REPUBLIQUE **A**LGERIENNE **D**EMOCRATIQUE **E**T **P**OPULAIRE

MINISTERE DE L'**E**NSEIGNEMENT **S**UPERIEUR ET DE LA **R**ECHERCHE

SCIENTIFIQUE

UNIVERSITE **M**OULOUD **M**AMMERI DE **T**IZI-**O**UZOU

FACULTE DU **G**ENIE **E**LECTRIQUE ET D'**I**NFORMATIQUE
DEPARTEMENT D'**E**LECTRONIQUE

Mémoire de Fin d'Etudes
De MASTER ACADEMIQUE

Domaine : **Sciences et Technologies F**ilière : **Génie électrique**

S**p**écialité : **Télécommunication et réseaux**

Présenté par

HADJEM KARIMA

ET ZIAD MALHA

Thème

Etude comparative de méthodes de compression d'images basées sur les codeurs par plans de bits

Mémoire soutenu publiquement le 28/09/2016.devant les jurys composé de :

Mme Z.AMEUR	Présidente
Mme N.HEMDANI	Encadreur
Mme L.AKROUR Epse LAHDIR	Examinateur
Mr D.ALOUACHE	Examinateur

Remerciements

Nous tenons à témoigner notre profonde gratitude et nos remerciements les plus sincères à notre promotrice M^me^ HEMDANI NAIMA de nous avoir encadré, suivi et soutenu tout au long de ce travail.

Nos remerciements les plus vifs vont également aux membres du jury qui nous ferons l'honneur de juger notre travail.

Que toute personne qui, d'une manière ou d'une autre, nous a aidées et encouragées pour l'aboutissement de ce travail, trouve ici l'expression de notre sincère reconnaissance.

Résumé :

Le développement remarquable dans le domaine des technologies de l'information et la diversité des applications multimédias dans les dernières années implique le développement des techniques de compression d'image plus efficaces dont l'objectif est d'améliorer la capacité de transmission et le stockage des données.

Les travaux de recherche récents ont montré que la compression d'images basée sur les codeurs par plan de bits offre un taux de compression très intéressant avec un bonne qualité de l'image restituée.

Nous proposons dans ce travail une étude de méthodes de compression d'images basée sur la transformée en ondelettes et les codeurs par plan de bits (EZW,SPIHT ,EBCOT qui est le codeur du standard JPEG 2000) afin d'évaluer les résultats obtenus par des codeurs dans l'objectif de sélectionner le meilleur codeur qui permet d'obtenir les résultats les plus performants.

Mots-Clés : Compression d'image, ondelettes ,algorithme pyramidal de S.MALLAT, , EZW,SPIHT ,EBCOT .

LISTE D'ABREVIATIONS

DCT	Transformée en cosinus discrète.
TOC	Transformée en ondelettes continue.
TOD	Transformée en ondelettes discrète.
TODI	Transformée en ondelettes discrète inverse.
AMR	Analyse multi résolution.
JPEG	Joint Photographic Expert Group.
JPEG2000	Standard de compression d'images fixes récent, introduit par JPEG.
MSE	Erreur quadratique moyenne.
PSNR	Rapport signal sur bruit crête.
RC	Rapport de compression.
Bpp	bit par pixel.
RLC	Codage de longueur de séquence.
AC	Codeur arithmétique.
LZW	Lempel Zip Welch.
EZW	Embedded zero tree wavelets.
SPHIT	Set Partitioning In Hierarchical Trees
EBCOT	Embedded Block Coding with Optimal Truncation Points

SOMMAIRE

Chapitre II **La Théorie des ondelettes**

Chapitre III Les algorithmes

INTRODUCTION GENERALE

L'image est un support d'information très performant, et comme on dit : une image vaut plus que mille mots. Vu l'importance de l'image, et la grande quantité d'information qu'elle peut contenir, le monde s'intéresse de plus en plus à l'image et tends vers l'universalisation de son utilisation. En effet, l'image a touché plusieurs domaines de notre vie : la médecine, la météo, la télécommunication, la cartographie, la géologie, etc.

Le stockage et la transmission de ces images prend beaucoup d'espace et de temps et ce malgré l'augmentation de la capacité de stockage et ra rapidité des nouveaux supports de transmission.

Pour remédier à ces contraintes, les chercheurs ont développé au cours des dernières décennies de nombreuses méthodes de compression de données déduites de la théorie de l'information et faisant appel à de nombreux domaines des mathématiques et de l'informatique. Les méthodes de compression de données peuvent être classées suivant la nécessité de récupération parfaite ou non de l'information originale en deux grandes catégories : La compression sans pertes d'information ou réversible qui a l'avantage de préserver la qualité de l'image originale, mais avec un taux de compression relativement faible et la compression avec pertes qui regroupe des algorithmes caractérisés par leur taux de compression assez élevé tout en gardant le mieux possible une acceptable qualité de l'image originale. Elle est basée généralement sur une phase de transformation qui sert à compacter l'information utile dans un nombre minimum de coefficients non nuls. La transformation la plus courante est celle par ondelettes classiques qui est à la base des standards de compression les plus récents. La nécessité de compresser les images apparaît donc aujourd'hui incontournable. De plus, la compression présente un intérêt évident pour la transmission des images qui peut s'avérer délicate du fait des bandes passantes existantes limitées. L'idée de base de la compression des images est de réduire le nombre moyen de bits par pixel nécessaire à sa représentation. Il est possible dans une certaine limite de réduire ce nombre sans perte d'information. Au-delà, il est nécessaire d'élaborer des algorithmes de compression irréversibles (avec pertes) induisant une distorsion pas ou peu visible dans les conditions normales d'observation des images.

Dans ce mémoire, nous avons effectué une étude comparative des méthodes de compression d'images basées sur la transformée en ondelettes et les codeurs par plans de bits (EZW, SPIHT, EBCOT qui le codeur du standard JPEG2000) afin d'évaluer les résultats obtenus par chacun des algorithmes dans l'objectif de sélectionner le meilleur codeur qui permet d'obtenir les résultats les plus performants.

Ce mémoire est structuré autour de quatre chapitres :

Après une introduction générale, vient le premier chapitre qui sera réservé à des généralités et les méthodes principales de compression des images fixes.

Dans le deuxième chapitre, nous exposerons la théorie des ondelettes.

Le troisième chapitre, a été consacré à l'étude détaillée des trois algorithmes EZW, SPHT, EBCOT.

Dans le dernier chapitre, nous présenterons le processus de compression et de décompression d'images établis dans la littérature, basée sur une transformation en ondelettes et associée à l'un des trois codeurs par plans de bits (EZW, SPITH, EBCOT), ainsi que les tests et les résultats obtenus.

Notre mémoire s'achèvera par une conclusion générale et quelques perspectives.

Introduction

La représentation des images fixes fait partie des applications multimédias dans la plupart des systèmes de communication. La numérisation des images fiabilise leur transmission à travers des réseaux informatique et facilite leur manipulation. Toutefois, l'image numérique occupe une place importante souvent trop importante pour être stockée, et encore plus pour être transmise sur des réseaux bas ou moyen débit. La question qui se pose, comment peut-on minimiser le temps de transmission, et optimiser en capacité de stockage ? En réponse à cette question des méthodes de compressions d'images sont développées. Nous allons détailler certaines dans ce présent chapitre.

I.1 L'image Numérique :

La définition du terme « image » lui-même, telle qu'elle est donnée par exemple par le dictionnaire **Petit Robert**, englobe une multitude de significations distinctes. Cela va de la « reproduction exacte ou représentation analogique d'un être, d'une chose », à la « représentation mentale d'origine sensible » ou à des concepts plus physiques comme un « ensemble de points » où vont converger des rayons lumineux (cas des images optiques). Afin de rendre l'image exploitable par les équipements audiovisuels ou informatiques, on procède à sa numérisation.

Le terme d'image numérique désigne, dans son sens le plus général, toute image qui a été acquise, traitée et sauvegardée sous une forme codée représentable par des nombres (valeurs numériques).

La numérisation est le processus qui permet de passer de l'état d'image physique (image optique par exemple) qui est caractérisée par l'aspect continu du signal qu'elle représente (une infinité de valeur dans l'intensité lumineuse par exemple), à l'état d'image numérique qui est caractérisée par l'aspect discret (l'intensité lumineuse ne peut prendre que des valeurs quantifiées en un nombre fini de points distincts). C'est cette forme numérique qui permet une exploitation ultérieure par des outils logiciels sur ordinateur.

- **Images à niveaux de gris :**

Le niveau de gris est la valeur de l'intensité lumineuse en un point. La couleur du pixel peut prendre des valeurs allant du noir au blanc en passant par un nombre fini de niveaux intermédiaires. Le nombre de niveaux de gris dépend du nombre de bits utilisés pour décrire la " couleur " de chaque pixel de l'image. Plus ce nombre est important, plus les niveaux possibles sont nombreux.

- **Images en couleur :**

La représentation des couleurs s'effectue de la même manière que les images monochromes avec cependant quelques particularités. En effet, il faut tout d'abord choisir un modèle de représentation. On peut représenter les couleurs à l'aide de leurs composantes primaires. Les systèmes émettant de la lumière (écrans d'ordinateurs,...) sont basés sur le principe de la synthèse additive : les couleurs sont composées d'un mélange de rouge, vert et bleu (RVB).

- **Le format BMP (Bit-MaP)** :

C'est un format non compressé d'une image matricielle comme son nom l'indique une matrice de cases. Chaque case, plus connue sous le nom de pixel, contient une couleur codée sur un nombre donné de bits. En grossissant une image matricielle, elle perd de la qualité, et l'image se "pixélise". Ce format d'image est utilisé dans la photographie.

I.2 Caractéristiques d'une image numérique [1] [17] :

L'image numérique est un ensemble structuré d'informations caractérisé par les paramètres suivants :

I.2.1 Pixel :

Le pixel est la contraction de l'expression anglaise " picture elements". Le pixel, étant le plus petit point de l'image, c'est une entité calculable qui peut recevoir une structure et une quantification.

Dans une image couleur (R.V.B.), un pixel peut être représenté sur trois octets : un octet pour chacune des couleurs : rouge (R), vert (V) et bleu (B).

I.2.2 Dimension :

C'est la taille de l'image. Cette dernière se présente sous forme de matrice dont les éléments sont des valeurs numériques représentatives des intensités lumineuses (pixels). Le nombre de lignes de cette matrice multiplié par le nombre de colonnes nous donne le nombre total de pixels dans une image.

I.2.3 Entropie :

L'entropie est une grandeur qui caractérise la quantité d'information que contient une image. Par exemple, une image dont tous les pixels ont la même valeur contient très peu d'informations car elle est extrêmement redondante, donc son entropie est faible. En revanche une image dont tous les pixels ont une valeur aléatoire contient beaucoup d'informations, son entropie est forte.
L'entropie (en bits) est calculée par la formule suivante :

$$H = -\sum_{i=1}^{N} p_i \log_2(p_i) \qquad \textbf{I.1}$$

Oùp_i est la probabilité de présence du niveau de gris i, et N étant le nombre du niveau de quantification (N = 256 pour une image à niveau de gris codée sur 8 bits).

Plus l'entropie est faible, moins on a besoin de bits pour représenter les niveaux de gris des pixels de l'image.

I.3 La Compression d'images :

I.3.1 Définition

La compression consiste à réduire la taille physique de blocs d'informations contenues dans une image. Elle s'appuie sur l'analyse du contenu de l'image et tire profit de son organisation interne, afin d'en éliminer les données redondantes qu'elles soient temporelles, spatiales ou statistiques.
Elle se fait dans l'objectif de rendre la taille de l'information compatible au débit d'un canal de transmission ou à la capacité de stockage d'un support ou de diminuer son temps de transmission.

I.3.2 Paramètres de performance des algorithmes de compression [1] [2] [17]:

I.3.2.1 Taux de compression :

Le taux de compression donne une mesure de performance des méthodes de compression des images. Etant donné que l'objectif d'une compression est de minimiser la quantité d'informations nécessaire à la représentation d'une image. On peut définir une quantité appelée rapport de compression (Rc) comme suit:

$$\mathrm{Rc} = \frac{Taille\ de\ l'image\ originale\ en\ bits}{Taille\ de\ l'image\ compréssée\ en\ bits} \qquad \textbf{I. 2}$$

Le taux de compression en pourcentage (τ_c) est donné par :

$$\tau_c = \left(1 - \frac{Taille\ de\ l'image\ compressée}{Taille\ de\ l'image\ originale}\right) \times 100 \qquad \textbf{I. 3}$$

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

$$R_{c(bpp)} = \frac{nombre\ de\ bits\ codés}{taille\ de\ l\ image\ originale(nombre\ de\ pixels)} \qquad \textbf{I. 4}$$

Notons, que pour les approches avec pertes, le taux de compression ne permet pas d'évaluer la performance de la compression. En effet, dans ce cas il est nécessaire d'introduire d'autres paramètres d'évaluation pour mesurer, la distorsion introduite et la vitesse du codage et du décodage.

I.3.2.2 La Distorsion :

L'information perdue entre le signal original et le signal décodé en fin de chaîne de compression, 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.

A) Erreur Quadratique Moyenne :

Appelée en Anglais MSE (Mean Square Error), elle a pour expression pour une image de taille $M * N$ donnée par la formule suivante :

$$MSE = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} \left(I_{(i,j)} - I'_{(i,j)} \right)^2 \qquad \textbf{I.5}$$

Avec $I(i, j)$: représente l'image originale ; $I'(i, j)$: représente l'image dégradée ; M et N sont le nombre de lignes et de colonnes d'une image.
Une grande valeur de MSE signifie que l'image est de mauvaise qualité.

B) Rapport signal sur bruit crête (PSNR) :

Le PSNR est défini comme suit :

$$PSNR = 10 \log_{10} \frac{(dynamique\ de\ l\ image)^2}{MSE} \qquad \textbf{I. 6}$$

I.3.2.3 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. Cette qualité sera cependant moins cruciale dans des applications visant l'archivage de données.

Lors de la compression irréversible, on est soumis à une contrainte qui est la dégradation plus au moins importante de la qualité de l'image originale, pour parier à ce problème on est amené à faire un compromis entre la taille et l'aspect visuel de l'image. Pour cela le choix de la méthode de compression dépend du type d'image et de son domaine d'application.

I.4 Méthodes de compression [17] :

La compression d'images fait appel à une variété d'algorithmes de codage qui exploitent les différents types de redondances existantes dans celle-ci. Le choix et l'association de ces algorithmes se fait en fonction des applications visées et des débits souhaités. On distingue deux grandes catégories d'algorithmes de compression :

- Ceux dits «sans pertes» ou réversibles.
- Ceux dits «avec pertes» ou irréversibles.

