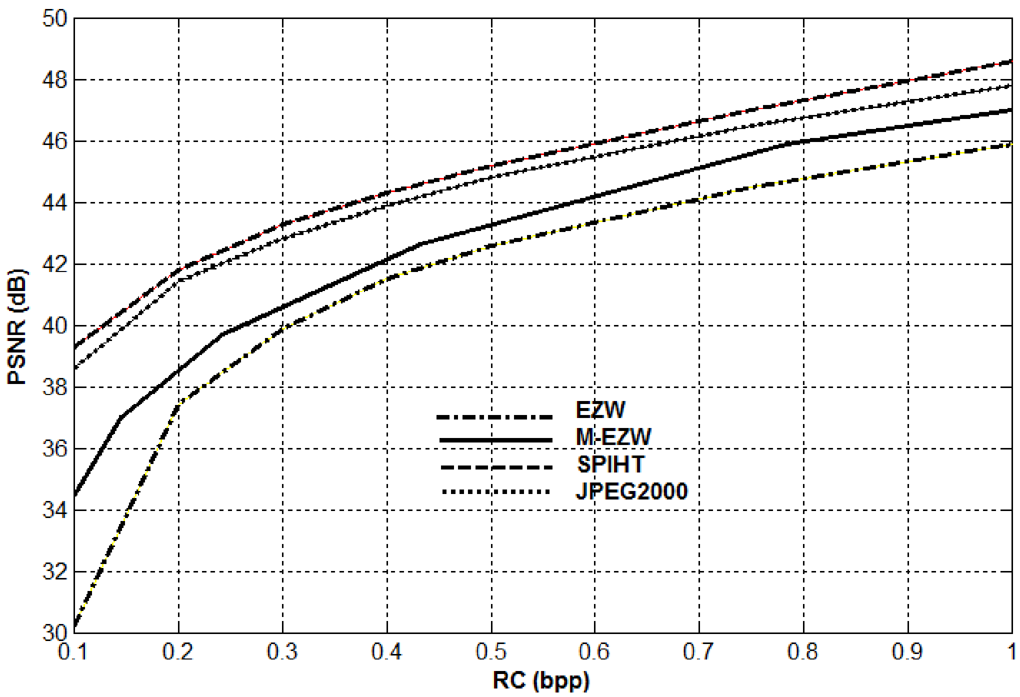


Figure 3.16. Comparaison des différents algorithmes de compression appliqués sur l'image **Chest**

a) PSNR=35.3 dB

b) PSNR=36.93 dB

c) PSNR= 37.3 dB

d) PSNR= 37.2 dB

Figure 3.17 : Résultats de l'image **Lena** reconstruite par les algorithmes : a) EZW, b) M-EZW, c) SPIHT et d) JPEG 2000 pour RC= 0.50 bpp

a) PSNR=29.6 dB

b) PSNR=31.41 dB

c) PSNR=32.1 dB

d) PSNR=32.9 dB

Figure 3.18 : Résultats de l'image **Barbara** reconstruite par a) EZW, b) M-EZW, c) SPIHT et d) JPEG 2000 pour RC= 0. 50 bpp

a) PSNR=31.7 dB

b) PSNR=32.92 dB

c) PSNR=33.1 dB

d) PSNR=33.2 dB

Figure 3.19 : Résultats de l'image **Goldhill** reconstruite par a) EZW, b) M-EZW, c) SPIHT et d) JPEG 2000 pour RC= 0.50 bpp

Conclusion générale

La compression à bas débit repose sur la construction d'opérateurs d'approximation non-linéaires qui permettent d'approcher efficacement le signal à l'aide d'un faible nombre de coefficients. Dans ce cadre, les ondelettes sont connues pour représenter de manière optimale les signaux possédant un nombre fini de discontinuités.

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

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

- D'une part, elle emploie six symboles au lieu de quatre, employés dans l'algorithme EZW 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 EZW. A cet effet, nous avons obtenu une quantité d'information inférieure par rapport à l'algorithme EZW.
- Et d'autre part, 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 l'EZW (codage de chaque symbole à part).