La figure I-1, illustre le schéma général de l'étape de la compression d'image :

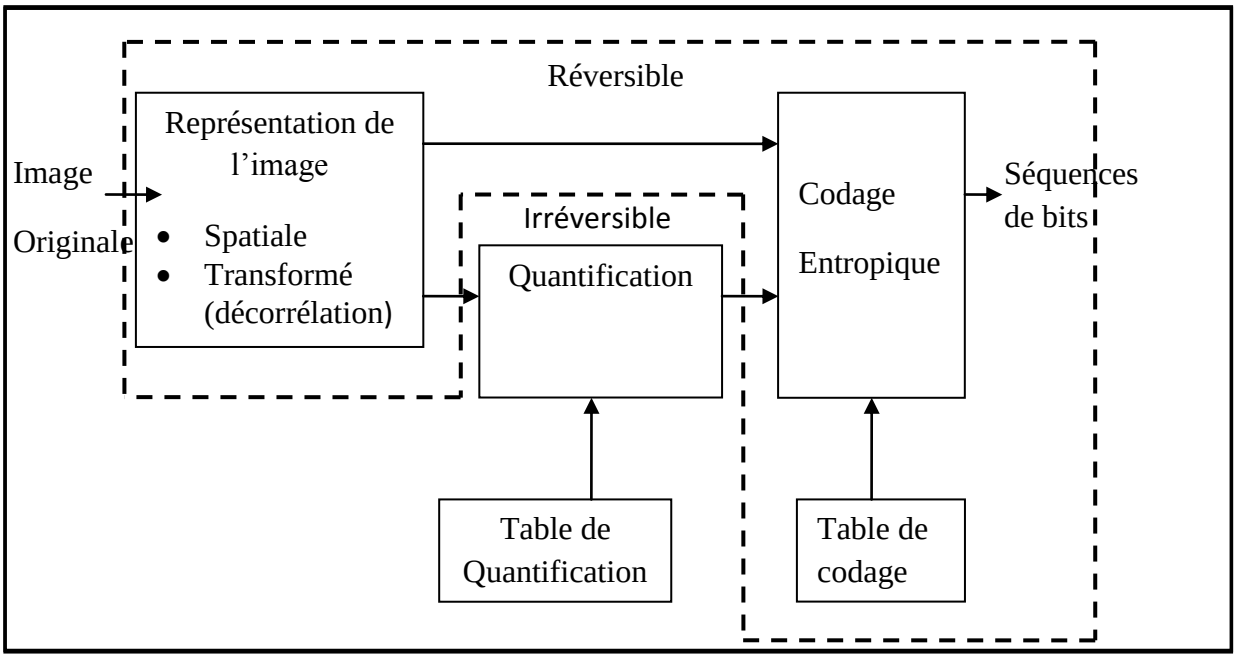


Figure I-1 : schéma général de l'étape de la compression.

I.4.1 Compression sans perte [1] [6] [17] :

I.4.1.1La Décorrélation :

La décorrélation a pour objectif de réduire les redondances statistiques de l'image et ainsi réduire les entropies d'ordre faible. Par la suite, cela permet d'utiliser t un codage entropique, c'est-à-dire le traitement des coefficients un par un pour la compression. Cette étape est réversible (sans pertes ou quasiment sans pertes). Elle fait intervenir une transformée. Par exemple, la DCT est utilisée dans le standard de compression d'images JPEG ou la transformée en ondelettes qui est à la base du standard JPEG2000.

La notion de codage sans perte d'une source encore appelé codage entropique, a pour objectif d'atteindre une limite théorique du gain de compression caractérisée par l'entropie de la source.

I.4.1.2 Codage entropique :

De nombreux algorithmes de compression sans pertes ont été mis en œuvre pour coder une source *S* avec la contrainte d'obtenir des mots de code de longueur moyenne aussi proche de son entropie que possible. Un codeur entropique permet de compresser une séquence de symboles en se basant sur leur probabilité (a priori) d'apparition. Chaque symbole se voit assigner une nouvelle représentation (dépendent de sa probabilité d'apparition) de manière est ce que le codage de la source s'approche au plus de son entropie.

Certains algorithmes exploitent les statistiques des symboles en faisant l'hypothèse d'indépendance statistique. Ces codeurs (tel que le codage de Huffman, de Shannon-Fano et le codage arithmétique) sont limités par la valeur de l'entropie d'ordre zéro de la source.

D'autres codeurs (tel le code de Lempel-Ziv et le codeur par longueur de séquence) utilisant les informations conjointes des réalisations des événements de la source. Ils sont limités par la valeur de l'entropie d'ordre supérieure, qui est plus faible que l'entropie d'ordre zéro. Ces derniers codeurs ont de meilleures performances lorsque les réalisations de la source ne sont pas indépendantes. En supposant une source stationnaire, la valeur de l'entropie baisse au fur et à mesure qu'elle est calculée avec un ordre élevé. Ainsi, les codeurs performants exploitent les probabilités conjointes avec un ordre de plus en plus élevé. Cette stratégie est cependant limitée par la complexité des algorithmes mis en œuvre.

I.4.1.2.1 Codage de Shannon-Fano :

C'est la première méthode largement utilisée, elle est basée sur la connaissance de la probabilité d'apparition de chaque symbole dans un message. Un arbre de Shannon-Fano est construit en fonction d'un algorithme spécifique conçu pour définir une table de codes efficace. L'algorithme comprend les quatre étapes suivantes :

1ére étape : classer les fréquences relatives d'occurrence de chaque symbole par ordre décroissant en deux parties, le total des compteurs de fréquences de la moitié supérieure devant être aussi proche que possible du total de la moitié inférieure. Poursuivre l'arborescence jusqu'à ce que toute fréquence soit isolée.

2ème étape : partitionner la table des fréquences en deux parties, le totale des compteurs de fréquence de la moitié supérieur devant être aussi proche que possible du total de la moitié inférieur. Poursuivre l'arborescence jusqu'à ce que toute fréquence soit isolée.

3ème étape : Attribuer dans l'arborescence le bit « 0 » aux moitiés supérieures de l'arborescence et le bit « 1 » aux moitiés inférieures.

4ème étape : Attribuer aux symboles, les codes binaires correspondant aux bits de description de l'arborescence.

Considérons l'exemple suivant d'une séquence S, à laquelle on a appliqué les différentes étapes ci-dessus :

S : AAADDDCAACCBBBEEEAAAAEECCCAADDDBBBBAAAA

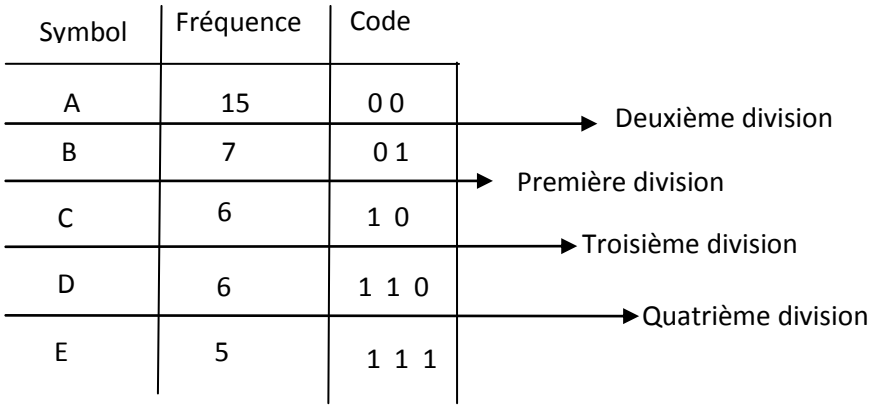


Figure I-2 : codage de Shannon-Fano

I.4.1.2.2 Codage de Huffman :

Le codage Huffman a été proposé par David Huffman en 1952. C'est une méthode statistique basée sur l'attribution d'un mot de code binaire pour chaque symbole de la chaine à compresser. La longueur des mots du code des symboles est variable. Les symboles ayant la probabilité d'apparition forte sont codés avec des chaînes de bits plus courtes, tandis que les symboles dont la probabilité d'apparition est faible sont codés par des chaînes plus longues.
Le codeur de Huffman est un arbre binaire ordonné par tous les symboles et par leurs fréquences d'apparition Les deux symboles les moins fréquents de la chaine sont reliés par leurs 'Parents' en faisant la somme de leurs fréquences. Les symboles prennent alors les valeurs 0 et 1. Le processus est répété sur l'ensemble des symboles jusqu'à ce qu'il ne reste qu'un seul symbole parent formant la racine de l'arbre binaire. L'opération inverse est utilisée pour le décodage.

Exemple de codage :

Soit la source S de l'exemple, de la figure I-3, ayant un alphabet de 05 symbole dont les probabilités sont respectivement : $p_i = \{0.39, 0.18, 0.15, 0.15, 0.13\}$; i = 0….4

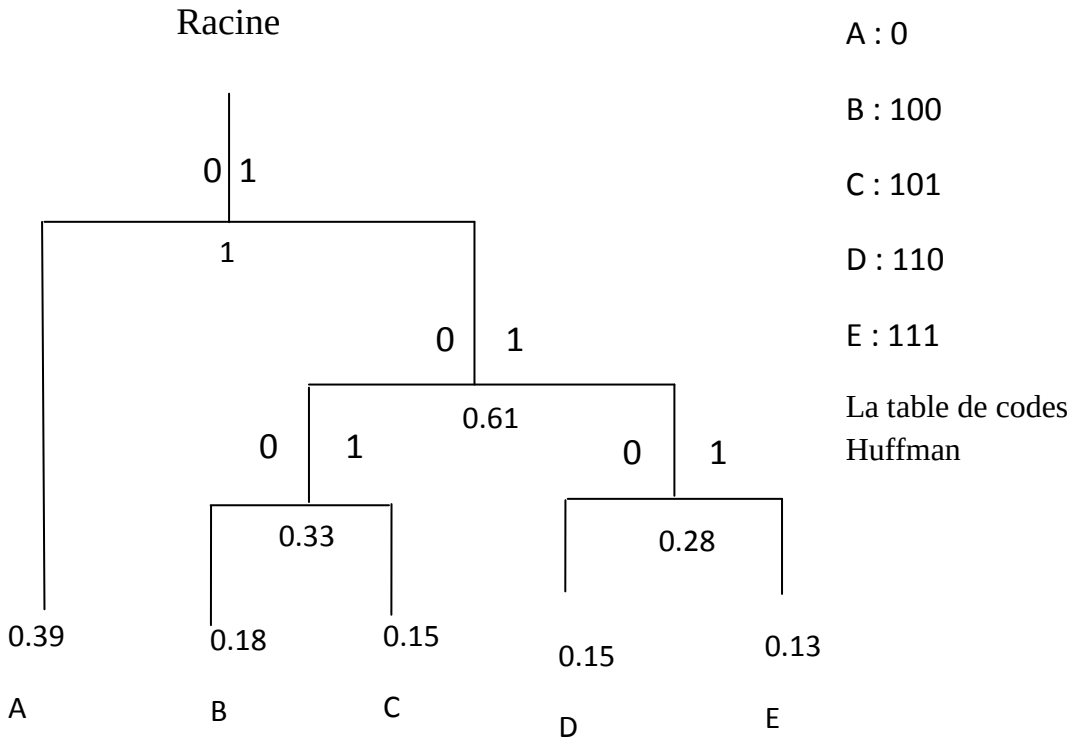


Figure I-3 : Exemple d'un codage de Huffman

I.4.1.2.3 Codage arithmétique :

Très prisé pour ses performances, le codage arithmétique se singularise par sa capacité à coder chaque symbole sur un nombre non entier de bits. En réalité, il n'assigne pas un mot de code à chaque symbole mais il associe un point de l'intervalle [0,1] à un ensemble de symboles. Le principe repose sur le découpage de l'intervalle [0,1]. Chaque symbole se voit attribuer une partition de l'intervalle dont la taille est égale à sa probabilité d'occurrence. L'ordre de rangement est mémorisé pour être utilisé lors du décodage.

Le codage arithmétique est généralement plus performant que le codage de Huffman. Il tend vers la limite inférieure théorique de Shannon. Il permet d'atteindre des taux très proches de l'entropie théorique. Cependant il est gourmand en ressources et nécessite de connaitre à priori l'intégralité du signal avant de pouvoir procéder au codage.

Algorithme de codeur arithmétique:

Les différentes étapes de l'algorithme de codage sont :

• L'initialisation : Nous affectons à chaque symbole une plage d'intervalle dont la longueur est égale à sa probabilité d'apparition fournie par le module. Les bornes externes de l'intervalle initial sont zéro et un.

• Le traitement du message : Nous initialisons un intervalle de travail en prenant comme bornes 0 et 1. Le premier symbole est représenté par la plage qui lui est affectée à l'étape 1.

Chaque symbole suivant restreint davantage l'intervalle et il est représenté par sa plage relative dans la plage précédente. Ainsi le flot de données est traduit par un nombre contenu dans la dernière plage calculée.

• On rajoute un symbole spécial pour déterminer la fin du message où l'on donne la longueur du flot avec le message codé pour permettre au décodeur de déterminer la fin du message.

I.4.1.2.4 Codage RLC (Run Length Coding):

Il est plus intéressant de coder un message contenant une suite d'éléments répétitifs par *"un couple répétition et valeur"* au lieu de coder seulement le message lui-même.
Le codage RLC consiste en effet à coder un élément du message par sa valeur de répétition.
Pour autant, s'il n'y a pas de répétition d'éléments, la technique ne donne pas de résultats satisfaisants. Notons que, le codage RLC introduit un système de contrôle (bits) pour réaliser l'encodage. Il réalise le codage s'il y a une répétition successive d'éléments (minimum égal à 4). Dans le cas contraire, il insert les bits contrôle (00).

I.4.1.2.5 Codage Lempel – Ziv [1] :

C'est une technique de codage à base d'un dictionnaire ou nous mémorisons les chaînes qui se répètent dans ce dictionnaire. Ensuite, on remplace les chaînes mémorisées par leur adresse(ou indice) construite dans le dictionnaire. L'élaboration du dictionnaire ainsi que la recherche de chaîne répétée sont différentes selon la version de l'algorithme. Il en existe trois versions.
– LZ77, utilisé pour l'archivage, la recherche s'effectue par une fenêtre glissante.
– LZ78, utilisé dans la compression d'image, la recherche s'effectue sur tout le fichier. La taille du dictionnaire est limitée en fonction du mode de codage (16, 32, ou 64 bits) ;
– LZW, introduite en 1984, C'est un algorithme utilisé dans la compression et la décompression. Il est basé sur la multiplicité des occurrences de séquences de caractères dans la séquence à encoder. Son principe est de substituer des motifs par un code d'affectation en construisant au fur et à mesure un dictionnaire. Il est rapide en compression et en décompression et ne nécessite pas de virgule flottante.

I.4.2 Compression avec perte [1] [11] [17] [22] :

Les méthodes avec perte (lossy) ou irréversibles sont des méthodes qui tirent parti d'une corrélation (ou redondance) existante dans l'image. L'information perdue est due à l'élimination de cette redondance, ceci rend possible une compression plus importante. La perte d'information est toujours discutable et nous nous posons alors la question de la limite acceptable. Cette limite est définie par le type d'application.

La quantification est un des mécanismes utilisé dans les algorithmes de compression, qui produit des pertes d'information.

I.4.2.1 La quantification :

La quantification est l'une des sources de perte d'information dans le système de compression. Son rôle est en effet de réduire le nombre de bits nécessaire à la représentation de l'information. Elle est réalisée avec la prise en compte de l'aspect psycho visuel (l'œil humain), ce qui permet de déterminer la distorsion tolérable à apporter au signal à coder. On distingue deux sortes de quantification : scalaire (QS) et vectorielle (QV).

- **Quantification scalaire (QS) :**

La quantification scalaire est réalisée indépendamment pour chaque élément. D'une manière générale, on peut la définir comme étant l'association de chaque valeur réelle x, à une autre valeur q qui appartient à un ensemble fini de valeurs. La valeur q peut être exprimée en fonction de la troncature utilisée : soit par l'arrondi supérieur, l'arrondi inférieur, ou l'arrondi le plus proche. On l'appelle le pas de quantification Δ. Il représente l'écart entre chaque valeur q. Arrondir la valeur x provoque une erreur de quantification, appelé le bruit de quantification.

La procédure suivante définit la réalisation d'une quantification scalaire. Soit *X l'ensemble* d'éléments d'entrée de taille N.

1. Echantillonner X en sous-intervalles $\{[x_n, x_{n+1}[, n \in \{0...N - 1\}\}$.
2. Associer à chaque intervalle $[x_n, x_{n+1}[$ une valeur q.
3. Coder une donnée $x \in X$ par q si $x \in [x_n, x_{n+1}[$.

Si Δ est constant, on parle d'une quantification uniforme. Sinon elle est dite non-uniforme.

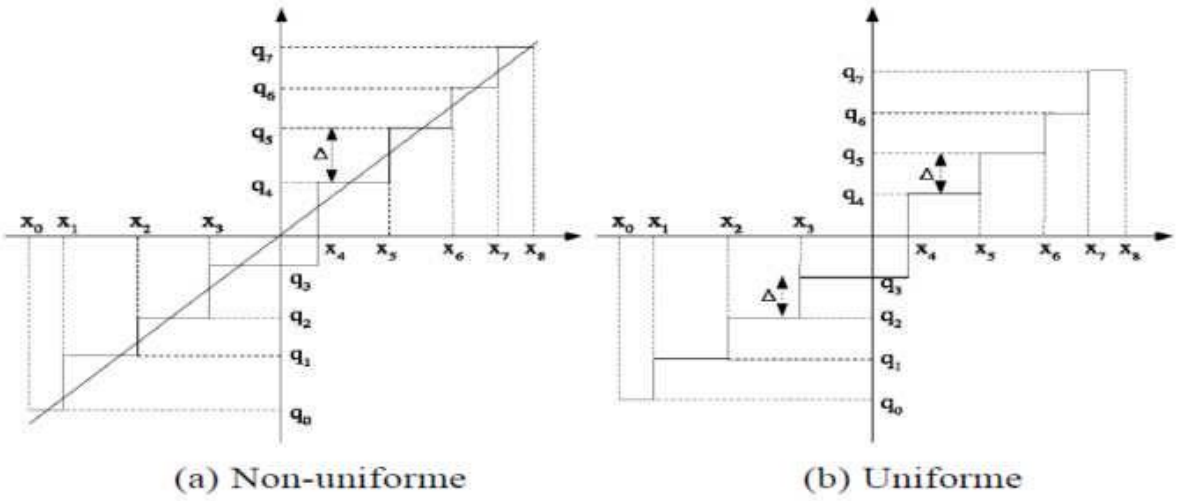


Figure. I.4 : Quantification scalaire

- **Quantification vectorielle (QV) [11] :**

La quantification vectorielle est plus complexe à mettre en œuvre car il faut préalablement engendrer un ensemble de vecteurs de référence appelé dictionnaire (code book), ce qui se fait à l'aide d'un algorithme d'apprentissage que l'on applique à un ensemble d'images. La quantification consiste alors à décomposer l'image en vecteurs de taille identique à ceux du dictionnaire, à rechercher pour chaque vecteur de l'image le plus proche dans le dictionnaire et à le remplacer par l'indice dans le dictionnaire du vecteur associé.

Il faut remarquer que la quantification vectorielle donne souvent de meilleurs résultats que la quantification scalaire**.** Récemment, des méthodes de quantification donnant de meilleurs résultats que les deux techniques citées précédemment ont été introduites. En effet ces méthodes sont avantageuses grâce au contenu spatial de la transformée en ondelettes de l'image, ceci en quantifiant les coefficients par approximations successives à travers les sous bandes de même orientation.

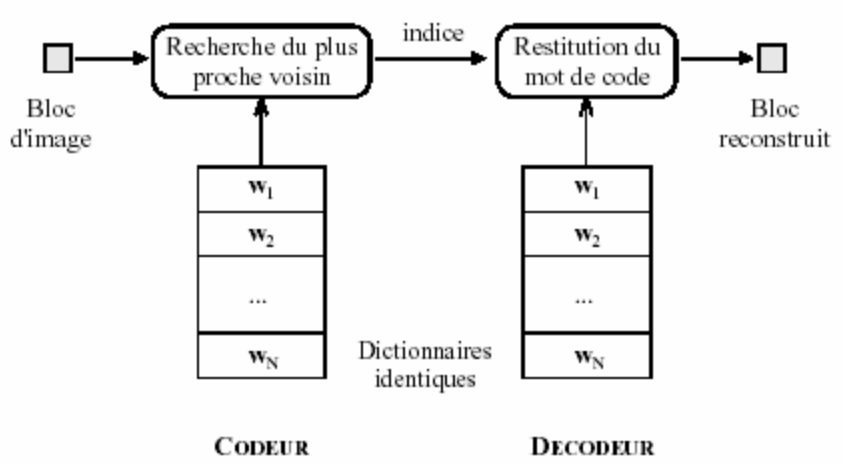


Figure.I.5. La quantification vectorielle

I.4.2.2 Codage fractal [17] :

Cette technique relativement nouvelle est basée sur une idée très originale ; en supposant qu'une image possède des propriétés fractales, reconstruisons-la à partir d'elle-même au travers des transformations mathématiques affines contractantes [17].

Plusieurs phénomènes naturels sont de type fractal. Par exemple, les nuages, les rives d'un cours d'eau, les arbres, …etc. Une image météorologique peut être considérée comme une figure géométrique extrêmement irrégulière et composée de plusieurs types de nuages. Cette figure, présente les mêmes irrégularités à plusieurs échelles, on dit alors qu'il y a auto similarité entre les blocs ou parties (les nuages) qui constituent l'image à différentes échelles. En conséquence, les images satellitaires peuvent être efficacement simulées et codées par une technique fractale.

Le principe du codage fractal est de partitionner une image en blocs destination (*range bloc*) et blocs source (*domain bloc*). Il s'agit ensuite de rechercher pour chaque bloc source considéré après transformations contractantes (réduction, symétrie et rotation), un bloc destination qui lui ressemble. En effet, le codage fractal d'une image consiste à la représenter par un système de fonctions itérées (*IFS*). Le code IFS est constitué des coordonnées des blocs source et des paramètres des transformations appliquées. Au décodage, on applique une procédure itérative, contenant le code IFS, sur une image initiale quelconque en remplaçant les blocs source par ceux transformés. Si certaines conditions sont vérifiées, l'image finale est une bonne approximation de l'image originale.

Le codage fractal est une méthode irréversible. L'inconvénient majeur de cette approche est le coût en temps de calcul pendant la procédure de codage.

I.4.2.3 Codage par transformation [11] :

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

- Transformation en cosinus discrète (TCD)
- Transformation en ondelettes (TOD).

Le principe d'un système de codage par transformation est le suivant :

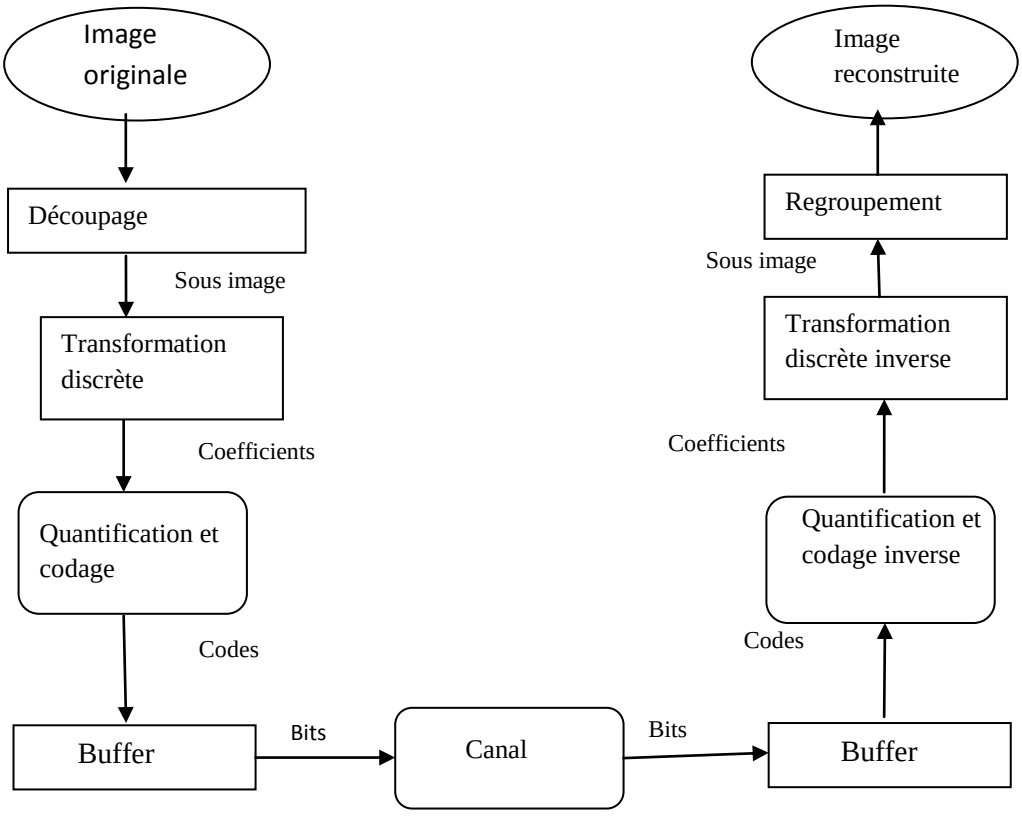


Figure I.6 : principe d'un système de transmission d'un codeur par transformation

I.5 Les Standards de compression :

I.5.1 La norme JPEG [8] :

La norme JPEG est un standard en compression d'images, elle a été finalisée vers le début des années 90. C'est une norme de compression d'images fixes, conçue à l'origine pour le monde de l'impression et de la photocomposition. JPEG accepte n'importe quelle définition des images et exploite les uniformités présentes à l'intérieur de chacune d'elles ; le codage est dit intra-image par conséquent, il élimine les redondances spatiales de ces images.

- **Transformation en cosinus discrète DCT [12] :**

C'est une méthode très utilisée dans tous les domaines d'imagerie y compris médicale. La matrice de transformation DCT est complètement dépendante de l'image, cette transformation est très utilisée pour l'exécution des algorithmes rapide en calcul.

La transformation cosinus discrète d'un bloc X de (n*n) pixels donne un autre bloc X_c de (N*N) coefficients, selon la formule suivante :

$$x(K,l) = \frac{4}{(N)^2} C(k,l) \sum_{K=0}^{N-1} \sum_{L=0}^{N-1} X(m,n) \cos[\pi \frac{(2m+1)K}{2N}] \cos[\pi \frac{2n+1)l}{2N}] \qquad \textbf{I. 7}$$

Avec : K, l =0, 1, 2…...N-1

C(0,0)=1/2

C(0,1) = C (k, 0)= $1/\sqrt{2}$

C (K, l)=$1/2$

Les techniques édifiées par la norme JPEG se divisent en deux classes : Méthodes de compression avec perte qui sont basées sur la DCT suivie d'une quantification et d'un codeur entropique, représentées sur la figure I-7.La seconde classe, concerne les processus de codage sans perte, cette classe de codeur n'est pas basée sur la DCT, mais sur le codage par prédiction suivi d'un codage entropique, représenté sur la figure I .8**:**

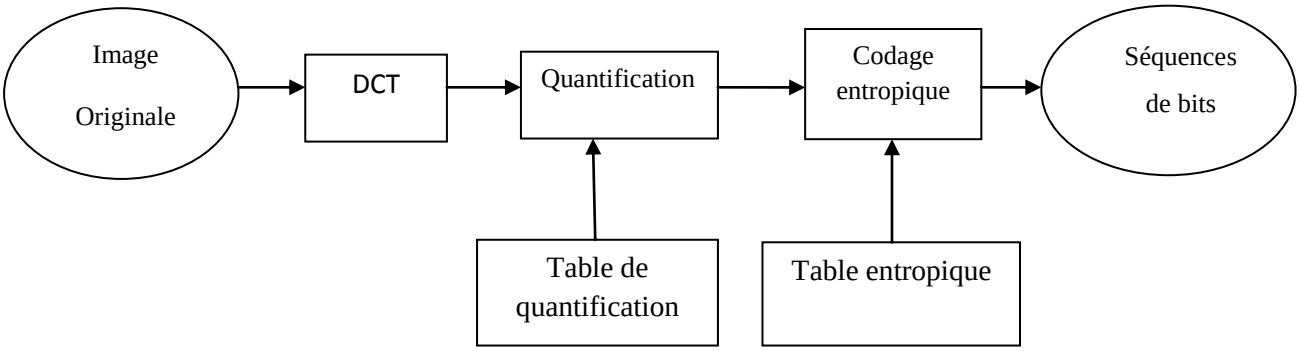


Figure I-7 : principe de compression JPEG avec perte

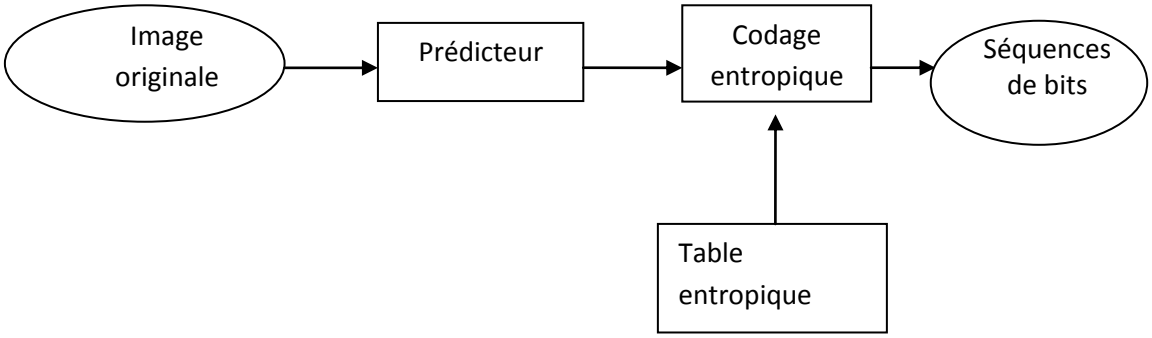
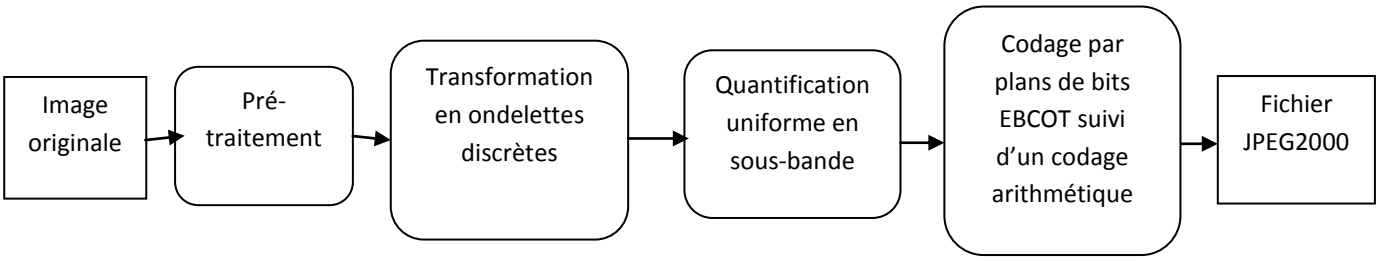


Figure I-8 : Principe de compression JPEG sans perte

I.5.2 Le standard JPEG2000 : [1] [17] [22] :

La norme JPEG2000 qui est le nouveau standard de compression d'images dans sa version basique, elle utilise une TOD (Transformée en Ondelette Discrète) sur chaque composante pour la réduction de la redondance spatiale.