En outre, la robustesse du point de vue rapport taux de compression/qualité d'image de l'algorithme M-EZW est meilleure dans la plupart des cas par rapport à l'algorithme EZW original et comparable avec celle des algorithmes SPIHT, SPECK et JPEG 2000.

En perspectives, nous proposons d'une part l'application du principe du codage M-EZW dans le cas des codeurs SPIHT et SPECK. 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 méthode M-EZW avec un algorithme de compression utilisant les ondelettes géométriques. 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 approche pourrait réduire en plus le nombre de coefficients à coder.

Annexe

Annexe A.1. Exemple d'application de l'algorithme SPIHT

Etape	Pixel ou ensemble testés	Bit de sortie	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) to LSP	LIP = {(0,1),(1,0),(1,1)} LSP = {(0,0)}
	(0, 1)	1–	(0,1) to LSP	LIP = {(1,0),(1,1)} LSP = {(0,0),(0,1)}
	(1, 0)	0	None	
	(1, 1)	0	None	
(3)	D(0, 1)	1	test offspring	LIS = {**(0,1)A**,(1,0)A,(1,1)A}
	(0, 2)	1+	(0,2) to LSP	LSP = {(0,0),(0,1),(0,2)}
	(0, 3)	0	(0,3) to LIP	LIP = {(1,0),(1,1),(0,3)}
	(1, 2)	0	(1,2) to LIP	LIP = {(1,0),(1,1),(0,3),(1,2)}
	(1, 3)	0	(1,3) to LIP	LIP = {(1,0),(1,1),(0,3),(1,2),(1,3)}
(4)			Changement de type	LIS = {(1,0)A,(1,1)A,(0,1)B}
(5)	D(1, 0)	1	test offspring	LIS = {**(1,0)A**,(1,1)A,(0,1)B}
	(2, 0)	0	(2,0) to LIP	LIP = {(1,0),(1,1),(0,3),(1,2),(1,3),(2,0)}
	(2, 1)	0	(2,1) to LIP	LIP = {(1,0),(1,1),(0,3),(1,2),(1,3),(2,0),(2,1)}
	(3, 0)	0	(3,0) to LIP	LIP = {(1,0),(1,1),(0,3),(1,2),(1,3),(2,0),(2,1),(3,0)}
	(3, 1)	0	(3,1) to LIP	LIP = {(1,0),(1,1),(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}
(6)	D(1, 1)	0	None	LIS = {**(1,1)A**,(0,1)B,(1,0)B}
(7)	L(0, 1)	0	None	LIS = {(1,1)A,**(0,1)B**,(1,0)B}
(8)	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}
(9)	D(2, 0)	0	None	LIS = {(1,1)A,(0,1)B,**(2,0)A**,(2,1)A,(3,0)A,(3,1)A}
(10)	D(2, 1)	1	test offspring	LIS = {(1,1)A,(0,1)B,(2,0)A,**(2,1)A**,(3,0)A,(3,1)A}
	(4, 2)	0	(4,2) to LIP	LIP = {(1,0),(1,1),(0,3),(1,2),(1,3),(2,0),(2,1),(3,0),(3,1),(4,2)}
	(4, 3)	1+	(4,3) to LSP	LSP = {(0,0),(0,1),(0,2),(4,3)}
	(5, 2)	0	(5,2) to LIP	LIP = {(1,0),(1,1),(0,3),(1,2),(1,3),(2,0),(2,1),(3,0),(3,1),(4,2),(5,2)}
	(5, 3)	0	(5,3) to LIP	LIP = {(1,0),(1,1),(0,3),(1,2),(1,3),(2,0),(2,1),(3,0),(3,1),(4,2),(5,2),(5,3)}
(11)			enlever (2,1)	LIS = {(1,1)A,(0,1)B,(2,0)A,(3,0)A,(3,1)A}
(12)	D(3, 0)	0	None	LIS = {(1,1)A,(0,1)B,(2,0)A,**(3,0)A**,(3,1)A}
	D(3, 1)	0	None	LIS = {(1,1)A,(0,1)B,(2,0)A,(3,0)A,**(3,1)A**}
(13)				LIS = {(1,1)A,(0,1)B,(2,0)A,(3,0)A,(3,1)A} LIP = {(1,0),(1,1),(0,3),(1,2),(1,3),(2,0),(2,1),(3,0),(3,1),(4,2),(5,2),(5,3)} LSP = {(0,0),(0,1),(0,2),(4,3)}