Le codage et décodage d'une image au format JPEG2000 s'effectuent en quatre étapes principales : Les trois étapes les plus utilisées en compression d'image (Transformation, quantification, codage) plus une étape de prétraitement de l'image qui a pour but de rendre l'opération de codage plus efficace. Ces différentes étapes sont illustrées sur la figure I.9.



FigureI-9 : Diagramme de chaine de codage de l'algorithme JPEG2000

Elle utilise un codage par plans de bits, ainsi qu'un codeur arithmétique adaptatif, à modélisation de contexte. Le codeur utilisé est EBCOT (Embedded Bloc Coding Optimized Truncation) dont l'une des particularités est la "scalabilité", c'est-à-dire que l'on peut tronquer le flot de bits n'importe où, et reconstituer une image avec une qualité optimale par rapport au flot de bits qui a été utilisé par le décodeur. L'utilisation de la TOD permet de s'affranchir de l'effet de blocs observé à bas débits avec le standard JPEG.

Des discussions et des tests se poursuivent pour la finalisation complète de ce nouveau standard qui offre des perspectives et des applications à une échelle plus large que JPEG.

La norme JPEG2000 est constituée de 12 parties relativement indépendantes :

Partie 1(Core Coding System)

Spécifie le « cœur » (ou fonctionnalités minimales) du codec JPEG 2000 Elle définit la technologie minimale de l'algorithme de décodage ainsi que le format du *codestream* devant être compris par tous les produits se réclamant compatibles avec la norme. En outre, cette partie introduit le format de fichier JP2 qui permet d'associer au *codestream* des informations additionnelles sur l'image. Enfin, elle a pour but de couvrir une grande majorité des applications touchant au domaine de l'imagerie numérique (archivage, diffusion sur Internet...).

Partie 2 (Extensions)

Celle-ci définit des extensions à la partie précédente afin d'améliorer les performances ou augmenter le nombre de fonctionnalités. Elle est surtout dédiée à des applications plus spécifiques telles que le codage de données hyper-spectrales (ex : images satellite).

Partie 3 (*Motion JPEG2000*)

Elle définit un format de codage de séquences audiovisuelles et d'animations en tant que succession d'images indépendantes compressées par un système compatible avec JPEG2000 Partie1.

Partie 4 (Conformance Testing)

Spécifie la procédure à utiliser pour les tests de conformité.

Partie 5 (Reference software)

Fournit aux développeurs une implémentation logicielle du « cœur » de la norme. Il y a en tout deux implémentations. La première, appelée JJ2000, est en Java et élaborée dans le cadre d'un partenariat entre l'EPFL, Canon Research France et Ericsson, alors que la seconde, appelée JasPer, est écrite en C.

Partie 6 (Compound image file format)

Elle spécifie un format de fichier pour des documents numérisés (i.e. contenant à la fois du texte, des photographies, des graphiques...).

Partie 7 partie abandonnée.

Partie 8 (JPSEC)

Elle est consacrée aux problèmes de sécurité, elle définit des mécanismes pour l'accès conditionnel, la protection des données, de la propriété intellectuelle et des droits.

Partie 9 (JPIP) Elle est consacrée aux outils d'interactivité, aux APIs et protocoles. Spécifie un protocole client-serveur pour communiquer efficacement via des réseaux.

Partie 10 (**JP3D**, Extensions for three-dimensional data)

Fournit des extensions pour manipuler des images 3D.

Partie 11 (JPWL, Wireless)

Fournit des outils de traitement des erreurs pour les transmissions sans fil.

Partie 12 (ISO Base Media File Format)

Définit un format de fichier média commun utilisé par Motion JPEG 2000 et MPEG 4.

I.6 Les algorithmes de codage par plans de bits :

I.6.1 L'algorithme de codage EZW [1] :

C'est le premier codeur en sous-bande par plans de bits appelé « zerotree » à avoir été introduit par Shapiro**.** L'idée c'est de trouver le meilleur ordre de transmission progressive des coefficients de représentation en ondelettes sur l'image codée tout en apportant d'excellentes performances débit/distorsion par rapport à la norme JPEG. Il procède au regroupement des coefficients non significatifs sous forme d'arbre de zéros (*zerotree).* Un zerotree est un quadruple-arbre dont tous les nœuds enfants sont égaux ou plus petits que les nœuds parents. L'arbre est codé avec un symbole unique et reconstruit par le décodeur comme quadruple-

arbre rempli de zéros. Nous devons insister sur le fait que la racine doit être plus petite que le seuil par rapport auquel les coefficients d'ondelettes sont comparés, sinon, ce coefficient ne serait pas considéré comme base de zerotree.

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.

I.6.2 L'algorithme de codage SPIHT (Set Partitionning In Hierarchical Trees)[1] [23] :

L'algorithme SPIHT a été proposé par Saïd et Pearlman en 1996 pour la compression avec et sans perte. Il a été étendu aux images 3D pour la vidéo et pour la compression d'images volumiques. La modification majeure de SPIHT par rapport à EZW réside dans la mise à jour d'une 3emeliste permettant de créer des ensembles non signifiants de grande taille. Ces ensembles non signifiants permettent de connaître l'état d'une descendance même si le coefficient n'est pas la racine d'un zerotree. L'algorithme SPIHT effectue un partitionnement récursif de l'arbre de manière à déterminer la position des coefficients significatifs dans la descendance du coefficient considéré. Il repose sur les mêmes concepts : codage progressif par plan de bits et utilisation des dépendances hiérarchiques entre les coefficients des différentes sous-bandes. Cependant, un nouveau protocole de dépendance entre les coefficients est défini ; et tout cela, sera détaillé dans le chapitre 3.

I.6.3 L'algorithme de codage EBCOT [4] [22]:

C'est le codeur adaptatif de JPEG2000, introduit par Taubman, le principe de base de l'EBCOT est le suivant : Après la quantification scalaire, les coefficients issus des différentes sous-bandes de la transformée en ondelettes sont rangés en blocs, appelés code-blocs, de forme rectangulaire, de taille paramétrable, leurs hauteurs et largeurs correspondant à une puissance de deux, et le produit largeur hauteur ne devrait pas dépasser 4096 (64 x 64) avec une hauteur et largeur minimale de 4. Chaque code-bloc est codé indépendamment, sans aucune référence aux autres code-blocs de la même sous-bande ou d'une autre. Ce codage

indépendant offre des avantages importants, comme un accès spatial aléatoire au contenu de l'image, calcul parallèle durant le codage ou décodage.

Le codage de chaque plan de bits s'effectue de manière adaptative. De nombreuses expériences numériques ont mené au choix d'un certain nombre de contextes (correspondant aux configurations des bits voisins dans le plan de bits précédents) qui capturent de façon compacte la structure locale des coefficients d'ondelettes. En construisant progressivement les histogrammes des bits conditionnés par rapports aux différents contextes, EBCOT est capable de coder les coefficients en ondelettes de façon très efficace.

Cet algorithme est considéré comme l'un des plus performants algorithmes de compression d'images aussi bien en termes de compression pure qu'en termes de fonctionnalités. C'est pourquoi cet algorithme a été choisi pour devenir le cœur du format JPEG2000.

Conclusion :

Nous avons développé dans ce chapitre les notions et le principe de base utilisés en compression d'images. Nous avons abordé aussi les paramètres utilisés pour l'évaluation des performances des méthodes de compression ; à savoir, pour les méthodes réversibles, la meilleure performance consiste à maximiser le taux de compression, pour les méthodes irréversibles, il faut maximiser le taux tout en minimisant les pertes.

Nous avons également décrit brièvement les Standards JPEG et JPEG2000. Ce dernier a pu surpasser son prédécesseur JPEG par sa qualité d'images à des taux de compression très élevée grâce à la transformée en ondelettes utilisée dont le principe sera exposé dans le prochain chapitre.

Introduction :

Analyser un signal à partir de son graphe uniquement est loin de permettre d'accéder à toutes les informations qu'il contient. Il est souvent nécessaire de le transformer, c'est à dire d'en donner une autre représentation, qui fasse apparaître plus clairement telle ou telle de ses caractéristiques.

Jusqu'au milieu des années 1980, les outils habituels du traitement du signal reposaient essentiellement sur l'analyse de Fourier parfaitement adaptée au traitement des signaux stationnaires.

La Transformée de Fourier permet de décrire la répartition des composantes fréquentielles du signal sans nous renseigner sur les instants de l'apparition de celles-ci. Les renseignements fréquentiels ainsi obtenus le sont au détriment de la description temporelle explicite du signal. Cette méthode ne convient donc pas à tous les types de signaux, notamment des signaux non-stationnaires qui se caractérisent par l'apparition d'événements transitoires. Elle est aussi insuffisante pour mettre en évidence les caractéristiques évolutives du signal. Ni la description temporelle et ni la description fréquentielle seules permettent de décrire l'évolution temporelle du contenu spectral d'un signal. Une autre description est donc nécessaire, combinant les deux descriptions, la représentation « temps-fréquence ».

C'est la Transformation de Fourier à Fenêtre GlissanteTFFG, introduite par le physicien D. GABOR dans les années quarante, qui suggérait de rendre locale l'analyse de Fourier en s'aidant d'une " fenêtre " spatiale régulière et bien localisée, ce qui signifie qu'elle est nulle en dehors d'une certaine zone, qui constitue son support.

Cependant, la méthode de la transformée de Fourier Glissante présente l'inconvénient majeur d'avoir une fenêtre de longueur fixe, car la taille de la fenêtre détermine le spectre de fréquences qui peut être observé. Il faut donc une fenêtre large pour observer les basses fréquences, mais aussi une fenêtre assez étroite pour localiser les hautes fréquences. La taille de la fenêtre est donc le résultat d'un compromis, qui se fait toujours au détriment de la détection ou de la localisation de certaines fréquences dans le cas d'une fenêtre fixe.

La théorie des ondelettes allait contourner cette difficulté, ce nouveau type d'analyse, proposé par A. GROSSMAN et J. MORLET en 1981 pour l'analyse de signaux sismiques, fut baptisé « ondelette » par Y. MEYER, est basée sur un concept quelque peu différent du concept de fréquence, le concept d'échelle.

Les ondelettes allaient rapidement rejoindre certaines techniques utilisées dans la représentation des images grâce aux travaux de I. DAUBECHIES et de S. MALLAT. Sous l'impulsion d'une technologie où l'imagerie numérique prend une place de plus en plus grande, les problèmes de codage (pour la transmission), de compression (pour l'archivage), d'estimation (pour la restauration d'images floues ou entachées de bruit) et de détection (pour extraire l'information en vue d'une décision) étaient autant de domaines où les ondelettes furent rapidement mises à contribution.

II. Eléments de la théorie des ondelettes :

II.1 Définition des ondelettes [17] [18] [23] :

Les ondelettes sont une famille de fonctions engendrées par translation en temps et dilatation en fréquence à partir d'une même fonction Ψ, appelée ondelette mère. Elles sont normalisées suivant l'équation :

$$\Psi_{a,b}(t) = |a|^{-1/2}\,\Psi\left(\frac{t-b}{a}\right) \qquad (2.1)$$

Avec Ψ(t) : fonction bien localisée en temps et en fréquence.

$$\begin{cases} (a,b) \in \Re^2 \\ a \neq 0 \end{cases}$$

a : paramètre de dilatation.

b : paramètre de translation.

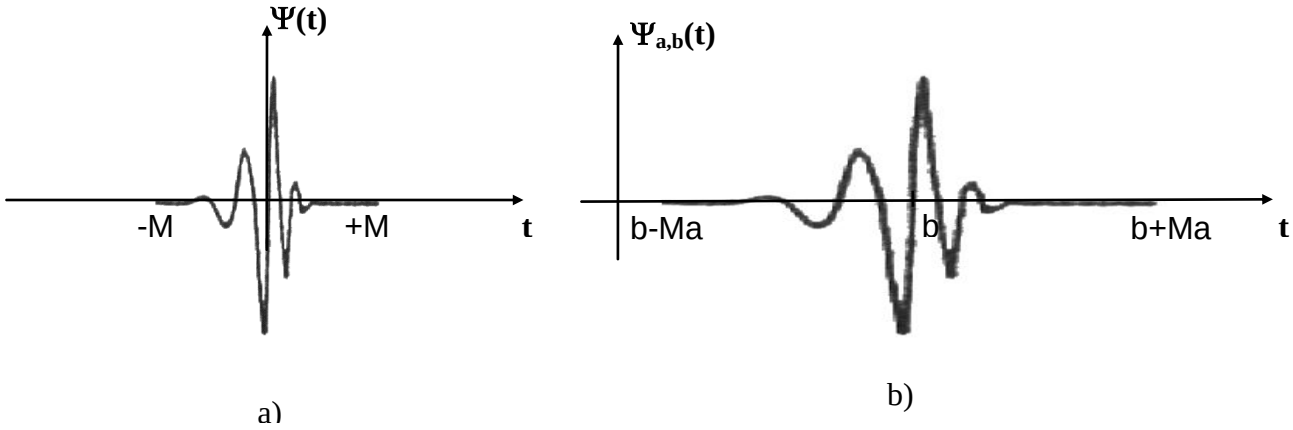


Figure II.1 : a) Ondelette analysante, b) Ondelette dilatée et translatée

Une meilleure localisation fréquentielle par l'ondelette entraîne une moins bonne localisation spatiale ou temporelle et réciproquement.

D'après le principe d'incertitude d'Heisenberg, aucun signal ne peut être simultanément et arbitrairement localisé en temps et en fréquence. L'amélioration de la résolution fréquentielle n'est possible qu'au détriment de la résolution temporelle et vice versa. Un exemple de boîtes de Heisenberg d'atomes d'ondelettes est donné à la figure II-2. L'écart-type en temps σ_t est proportionnel à l'échelle s. L'écart-type en fréquence $\sigma_\omega$est inversement proportionnel à l'échelle s.

L'atome de base de la transformée en ondelette est ψ, Si le centre de fréquence de ψ est ξ, le centre de fréquence de la fonction dilatée est en ξ/s .

La transformée en ondelettes a donc une résolution temps-fréquence qui dépend de l'échelle *s*.

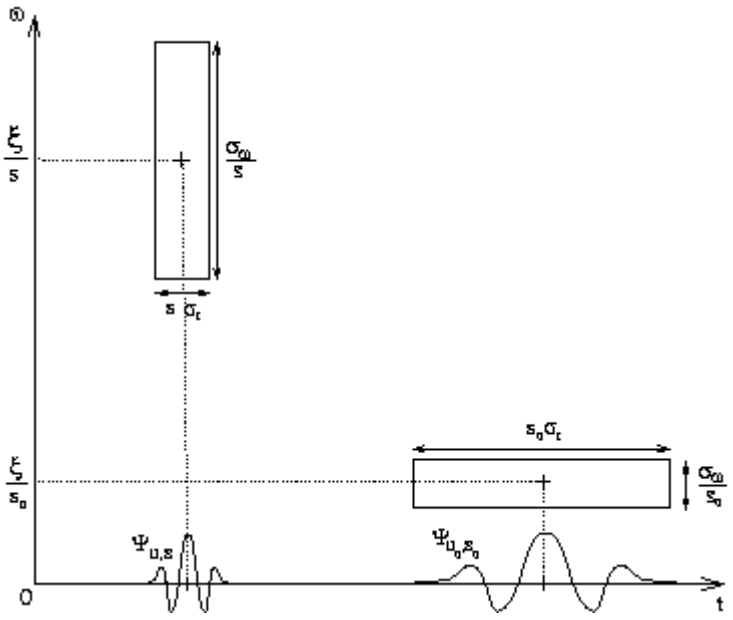


Figure II.2 : un exemple de boîtes de Heisenberg d'atomes d'ondelettes

- **La dilatation :**

Le facteur d'échelle **a** relie la notion de fréquence.

Plus **a** est grand, plus l'ondelette est dilatée, par conséquent la valeur de **a** est inversement proportionnelle à la fréquence.

- **La translation :**

Le décalage **b** relie la notion de position temporelle.

La translation correspond au déplacement de l'ondelette dans le domaine temporel.

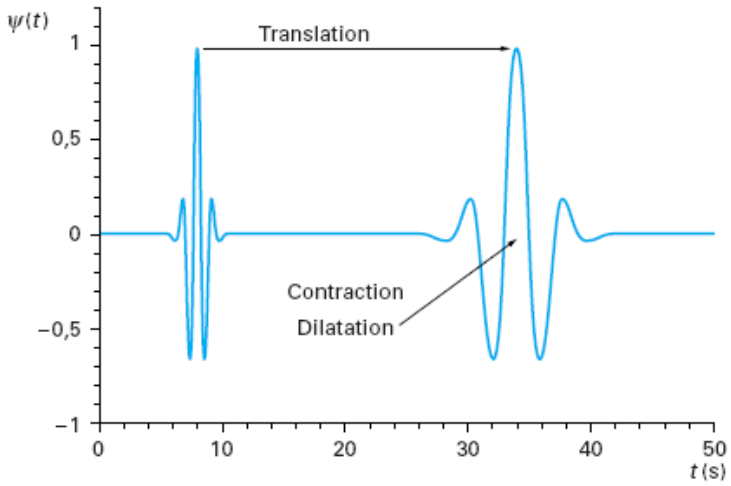


Figure-II-3dilatation et translation

II.2. Condition d'admissibilité :

La transformée en ondelettes doit permettre de reconstruire le signal sans perte d'informations à partir de sa transformée. Ceci n'est réalisable que si certaines conditions dites d'admissibilité sont réalisées :

$$* \qquad \int_{-\infty}^{+\infty} \psi(x)\,dx = 0 \qquad (2.2)$$

$$** \qquad \int_{-\infty}^{+\infty} \frac{\left|\hat{\psi}(\omega)\right|^2}{|\omega|}\,d\omega \quad = k < +\infty \qquad (2.3)$$

$\hat{\Psi}(w)$ est la TF de $\Psi(t)$.

Il faut donc s'assurer que la transformée de fourrier de l'ondelette a la fréquence continue (w égale à 0) doit être nul pour que la condition soit satisfaite.

Si les conditions sont satisfaites donc la transformée en ondelettes est inversible, elle est donnée par la formule suivante :

$$f(x) = \frac{1}{k_\psi} \iint_{R^2} W_f(a,b)\psi\left(\frac{x-b}{a}\right)\frac{dadb}{a^2} \qquad (2.4)$$

Avec : $K\psi = 2\pi \int_{-\infty}^{+\infty} \frac{\left|\hat{\Psi}(w)\right|^2}{w} dw$ (2.5)

II.3. Propriétés des ondelettes [17] [19] [23] :

- **La symétrie :**

Cette propriété est très importante en traitement numérique des images, la seule base qui possède cette propriété dans le cas orthogonale est la base de Haar. Les filtres associes aux bases d'ondelette bi-orthogonale sont symétriques, c'est-à-dire phases linéaires. C'est pourquoi, les ondelettes bi-orthogonales ont été introduites et sont abondamment utilisées.

- **Régularité :**

La régularité de l'ondelette est importante en compression. Cette propriété permettant de localiser les singularités dans un signal. Elle se traduit sur les coefficients d'ondelettes par une amplitude importante, caractérisant une singularité dans le signal, et par la décroissance des valeurs de coefficients avec l'échelle de résolution. En effet, on désire alors obtenir des coefficients d'ondelettes les plus petits possibles (afin de les annuler), pour tout ce qui concerne les détails du signal ; la décroissance des coefficients en fonction de l'échelle est donc primordial. On peut noter qu'il existe un lien entre la régularité et les moments nuls d'une ondelette.

- **La compacité :**

La compacité de fonction d'échelle augmente la régularité de l'ondelette résultante. Mais ces convolutions ont pour effet d'accroître linéairement la taille du support d'ondelette. Les ondelettes de Daubechies, par exemple, sont orthogonales et à support compact. Elles ont été créées en garantissant une certaine régularité par l'annulation d'un nombre fixe (p) de ses moments.

Donc, on ne peut pas avoir d'ondelettes qui soient compacités et à support compact. Il y a par conséquent un compromis entre les propriétés de régularité et de décroissance à l'infini. Par ailleurs, la propriété de support compact permet de garantir une grande précision dans le calcul effectif des coefficients, car elle évite les problèmes de troncature dans le cas de support à durée infinie sur un support compact suffisamment étroit pour être considéré bien localisé dans le temps.

- **L'Orthogonalité :**

L'orthogonalité permet de minimiser la redondance, autorisant un codage efficace grâce aux faibles nombres de coefficients.

L'orthogonalité simplifie la reconstruction qui reste néanmoins possible même lorsque cette propriété est vérifiée.

La redondance (non orthogonalité) n'empêche pas la reconstruction, mais la rendre plus compliquée, elle donne cependant plus de robustesse dans les calculs et une meilleure précision de reconstruction.

- **Localisation :**

La localisation en temps et en fréquence peut se mesurer par la borne d'incertitude du principe de Heisenberg, qui assure une meilleure localisation. L'lorsque cette incertitude est atteinte, une mauvaise localisation induit un étalement de l'énergie du signal autour d'un instant moyen et d'une fréquence moyenne pour une échelle donnée.

II.4. Définition de la transformée en ondelettes continues (TOC) [17] :

La transformée en ondelettes d'une fonction est une représentation de cette fonction sur la base d'ondelettes.

La transformée en ondelettes continue est une décomposition du signal sur une famille de fonctions (ondelettes) localisées à la fois en temps et en fréquence et de taille variable. La taille de l'ondelette est inversement proportionnelle à la fréquence tout en vérifiant le principe d'incertitude d'Heisenberg.

La transformée en ondelettes continue d'un signal $f(x) \in L^2(R)$ est donnée par :

$$W_f(a,b) = \langle f, \psi_{ab} \rangle \qquad (2.6)$$

$$W_f(a,b) = a^{-\frac{1}{2}} \int_{-\infty}^{+\infty} f(x)\overline{\psi}\left(\frac{x-b}{a}\right)dx \qquad (2.7)$$

$\overline{\psi}$ représente le complexe conjugué de ψ.

Cette transformée possède la propriété de conservation de l'énergie, ce qui signifie qu'il n'ya pas de perte d'information entre la fonction et sa transformée.

$$\iint_{R^2} \left| W_f(a,b) \right|^2 \frac{dadb}{a^2} = k_\psi \int_{-\infty}^{+\infty} \left| f(x) \right|^2 dx \qquad (2.8)$$

Cette transformée, permet de passer de la représentation temporelle du signal :

$$f(x) = \int_{-\infty}^{+\infty} f(u)\delta(u-x)dx \qquad (2.9)$$

à une représentation temps-échelle bien adaptée pour l'analyse localisée des signaux dans le temps et la fréquence notamment lorsque les signaux à analyser sont irréguliers et non-stationnaires.

La condition (2. 2) signifie que : $\psi(x)$ est une fonction à largeur temporelle finie (fenêtre temporelle) possédant un caractère oscillatoire. On est donc bien en présence d'une **petite onde** : une ondelette.

La condition (2.3) ou condition d'admissibilité entraîne que :

$$\hat{\psi}(0) = 0 \text{ et } \hat{\psi}(\omega) << 1 \text{ pour } \omega \text{ au voisinage de 0.}$$

En autre:

$$\hat{\psi}(\omega) \xrightarrow[|\omega|\to\infty]{} 0$$

Car $\psi \in L^2(R)$, ψ agit donc comme la réponse impulsionnelle d'un filtre passe -bande.

La fonction ψ est aussi caractérisée par la propriété de régularité suivante :

$|\hat{\psi}(\omega)|$ doit décroître plus rapidement que $C(1+|\omega|)^{-\varepsilon-0.5}$ pour $|\omega| \to +\infty$

et $\varepsilon > 0$, C étant une constante réelle, et doit avoir un certain nombre de moments nuls.

- Transformée inverse

Tout comme la transformée de Fourrier, la transformée en ondelettes continue est inversible, elle admet une formule de reconstitution dans $L^2(R)$:

$$f(x) = \frac{1}{k_\psi} \iint_{R^2} W_f(a,b)\psi\left(\frac{x-b}{a}\right)\frac{dadb}{a^2} \qquad (2.10)$$

Dans le sens

$$\lim_{\varepsilon\to 0^+}\left(\frac{1}{k_\psi}\iint_{\substack{|a|>\varepsilon \\ b\in R}} W_f(a,b)\ \psi\left(\frac{x-b}{a}\right)\frac{dadb}{a^2}\right) = f(x) \qquad (2.11)$$

II.5 La transformée en ondelettes discrètes (TOD) [17]

En pratique, on a plus souvent affaire à des signaux discrets, mais même sans cela, on a intérêt à discrétiser les valeurs de a et b. Pour pouvoir être implémentée efficacement sur un calculateur numérique, Il ne suffit pas d'effectuer les calculs avec des versions échantillonnées des fonctions ondelettes continues (comme il est fait avec les fonctions sinus et cosinus pour le calcul de la TFD et de la TCD par exemple), cette approche pose beaucoup de difficultés pour satisfaire le théorème de Shannon et assurer la reconstruction exacte des signaux avec un coût numérique raisonnable, il faut donc procéder autrement.

En 1987, Y Meyer a démontré qu'il était possible de construire des bases d'ondelettes orthonormées en discrétisant les paramètres de dilatation et de translation a et b respectivement de la manière suivante :

$$a = 2^{-j} \quad , \quad b = k\,a \qquad\qquad j,k \in Z$$

Et plus encore, on choisissant des opérateurs de translation et de dilatation dyadiques (a_0=2, b_0 =1), la réduction de la redondance de la représentation en ondelette est maximale.

On obtient ainsi des bases dans $L^2(R)$ de la forme :

$$(\psi_{j,k})_{j,k\in Z} = \{\, 2^{\frac{-j}{2}} \psi\,(2^j x - k) \;/\; j,k \in Z \,\} \qquad (2.12)$$

et la décomposition d'un signal $f(x) \in L^2(R)$ peut alors s'écrire :

$$f(x) = \sum_{j\in Z}\sum_{k\in Z} \left\langle f, \psi_{jk} \right\rangle \psi_{jk} \qquad (2.13)$$

où

$$\left\langle f, \psi_{jk} \right\rangle = 2^{\frac{j}{2}} \int_{-\infty}^{+\infty} f(u)\, \overline{\psi}\,(2^j u - k)\, du \qquad (2.14)$$

$\langle f, \Psi_{j,k} \rangle$: représente les coefficients d'ondelettes qui sont décorrélés entre eux.

La base la plus simple étant le système de Haar, d'autres bases plus régulières ont été construites par la suite par Meyer, Lemarié, Battle, Daubechies, Coiffman et d'autres chercheurs selon des méthodes diverses.

Les bases construites par I.Daubechies sont à supports compacts ce qui les rend particulièrement intéressantes pour des applications nécessitant un coût numérique faible. Les symmlets sont des bases d'ondelettes presque symétriques dérivées des bases de Daubechies

Dans un article publié en 1987, S. Mallat propose un algorithme de transformation rapide (FWT) effectuant une décomposition hiérarchique analogue aux décompositions pyramidales déjà utilisées en codage des images (Codage en sous bandes, bancs de filtres, pyramides de Burt et Adelson, de Chin et de Meyer …).

Les fondements théoriques de l'algorithme de Mallat sont décrits dans la théorie des analyses multirésolutions.

II.6 Analyse multirésolution (AMR) [17] [23] :

Pour mieux comprendre ce que l'on appelle Analyse MultiRésolution, prenons cet exemple : quelle est la longueur de la côte de Tigzirt ? Nous pouvons prendre un globe, avec une règle on calcul la distance et avec l'échelle de la carte on en déduit la longueur de la côte. Il s'agit d'une vision très grossière de l'allure de la côte. Prenons maintenant une carte de l'Algérie, on répète les mêmes opérations. On aura alors un aperçu beaucoup plus précis de la côte et de sa longueur.

Maintenant on prend une carte de Kabylie, là encore notre vision de la côte sera bien plus fine. Finalement, on peut se rendre sur le terrain et on aura la vision la plus qui soit pour calculer la longueur de la côte. Ainsi, d'échelle en échelle, les détails viennent affiner notre image de la côte, d'une vision grossière on passe à une vision fine et claire.

L'approche multi-échelle en analyse fonctionnelle est apparue au début du siècle, pour faire face aux problèmes non résolus par la transformée de fourrier. (Exemple de la régularité et des propriétés locales d'une fonction).

L'approche multirésolution consiste à projeter le signal sur une série de sous espaces orthogonaux de L^2 (R), (les espaces d'approximations V_J et de détails W_J). Nous verrons que la projection d'un signal sur les espaces de détails fournit sa transformation en ondelettes discrète. Les espaces de projection de signal sont entièrement caractérisés par la donnée de deux filtres (passe haut et passe bas). Ces filtres permettent le calcul rapide des coefficients de la transformée en ondelettes discrète par un algorithme itératif.