Table A.1 : Différentes étapes de l'algorithme SPIHT appliqué sur la matrice de test de la figure 1.8 pour une itération (T_0=32),

Bibliographie

[1] J-P Guillons, “Techniques de compression des images”, Edition Hermes, 1996.

[2] P. Destuynder, “Analyse et traitement des images numériques“, Eddition Hermes, 2006.

[3] M. Barlaud , C. Labit, “Compression et codage des images et des vidéos“, Edition Hermes,2002

[4] A. Naït-Ali, C. Cavaro-Ménard, “Compression des images et des signaux médicaux“, Edition Hermes, 2007.

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

[6] G. Burel, “Introduction au traitement d’images, simulation sous matlab“, Edition Lavoisier, 2001.

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

[8] D. Lingrand, “Introduction au traitement d’images“, Edition Vuibert, Paris 2004.

[9] M. A. Losada, G. Tohumoglu, D. Fraile, A. Artès, “Multi-itération wavelet zero-tree coding for image compression”. Elsevier Science, Signal Processing, vol. 80, pp. 1281-1287, 2000.

[10] G. K. Wallace, “The JPEG still-picture compression standard,” *Commun.ACM*, vol. 34, pp. 30–44, Avr. 1991.

[11] “MPEG-2video,” ITU-T Recommendation H.262-ISO/IEC 13818-2, Jan. 1995.

[12] K.R. Rao and P. Yip, "Discrete Cosine Transform: algorithms, advantages, applications", Academic Press, 1990.

[13] Z. Xiong, K. Ramchandran, M. T. Orchard, and Ya-Qin Zhang, “A Comparative Study of DCT- and Wavelet-Based Image Coding”. IEEE Transctions on circuits and systems for video technology, vol. 9, pp. 692-695, 1999

[14] R. Ruhl, H. Hartenstein et D. Saupe ,"Adaptative partitionning for fractal image compression", IEEE international Conference on image processing , vol. 5, n°9, Santa Barbara. October 1997.

[15] J. Oliver. M.P. Malumbres, “A new fast lower-tree wavelet image encoder”. International Conference on Image Processing, vol. 3, pp. 780-783 vol. 3, 2001.

[16] D. Taubman, "High performance scalable image compression with EBCOT". IEEE

Transactions on Image Processing, 9(7),1158-1170 (2000).

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

[18] M. D. Adams, A. Antoniou, “Reversible EZW-Based Image Compression Using Best-Transform Selection and Selective Partial Embedding”. IEEE Transactions on circuits and systems-II: Analog and digital processing, Vol. 47, NO. 10, pp. 1119-1122, Oct. 2000.

[19] R. Thomas, “Algorithmes Rapides de Reconstruction en Tomographie par Compression des Calculs. Application à la Tomofluoroscopie 3D”. Thèse de Doctorat, Institut national polytechnique de GRENOBLE, Oct 2002.

[20] J. Woe, C. Olivier, C. Tellier, “Embedded zero-tree runlength wavelet image coding”, International Conference on Image Processing, vol. 13, pp. 784-787, 2001

[21] V.N.Ramaswamyt, K.R.Vamuduri and N.Ranganathan, “Context-Modeling of wavelet coefficients in EZW-Based lossless image coding”, Acoustics, Speech, and Signal Processing, 1999. ICASSP '99 Proceedings pp. 3165-3167.