II.6.1. Définitions :

Définition 1 :

Une analyse multirésolution de L^2 (R^n) (espace des fonctions carrées sommables) est une suite croissante de sous espaces vectoriels fermés $\{V_j\}_{j\in Z}$ de L^2 (R^n) ayant les propriétés suivantes :

$$i) \quad V_j \subset V_{j+1} \tag{2.15}$$

$$ii) \quad f(x) \in V_j \Leftrightarrow f(2x) \in V_{j+1} \tag{2.16}$$

$$iii) \quad \bigcap_{j\in Z} V_j = \{0\} \quad et \quad \bigcup_{j\in Z} V_j \ est\ dense\ dans\ L^2(R^n) \tag{2.17}$$

iv) Il existe une fonction $g(x)$ *dans* V_0 *telle que la suite* $\{g(x-k) / k \in Z^n\}$ *soit une base inconditionnelle de* V_0 .

Un sous espace V_j peut être interprété comme l'espace des approximations à la résolution 2^{-j}, de cette façon si $f(x)$ est une fonction de $L^2(R^n)$ et $(F_j(x))_{j\in Z}$ la suite d'approximations de $f(x)$ dans $\{V_j\}_{j\in Z}$, alors nous avons les propriétés suivantes :

$$\bigcap_{j\in Z} V_j = 0 \quad \Rightarrow \quad \lim_{j\to-\infty}(F_j(x)) = 0 \qquad (2.18)$$

$$\text{et } \lim_{j\to+\infty} \| F_j(x) - f(x) \| = 0 \qquad (2.19)$$

La suite $(F_j(x))_{j\in Z}$ converge uniformément vers $f(x)$.

Définition 2 :

Une analyse multi résolution $\{V_j\}_{j\in Z}$ de $L^2(R^n)$ est régulière (r ∈ N) si la fonction $g(x)$ vérifie les propriétés :

$$\left|\partial^\alpha g(x)\right| \le C_m \left(1+|x|\right)^{-m} \qquad (2.20)$$

pour tout $\alpha \in N^n$ tel que $|\alpha| < r$ et tout $m \in N$, relation où :

$$\partial^\alpha = \left(\left(\frac{\partial}{\partial x_1}\right)^{\alpha_1}, \ldots, \left(\frac{\partial}{\partial x_n}\right)^{\alpha_n}\right) \quad et \quad |\alpha| = \alpha_1 + \alpha_2 + \ldots\ldots + \alpha_n . \qquad (2.21)$$

Ce qui veut dire que toutes les dérivées (au sens de l'opérateur différentiel ∂^α) de $g(x)$ jusqu'à l'ordre r sont localisées (elles décroissent rapidement vers 0 (en module) lorsque $|x|$ crois vers $+\infty$).

II.6.2. La fonction d'échelle [2] [21] :

Théorème1.

Soit $\{V_j\}_{j\in Z}$ une analyse multi résolution de $L^2(R^n)$. Il existe alors $C_2 > C_1 > 0$ tels que :

$$C_1 \le \left(\sum_{K\in Z^n} \left|\hat{g}\left(\varepsilon + 2K\pi\right)\right|^2\right)^{\frac{1}{2}} \le C_2 \qquad (2.22)$$

pour presque tout $\varepsilon \in R$, $\hat{g}(\omega)$ étant la transformée de Fourier de $g(x)$. On définit alors une fonction ϕ :

$$\hat{\phi}(\omega) = \hat{g}(\omega)\left(\sum_{K\in Z^n} \left|\hat{g}\left(\omega + 2K\pi\right)\right|^2\right)^{\frac{1}{2}} \qquad (2.23)$$

telle que :

$\{\phi(x-k)\}$, $k \in Z^n$ soit une base orthonormée de V_0.

Toute autre fonction $f(x)$ définissant une base orthonormée ($f(x-k)$) , $k \in Z^n$ de V_0 sera reliée à $\phi(x)$ par :

$$\hat{f}(\omega) = \theta(\omega)\hat{\phi}(\omega) \qquad (2.24)$$

Où :

$\theta(\omega) \in L^{\infty}(R^n)$ est une fonction 2π -périodique et $|\phi(\omega)| = 1$ pour presque tout ω.

Théorème 2.

Soit $\{V_j\}_{j \in Z}$ une analyse multi résolution de $L^2(R^n)$. Il existe une fonction $\phi(x) \in L^2(R^n)$ telle que la suite $((\phi_{jk})_{j \in Z, k \in Z^n})$ définie par :

$$\phi_{jk}(x) = 2^{\frac{nj}{2}} \phi(2^j x - k) \qquad (2.25)$$

Constitue une base orthonormée de V_j c'est à dire :

$$\langle \phi_{jk}, \phi_{jl} \rangle = \delta_{k,l} \qquad k, l \in Z^n .$$

La fonction ϕ est appelée fonction d'échelle, fonction d'interpolation ou encore "ondelette père ". On peut définir alors une suite d'opérateurs linéaires de projection orthogonale : $\{E_j\}$, $j \in Z$

tels que

$$\begin{array}{ccc} E_j & : L^2(R^n) \longrightarrow V_j & \\ f(x) & \mapsto \quad F_j(x) = E_j(f(x)) & \end{array} \qquad (2.26)$$

où

$$F_j(x) = \sum_{k \in Z^n} S_{jk} \phi_{jk}(x) \qquad (2.27)$$

et

$$S_{jk} = 2^{nj} \int_{R^n} f(u) \phi(2^j u - k) \, du \qquad (2.28)$$

$F_j(x)$ est l'approximation multi résolution de la fonction $f(x)$ dans l'espace V_j à l'échelle 2^j (à la résolution 2^{-j}) ; $(F_j(x))_{j \in Z}$ est une suite d'approximation de plus en plus fines de $f(x)$ pour j croissant (à mesure que j croît).

Si on note W_j le complément orthogonal de V_j dans V_{j+1} on obtient la suite de sous espaces $(W_j)_{j\in z}$ telle que : $L^2(R^n) = \overset{+\infty}{\underset{-\infty}{\oplus}} W_j$ et $\overset{j}{\underset{-\infty}{\otimes}} W_i = V_{j+1}$

Si $F_j(x)$ et $F_{j+1}(x)$ sont les approximations de $f(x)$ dans V_j et V_{j+1} respectivement alors la fonction :

$$D_j(x) = F_{j+1}(x) - F_j(x) \qquad (2.29)$$

appartient à W_j .

$D_j(x)$ représente le supplément d'informations (détails) à apporter à l'approximation $F_j(x)$ pour obtenir $F_{j+1}(x)$ qui est une représentation plus fines de $f(x)$.

II.6.3. Les fonctions ondelettes :

Théorème 3.

Soit $\{V_j\}_{j\in Z}$ une analyse multirésolutionde $L^2(R^n)$. Il existe $q = (2^n - 1)$ fonctions $\psi_l(x)$ $1 \le l \le q$ appartenant à V_1 telles que:

a) $\left|\partial^\alpha \psi_l(x)\right| \le C_N \left(1 + |x|\right)^{-N}$ (2.30)

Pour tout multi-indice $\alpha \in N^n$ tel que $|\alpha| \le r$, tout $x \in R^n$ et tout N >1, ce qui signifie que les dérivées de $\psi_l(x)$ ont le même ordre de régularité que la fonction d'échelle associée à l'analyse multirésolution $\{V_j\}_{j\in Z}$: $\phi(x)$.

b) les fonctions $\psi_l(x-k)$, $1 \le l \le q$, $k \in Z^n$ (2.31)

forment une base orthonormée de W_0.

c) les fonctions $2^{\frac{nj}{2}} \psi_l(2^j x - k)$ $1 \le l \le q$, $k \in Z^n$, $j \in Z$ (2.32)

forment une base orthonormée de $L^2(R^n)$, cette base est appelée base orthonormée d'ondelettes.

La fonction $D_j(x)$, précédemment introduite peut alors être écrite :

$$D_j(x) = \sum_{l=1}^{q} \left(\sum_{k=-\infty}^{+\infty} B_{jk}^{l} \ \psi_l(2^j x - k) \right) \qquad (2.33)$$

où

$$B_{jk}^{l} = 2^{nj} \int_u f(u)\, \psi_l(2^j u - k) \quad du \qquad (2.34)$$

L'ensemble des définitions et théorèmes suscités, sont valables pour toute dimension n de l'espace de définition (R^n). Il est cependant, plus commode de raisonner sur une dimension, la généralisation (notamment en 2-D, en vue d'une application sur des images) se fait le plus souvent, en utilisant des produits tensoriels.

II.7. Propriétés des filtres de décomposition – reconstruction :

L'analyse s'effectue selon la résolution choisie. Les fonctions de transferts des filtres utilisés doivent avoir une distorsion d'amplitude la plus réduite possible. Cela permet de ne pas défavoriser d'une part, certaines fréquences par rapport à d'autres et d'autre part, les filtres à phase linéaire de ceux dont la phase ne l'est pas. Cette dernière caractéristique est d'autant plus importante que le signal que nous voulons analyser est un signal bidimensionnel représentant une image et que la perception visuelle chez l'homme est sensible aux fluctuations irrégulières de la phase.

De ce fait, les Filtres Miroirs en Quadrature (QMF) à réponse impulsionnelle infinie (RII) ont été retenus dans un premier temps pour avoir une reconstruction exacte, mais l'inconvénient résidait dans un temps de calcul important. La limitation des filtres RII pour obtenir des filtres RIF (réponse impulsionnelle finie) permettait de réduire les temps mais provoquait à leur tour des dégradations pendant la reconstruction. C'est pour résoudre ces problèmes que les Filtres Conjugués en Quadrature (QCF) ont été développés. En effet, ces filtres présentent l'avantage d'être à reconstruction exacte.

Daubechies a réussi à calculer des filtres *h* (*k*) (appelés filtres de Daubechies) en posant les relations entre les filtres *G0*, *H0*, *G1* et *H1* sont donnée par :

$$H(w) = \tfrac{1}{2}\left(1 + e^{jw}\right)^N Q\left(e^{jw}\right) \qquad (2.35)$$

Avec

$$Q\left(e^{jw}\right) = \textstyle\sum_{\mathrm{n}=0}^{\mathrm{N}-1} \mathrm{q(n)}\, e^{jnw} \quad , q(0) = 0 \qquad (2.36)$$

Les relations entre les filtres *G0*, *H0*, *G1* et *H1* sont donnée par :

$$\begin{cases} H_1(z) = -G_0(-z) \\ G_1(z) = -H_0(-z) \\ G_{10}(z) = H_0(-z^{-1})Z^{-(N-1)} \end{cases} \quad (2.37)$$

$$\begin{cases} h_1(k) = (-1)^{k+1} g_0(k) \\ g_1(k) = (-1)^k h_0(k) \\ g_0(k) = (-1)^{k+1} h_0(N-1-k) \end{cases} \quad (2.38)$$

II.8. Les ondelettes biorthogonales [17] :

Les bases d'ondelettes biorthogonales sont une généralisation des bases orthogonales qui permet, en s'affranchissant de la contrainte d'orthogonalité, d'obtenir des bases d'ondelettes symétriques, de forte régularité et à support fini dont les filtres associés sont à phase linéaire et à reconstruction exacte. Les ondelettes biorthogonales ont notamment été utilisées pour le débruitage des signaux et des images, la reconnaissance d'empreintes digitales et la résolution de certains problèmes mathématiques (calcul matriciel et différentiel).

Pour les applications en traitement d'images, il est important que les filtres associés aux fonctions échelle et ondelette soient à reconstruction exacte et à phase linéaire ou nulle pour préserver les contours de l'image et éviter les effets de bords. Ces filtres doivent être à supports courts pour diminuer le nombre d'opérations à effectuer lors des calculs et permettre ainsi, une exécution rapide des traitements. De plus, un ordre élevé de régularité est souhaitable, en particulier en compression d'images.

Cependant, ces critères ne peuvent être satisfaits simultanément dans le cadre des bases orthogonales. En effet, parmi les filtres RIF orthogonaux, symétriques et à reconstruction exacte, seul le filtre de Haar (dont l'ordre de régularité est faible) peut satisfaire ces critères.

II.9 Analyse multirésolution bi-orthogonale :

Une analyse multirésolutionbiorthogonale de $L2$ (R) est définie par deux suites de sous espaces$\{V_j\}_{j\in Z}$ et $\{\tilde{V}_j\}_{j\in Z}$ ayant les propriétés suivantes :

$$* \bigcap_{j\in Z} V_j = \{0\} \quad \text{et} \quad \bigcap_{j\in Z} \tilde{V}_j = \{0\}$$

$$*\bigcup_{j\in Z} V_j = L^{(2)}(R^n) \quad \text{et} \quad \bigcup_{j\in Z} \tilde{V}_j = L^{(2)}(R^n) \tag{2.39}$$

$$*V_j \subset V_{j+1} \quad \text{et} \quad \tilde{V}_j \subset \tilde{V}_{j+1}$$

et de deux fonctions $\emptyset(x)$ et $\tilde{\emptyset}(x)$ de norme 1 dans $L^{(2)}(R)$ engendrant des bases non orthogonales de V_j et $\tilde{V}_j$ respectivement, tel que la base soit biorthogonale : $(\emptyset_{jk}, \tilde{\emptyset_{jk}})$ On peut définir alors deux suites de carrés sommables h_k et $\tilde{h_k}$ telles que :

(2.40)

$$\phi(x) = \sum_{k=-\infty}^{+\infty} h(k)\, \phi(2x-k) \tag{2.41}$$

$$\tilde{\phi}(x) = \sum_{k=-\infty}^{+\infty} \tilde{h}(k)\, \tilde{\phi}(2x-k)$$

On introduit deux fonctions ondelettes Ψ et $\tilde{\Psi}$ définies par :

$$\psi(x) = \sum_k g_k \phi(2x-k) \quad , g_k = (-1)^{k-1} \tilde{h}(1-k) \tag{2.42}$$

$$\tilde{\psi}(x) = \sum_k \tilde{g}_k \tilde{\phi}(2x-k) \quad , \tilde{g}_k = (-1)^{k-1} h(1-k) \tag{2.43}$$

Le système $(\Psi_{jk}, \tilde{\Psi}_{jk})$ est appelé base biorthogonale de $L^2(R)$.

Les algorithmes de décomposition et de reconstruction utilisant les ondelettes biorthogonales ne diffèrent de ceux définis dans le cadre des ondelettes orthogonales que par l'introduction des filtres $\tilde{h}$ et $\tilde{g}$ utilisés lors de la reconstruction au lieu de h et g respectivement.

Les relations 2.44 et 2.45 sont utilisées pour la décomposition et la relation 2.46 pour la reconstitution.

$$S_j(k) = \sum_{l=-\infty}^{+\infty} h(l)\, S_{j+1}(l+2k) \tag{2.44}$$

$$D_j(k) = \sum_{l=-\infty}^{+\infty} g(l)\, S_{j+1}(l+2k) \tag{2.45}$$

$$S_{j+1}(k) = 2\sum_{-\infty}^{+\infty} \tilde{h}(k-2n)\, S_j(n) + 2\sum_{-\infty}^{+\infty} \tilde{g}(k-2n)\, D_j(n) \tag{2.46}$$

Les ondelettes biorthogonales permettent d'avoir des ordres de régularité plus grands que ceux des ondelettes orthogonales pour des longueurs de filtres équivalentes. Elles permettent

aussid'avoir un grand nombre de moments nuls pour un support réduit. Ces qualités sont très appréciées dans diverses applications et notamment en compression d'images.

Les ondelettes biorthogonales comptes plusieurs familles (pseudo-Coiflets, filtres récursifs…)mais la plus usuelle est la famille des ondelettes Splines. Par ailleurs, les filtres définis par le standard JPEG2000 appartiennent à cette famille (9/7 pardéfaut pour la compression avec perte, 5/3 par défaut pour la compression sans perte, et 10/2 pour la compression sans perte).

Les recherches sont actuellement orientées vers la recherche d'algorithmes de calcul de filtres etd'analyse/synthèse optimaux.

II.10 Algorithme pyramidal de S.Mallat [17] [21] :

L'algorithme pyramidal adapté par S. Mallat, utilise la multiplication dyadique séparable. Cette dernière a pour avantage de permettre un calcul rapide, elle permet aussi d'interpréter les images de détails comme les hautes fréquences sélectives en direction horizontale, verticale et diagonale.

- **Cas d'un signal monodimensionnel :**

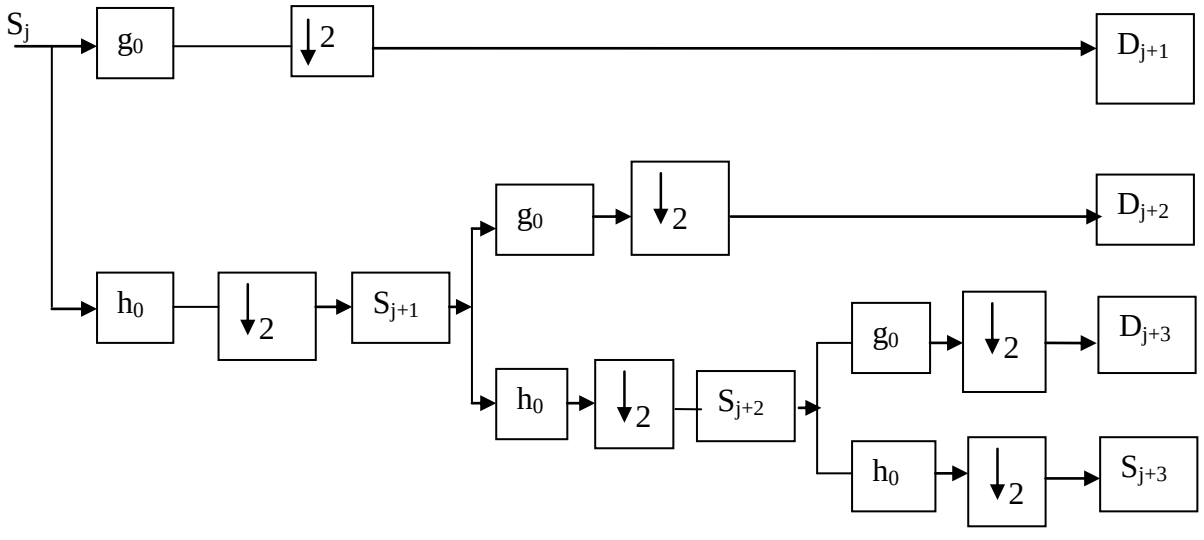


Figure II.4. Algorithme de décomposition d'un signal

Le principe de ce schéma consiste à décomposer le signal S_j en deux sous signaux Dj_{+1} et S_{j+1}. Cette opération de décomposition peut se répéter sur le signal S_{j+1} et ainsi de suite, le nombre d'approximations est limité par l'ordre du filtre.

L'ensemble de détail et du signal lissé représenté à la dernière résolution, fournit une reconstruction exacte du signal S_j sans perte d'informations. D_{j+1} ($D_{J+1} = S_j - S_{j+1}$),représente la fluctuation en passant d'une résolution à une résolution plus fine.

S_{j+1} représente le signal lissé (L'approximation) obtenu à partir de S_j après l'opération du filtrage en utilisant un filtre passe bas (h_0) suivi de l'opération de décimation.

D_{j+1} représente le détail perdu en passant d'une approximation à une autre, il est appelé aussi coefficient d'ondelettes. Pour aboutir à ce dernier, on applique un filtre passe haut (g_0) sur S_j suivi d'une décimation.

S_j est reconstitué en multipliant les valeurs des échantillons par deux après avoir effectué une sommation des deux signaux obtenus, l'un par filtrage passe bas (h_1) et l'autre par filtrage passe haut (g_1) des signaux S_{J+1} et D_{j+1} respectivement, avec interpolation (insérer un échantillon nul entre deux), comme le montre la figure ci–dessus.

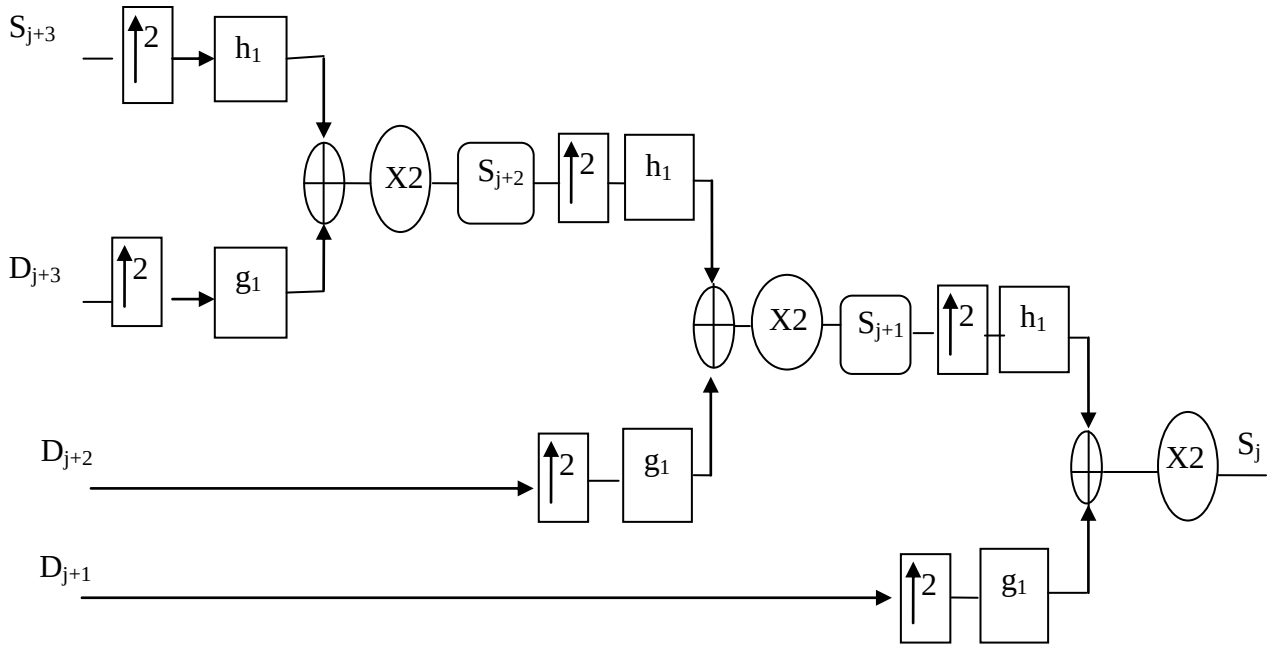
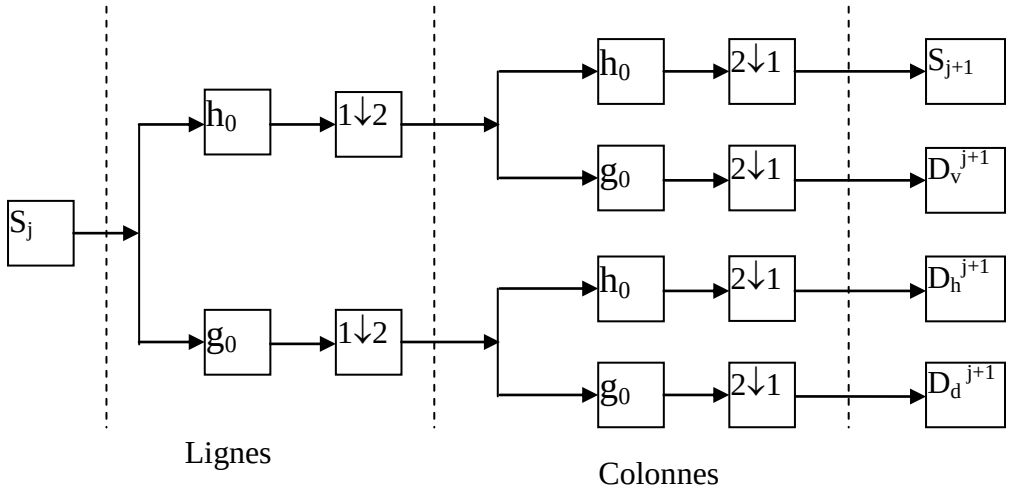


Figure II.5. Algorithme de reconstruction d'un signal monodimensionnel

- **Cas d'un signal bidimensionnel:**

Dans le cas d'un signal bidimensionnel, spécialement dans le cas des images, on utilise les schémas de décomposition et de reconstruction suivants :



X	Convolution (ligne ou colonne) avec le filtre X
1↓2	Décimation d'une ligne sur deux
2↓1	Décimation d'une colonne sur deux

Figure .II.6 Etape de décomposition

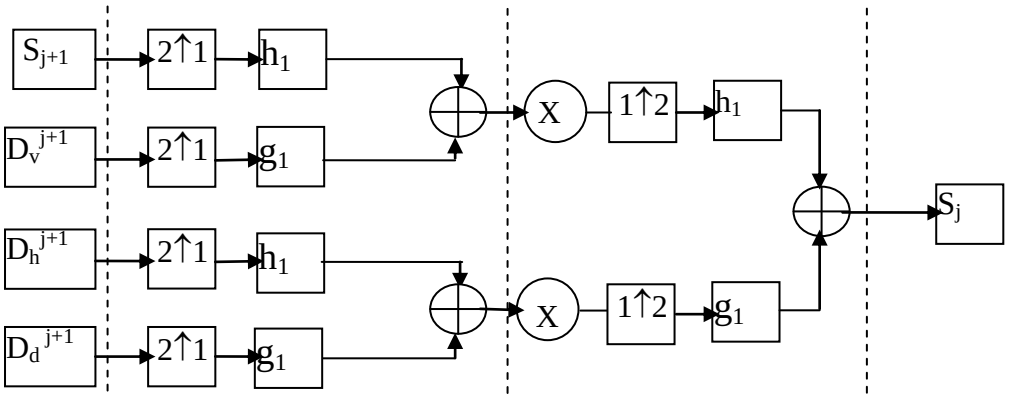


Figure II.7. Etape de reconstruction

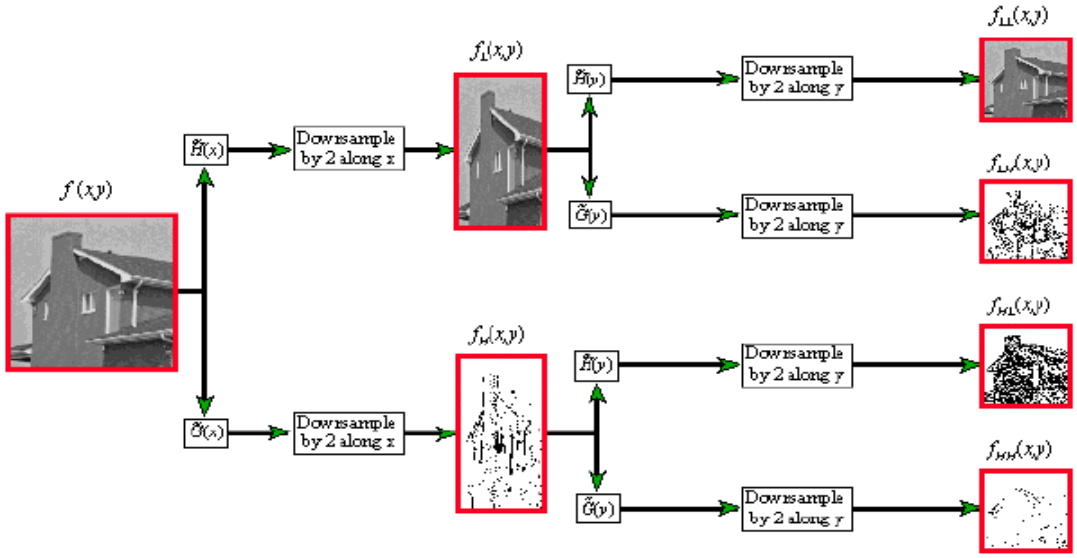


Figure II.8 Exemple de la décomposition en ondelette de l'image maison à un seul niveau

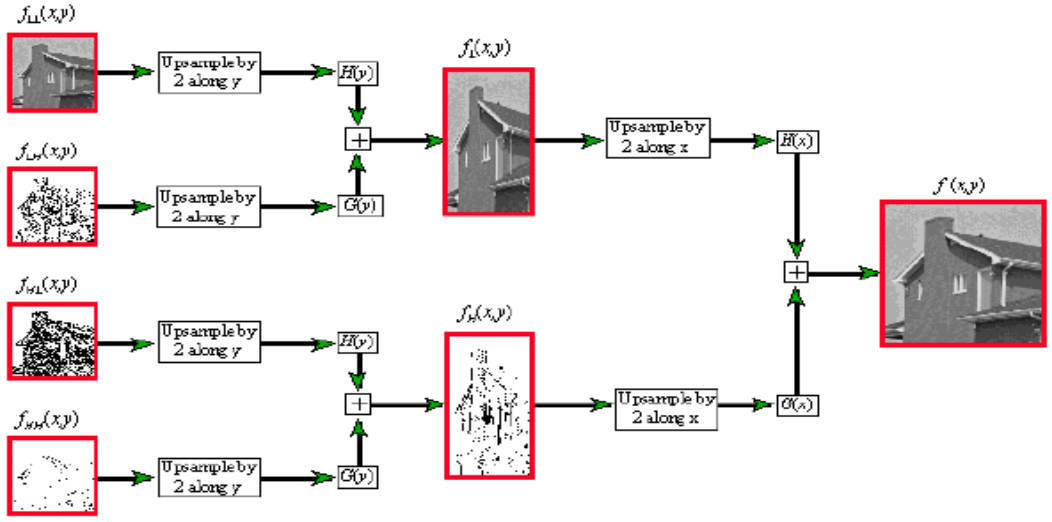


Figure II.9 Exemple de la reconstruction en ondelette de l'image maison à un seul niveau

Conclusion :

Dans ce chapitre nous avons abordé les étapes nécessaires à la mise en œuvre de la théorie des ondelettes. La transformée en ondelettes est un outil très puissant pour l'analyse des signaux numériques. Les ondelettes constituent le cœur de JPEG 2000, elle présente tous les avantages d'une représentation temps-fréquence ou temps échelle et possède en plus un algorithme rapide. Des études ont montré que la décomposition effectuée par la transformée en ondelettes est dans une certaine mesure analogue aux mécanismes de perception auditive et visuelle des humains.