[22] M. Penedo, R. Sergio. And R. Seara, "An improved EZW algorithm based on set partitioning in hierarchical trees using wavelet regularity", ICIP '04. 2004., vol. 5, pp. 3169 - 3172, 24-27 Oct. 2004.

[23] J. Zhou, P. Huang, and F-P Chiang, “Wavelet-based pavement distress image compression and noise reduction,” Proc. SPIE Vol. 5914, pp. 629-639, 2005.

[24] T. Muzaffar and T-S Choi, "Simplified EZW image coder with residual data transmission," SPIE International Symposium on Optical Science, Engineering and Instrumentation, San Diego, Juillet 2000.

[25] T. Muzaffar, T.S. Choi, "A Simple & Efficient Modified EZW Image Coder Based on Fixed Value Transmission," ISCE'99, vol.983, pp 207-210. 1999.

[26] D. Le Guen, ”Stratégies de quantification pour la compression d’images par transformées en ondelettes : un état de l’art, Projet TEMICS – IRISA, Mars 1998.

[27] 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.

[28] 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.

[29] V. Chappelie, ”Codage progressif d'images par ondelettes orientées”. Thèse de

doctorat, Université de Rennes 1. Déc 2005.

[30] K. Sayood, "Introduction to data compression". Second Edition, Morgan Kaufmann Publishers, 2000.

[31] M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies. “Image coding using wavelet transform”. IEEE Transaction, Image Processing, 1(2):205–220, Avril 1992.

[32] 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.

[33] H. Guitter, "La compression des images numériques", Hermes, 1995.

[34] 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.

[35] A. Munteanu, J. Cornelis and P. Cristea, "Wavelet-Based Lossless Compression of Coronary Angiographic Images ", IEEE Transaction on medical imaging, vol. 18, no. 3, Mars 1999.

[36] J. Liang, "The predictive embedded zerotree wavelet (PEZW) coder: Low complexity image coding with versatile functionality," Proc. IEEE Internatl. Conf. on Signal Processing (ICASSP), Phoenix, AZ, March 1999.

[37] S-T. Hsiang and J-W. Woods, “Embedded image coding using zeroblocks. of subband/wavelet coefficients and context modeling,” Proc. IEEE. ISCAS’00, vol 3, pages 662–665, 2000.

[38] M. Barlaud, C. Labit, “Compression et codage des images et des vidéos », Edt Hermes Sciences, 2002

[39] I. Avcibas; N. Memon; B. Sankur; K. Sayood; “A successively refinable lossless image-coding algorithm”, IEEE transactions on communications, vol. 53, n^{o}3, pp. 445-452, 2005.

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

[41] H. Guitter, "La compression des images numériques", Edition Hermes, 1995.

[42] 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.

[43] 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.

[44] J.D. Villasenor, B. Belzer, J. Liao, “Wavelet Filter Evaluation for Image Compression”, EEE TRANSACTIONS ON IMAGE PROCESSING, VOL. 4, NO. 8, pp 1053-1060, AUGUST 1995.

[45] M.D. Adams, and F.Kossentini, ‘‘Performance Evaluation of reversible ‘‘integer to integer’’ Wavelet Transforms for Image compression’’, IEEE Trans on image Processing Vol9, pp1010-1024, June 2000.

[46] F. Sheng, A. Bilgin, P.J. Sementilli, M.W.Marcellin, ‘‘Lossy and lossless image compression using reversible integer wavelet transforms’’, Proc. In International Conference on Image Processing ICIP’98, CA: IEEE Press, vol.3, pp.876-880, Ls Alamitos, 1998.

[47] I. Daubechies, “Orthonormal basis of compactly suppord wavelets”, Pure and APPL Math, Vol 41, Novembre 1988.

[48] A.R. Calderbank, I. Daubechies, W Sweldens, and B-L. Yeo. “Lossless image compression using integer to integer wavelet transforms”. International conference on Image Processing, volume 1, p596-599, Santa Barbara, CA, USA, Oct 1997.

[49] A.R. Calderbank, I. Daubechies, W.Sweldens and B.L.Yeo, “Wavelet transforms that map integers to integers”,Applied and Computational Harmonics Analysis, Vol5, No.3, pp.332-369,1998.

[50] A. Vassiliou, V. Wickerhauser, “Comparison of Wavelet Image Coding Schemes for Seismic Data Compression”, SPIE Proceedings vol. 3169, 27 Fev 1997.

[51] J.D. Villasenor, B.Belzer and J.Liano “Wavelet Filter Evaluation for image Compresion”, International conference on Image Processing ICIP’97, Vol.1, pp.624-627,1997.

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

[53] 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.

[54] S. Hsiang and J. W. Woods, “Embedded video coding using invertible motion compensated 3-d subband/wavelet filter bank,” Signal Processing : Image Communications, vol. 16, pp. 705-724, May 2001.

[55] S. M. LoPresto, K. Ramchandran, and M. T. Orchard, “Image coding based on mixture modeling of wavelet coefficients and a fast estimation-quantization framework,”

in IEEE Data Compression Conference, 1997, pp. 221-230.

[56] Z. Xiong, K. Ramchandran, and M. T. Orchard, “Space-frequency quantization for wavelet image coding,” IEEE Transactions on Image Processing, vol. 6, pp. 677-693, 1997.

[57] M. A. Losada, G. Tohumoglu, D. Fraile, A. Artès, “Multi-itération wavelet zero-tree coding for image compression”. Elsevier Science, Signal Processing, vol. 80, pp. 1281-1287, 2000.

[58] M. Penedo, R. Sergio. And R. Seara, "An improved EZW algorithm based on set partitioning in hierarchical trees using wavelet regularity", ICIP '04. 2004., vol. 5, pp. 3169-3172, 24-27, Oct. 2004.

[59] J. Zhou, P. Huang, and F-P Chiang, “Wavelet-based pavement distress image compression and noise reduction,” Proc. SPIE Vol. 5914, pp. 629-639, 2005.

[60] T. Muzaffar and T-S Choi, "Simplified EZW image coder with residual data transmission," SPIE International Symposium on Optical Science, Engineering and Instrumentation, San Diego, Juillet 2000.

[61] T. Muzaffar, T.S. Choi, "A Simple & Efficient Modified EZW Image Coder Based on Fixed Value Transmission," ISCE'99, vol.983, pp 207-210. 1999.

[62] A. Munteanu, J. Cornelis and P. Cristea, "Wavelet-Based Lossless Compression of Coronary Angiographic Images ", IEEE Transaction on medical imaging, vol. 18, no. 3, Mars 1999.

[63] J. Liang, "The predictive embedded zerotree wavelet (PEZW) coder: Low complexity image coding with versatile functionality," Proc. IEEE Internatl. Conf. on Signal Processing (ICASSP), Phoenix, AZ, March 1999.

[64] M. Babel, “Représentation et compression de séquences d’images par la méthode LAR”, thèse de doctorat, INSA de Rennes, Septembre 2005.

[65] Patrice ABRY. «Ondelettes et Turbulence». DIDEROT multimédia (1997). Pages (2-26, 71-73).

[66] Martin VETTERLI, Jelena KOVOCEVIC. «Wavelets and sub-band coding ». IEEE Library (1995). Pages (77-79, 223, 98-100, 151-152, 311).

[67] Emmanuelle Bournay Bouchereau. « Analyse d'images par transformées en ondelettes, application aux images sismiques ». Thèse de doctorat, Université JOSEPH FOURIER- GRENOBLE (Mars 1997).

[68] A. COHEN. «Ondelettes et traitement numérique du signal». Edition MASSON (1992).

[69] S.Mallat, "A Wavelet tour of signal processing”, Academic Press, New York, 1998.