Nous avons vu que l'analyse multirésolution est l'outil déterminant pour exploiter tous les atouts de cette théorie, celle-ci combinée aux bancs de filtres. En effet elle permet de séparer les détails d'un signal à plusieurs résolutions données, ces résolutions variant avec un pas dyadique.

L'objectif d'une compression basée sur les ondelettes est bien de minimiser le nombre de coefficients de détails à traiter avant de procéder à la reconstruction de signal.

INTRODUCTION

La transformée en ondelettes permet, comme décrit dans le chapitre précédent, de représenter les images sous forme de coefficients ordonnés en bandes de fréquences. Pour la compression d'images, la transformée représente le premier maillon de la chaîne, afin de compresser l'information, il faut compléter le cycle par la quantification et le codage.

La méthode de codage progressif connue sous le nom en anglais " Embedded Zerotree Wavelet : EZW ", proposée par Shapiro est une méthode simple et très efficace de codage d'images basé sur la transformée en ondelettes. Elle a démontré sa puissance dans les deux formes de compression (avec et sans perte d'informations) depuis son élaboration en 1993. Plusieurs variantes de ce type de codage ont été proposées par différents chercheurs dans le domaine ; On peut citer le SPIHT (Set Partitionning in HierarehicalTree) réalisé par A. SAID et W.PEARLMAN et EBCOT (Embedded BlockCodingwith Optimal Truncation) utilisé dans JPEG2000 ; sont des variantes les plus populaires utiliser dans le codage progressif.

III.1 Algorithme de codage EZW [13] [17] [24] :

Quand on applique la transformée en ondelettes sur une image, on remarque deux choses : la première, est que les formes obtenues pour toutes les sous-bandes se ressemblent. Ceci implique qu'il ya certainement des redondances entre ces derniers et qui pourront être utilisés pour améliorer le taux de compression. La seconde, est que l'image de la sous-bande d'approximation présente la même image que l'original, mais avec une résolution plus petite. Ceci implique que cette zone est la zone privilégiée.

L'algorithme EZW est le mieux indiqué, vu qu'il exploite au mieux ces caractéristiques. Par l'intermédiaire d'une structure de donnée particulière, qui réorganise les coefficients sous forme d'une arborescence inter-bande, qui est nommée "quadtree".

EZW (Embedded Zerotree Wavelet) est le premier algorithme inter-bande développé pour les images, il a été proposé en 1993 par Jérôme SHAPIRO, c'est une méthode qui est utilisée pour la compression des images par ondelettes, il est de type imbriqué (Embedded), vu qu'on peut couper le flux binaire généré par le codeur à n'importe quel point de l'encodage pour obtenir une qualité d'image désirée.

Pour accomplir la tâche de compression, il est préférable d'utiliser un codeur arithmétique, qui puisse coder toute une chaine avec un seul et unique code ; en s'inspirant des probabilités d'apparition de chaque mot que contient cette chaine.

III.1.1. Présentation du codeur EZW :

L'intérêt principal d'utiliser EZW, est qu'il exploite au mieux la caractéristique de multi-résolution; vu qu'il code les coefficients d'ondelettes sous forme binaire par ordre décroissant, en commençant par les basses fréquences (approximation) vers les hautes fréquences(les détails) afin de transmettre l'image d'une manière progressive. Ce qui implique le choix à tout moment d'arrêter ou de continuer le codage en fonction de la qualité d'image reconstruite obtenue.

Pour commencer, il faut d'abord avoir la position des coefficients importants, sachant que l'importance d'un coefficient est évaluée par rapport à un seuil désigné.

Pour ce faire le codeur introduit la notion de redondance inter-bande, qui suppose que lorsqu'on diminue l'échelle (on passe à la haute résolution) les coefficients d'ondelettes diminues aussi. Autrement dit, si une zone d'image est peut significative, ceci implique que cette même zone sera insignifiante à la haute résolution.

Ceci prouve que le codage progressif est un choix très normal pour des images transformées par ondelettes, puisque les sous-bandes plus hautes ajoutent seulement les détails fins.

Le codeur introduit également une hiérarchie, qui permet de relier chaque coefficient dans une sous-bande avec d'autres coefficients dans la sous-bande qui suit et les réorganiser sous forme d'arborescence. C'est ainsi qu'on parle de l'arbre de zéros (zerotree).

- **Définition de zerotree :**

Un zerotree est une structure de donnée quadtree, comme son nom l'indique, c'est un arbre dont les nœuds ont quatre branches ; autrement dit, on associe pour chaque fils, quatre descendants (quadtree) dans la sous-bande qui suit et ainsi de suite jusqu'à ce que l'on atteint les sous-bandes de haute fréquences.

Cette équivalence mathématique n'est pas sortie de nulle part, mais elle est issue de l'application d'ondelettes sur les images et donc la création des sous-bandes. Car, quand on applique la transformée en ondelettes sur une image jusqu'à la $(K+1)^{éme}$ résolutions, on remarque qu'une sous-bande LH_k (respectivement HH_k ou HL_k) a quatre fois plus de point qu'une sous-bande LH_{k+1}(respectivement HH_{k+1} ou HL_{k+1}). Sauf pour la sous bande LL_{k+1} qui est réduite à un seul et unique point, qui présente la racine de l'arbre. Ceci est illustré dans la figure suivante :

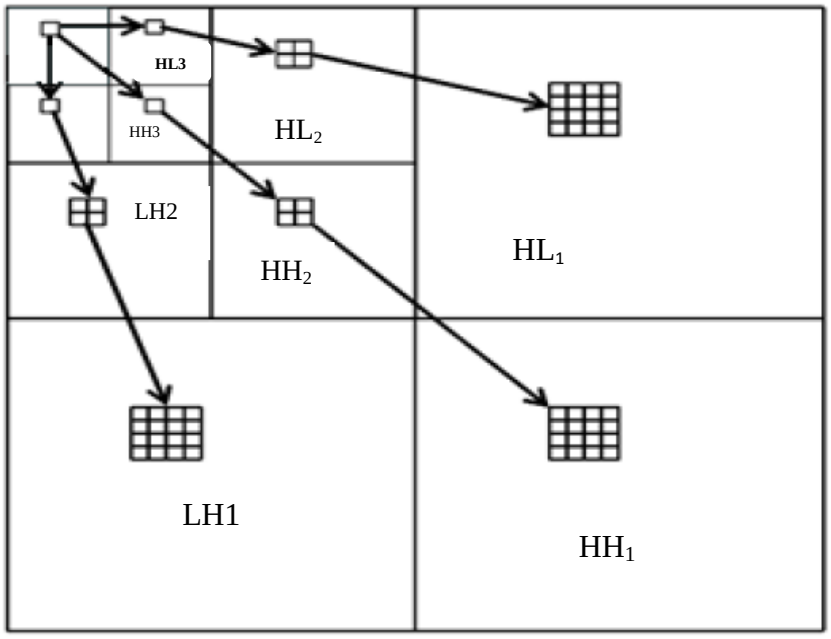


Figure III-1 : Représentation de l'organisation en arbre des coefficients d'ondelettes

- **La combinaison du zerotree avec la redondance inter-bande *:***

Pour exploiter la redondance inter-bande, EZW code les coefficients, nœud après nœud dans plusieurs passages. Pour chaque passage, il choisit un seuil, auquel tous les coefficients seront comparés. Si un coefficient est inférieur au seuil, il vérifie si l'un de ses descendants est supérieur au seuil, pour qu'il le laisse au prochain passage. Sinon, si tous ses descendants sont faibles (c'est souvent le cas d'après la propriété évoquée précédemment), on les code pas. Et si le coefficient est grand il sera codé. De cette manière on ne code que la partie importante, ce qui permet d'avoir une bonne qualité d'image reconstruite avec des taux de compression élevé.

III.I.2. Déroulement de l'algorithme EZW :

Après avoir présenté l'aspect général d'EZW, nous allons passer à détailler cet algorithme. En effet, il effectue récursivement deux passes successives : dominante et subalterne.

A. La passe dominante :

Son nom indique son rôle et son importance. Car c'est là que l'algorithme effectue les tâches importantes, qui sont organisées de la manière suivante :

❖ **Le choix des coefficients significatifs :**

Pour ce faire, il faut d'abord choisir un seuil T_0, puis designer les coefficients significatifs et on compare chaque coefficient à ce seuil. Si la valeur absolue du premier est supérieur au second, cela implique que le coefficient est significatif.

Autrement dit : Si 'A' est l'amplitude du coefficient en cours et que $|A|$ est la valeur absolue de ' A', on note que : $\begin{cases} \text{si } |A| > T0 : \text{ A est significatif} \\ \text{sinon } \quad \text{A n'est pas significatif} \end{cases}$

La valeur du seuil de départ T_0 est définie par la relation suivante :

$$T0 = 2^{\lfloor \log_2 \max(im(x,y)) \rfloor} \qquad \text{(III.1)}$$

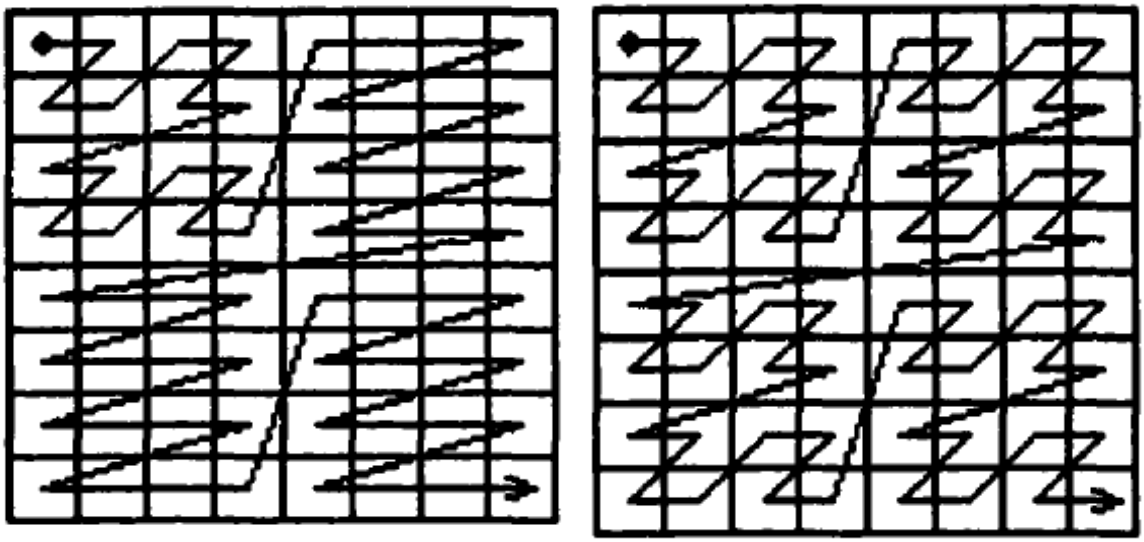
Tel que max (im(x, y)) : désigne le plus grand coefficient d'ondelette existant dans l'image à la position x et y.

❖ **Le parcourt des coefficients :**

Pour accomplir cette tâche, deux méthodes ont été choisies : 'Raster scan' et 'Morton scan' ; Elles parcourent les coefficients, du plus fort (faible fréquence) vers le plus petit (haute fréquence).La différence qui existe entre ces deux méthodes, est dans le chemin que trace chacune lors du parcours de la même sous-bande (sauf pour la dernière sous-bande).

Ceci est illustré par la figure (2), qui présente une matrice de coefficients d'ondelettes à la troisième résolution.

On voit bien que le chemin tracer dans la sous-bande d'approximation, est le même pour les deux méthodes. Par contre, il est différent dans les autres sous-bandes.



Raster scan Morton scan

Figure III.2 Méthodes de parcours des coefficients

❖ **Le codage des coefficients :**

Après le parcours des coefficients et la comparaison de chacun d'eux au seuil T_0, on dispose de quatre cas qui sont en relation directe avec la signification du coefficient (respectivement ses descendants) ou non.

Car si on trouve un coefficient 'A' supérieure au seuil, il est donc significatif ; mais en fonction de son signe, on a deux cas :

*le coefficient 'A' est significatif avec un signe positif : Si $A > T_0$.

*le coefficient 'A' est significatif avec un signe négatif : Si $A < T_0$.

Et si le coefficient 'A' est inférieur au seuil donc il est insignifiant, mais on devrait vérifier son arborescence. Car si l'un de ses descendants est significatif ou non, le code va changer. Ce qui donne deux autres cas à traiter :

*le coefficient 'A' est insignifiant, mais il existe un coefficient significatif parmi ses fils.

*le coefficient 'A' est insignifiant, et il n'existe aucun coefficient significatif parmi ses fils.

C'est pour cette raison que l'algorithme a introduit un dictionnaire de codage qui contient quatre symboles, qui reflète et répond aux cas traités au-dessus, ainsi que leurs codes correspondants.

Le tableau (1) représente les symboles du codage ainsi que leurs indications :

Symboles	Indications
Positif (P)	indique que le coefficient est significatif avec un signe positif.
Négatif(N)	indique que le coefficient est significatif avec un signe négatif.
Zéro isolé(Z)	indique que le coefficient est insignifiant et il existe un coefficient significatif parmi ses descendants (fils).
Zerotree (R)	indique que le coefficient est insignifiant mais il n'existe aucun coefficient significatif parmi ses fils.

Tableau III.1 : les symboles du codage EZW ainsi que leurs indications.

❖ **La quantification par approximation successive :**

Pour faire apparaitre et avoir accès aux coefficients qui sont codés insignifiants avec le seuil initial T_0, on réduit ce dernier à moitié et on aura un nouveau seuil T_n exprimé comme suit :

$$T_n=T_0/2 \qquad \text{(III.2)}$$

Où : n est le nombre de résolution.

Puis on refait les étapes précédentes avec ce nouveau seuil, chose qui va nous permettre de récolter d'autres informations, qui seront ajoutées à celle d'auparavant d'une manière successive. Tant que $T_n>1$, cette opération peut être refaite tant de fois.

Cette division du seuil sur 2 est derrière la transmission successive de l'image. Car on a comme première vue l'approximation (avec T_0) puis étape par étape on ajoute les nouvelles informations issues des T_n, qui présentent réellement les détails.

B. La passe subalterne :

Les coefficients traités par cette passe, sont ceux jugés significatifs dans la passe dominante, afin d'avoir plus de précision sur eux.

Pour ce faire, ces coefficients subissent une autre comparaison avec un autre seuil, ce dernier est en fonction de celui utilisé dans la passe dominante. La relation mathématique qui définit ce seuil est donnée par l'équation (III.3).

$$T_s=3*2^{n-1} \qquad \text{(III.3)}$$

Où : n est la puissance du seuil de la passe dominante. ($T_0=2^n$).

Puis, on attribue le bit '1' au coefficient supérieur à T_s et le bit '0'au coefficient inferieur à T_s.

Le rôle essentiel de cette passe, est qu'elle nous assure une meilleure reconstruction de l'image que celle issue de la passe dominante. C'est pour cette raison que le processus de codage alterne entre les deux passes et réduit le seuil de moitié après la passe dominante. Et la chose importante à retenir, est que le codage peut s'arrêter à n'importe quel moment en fonction du taux de compression atteint. Les étapes du codage peuvent être résumées par l'organigramme suivant :

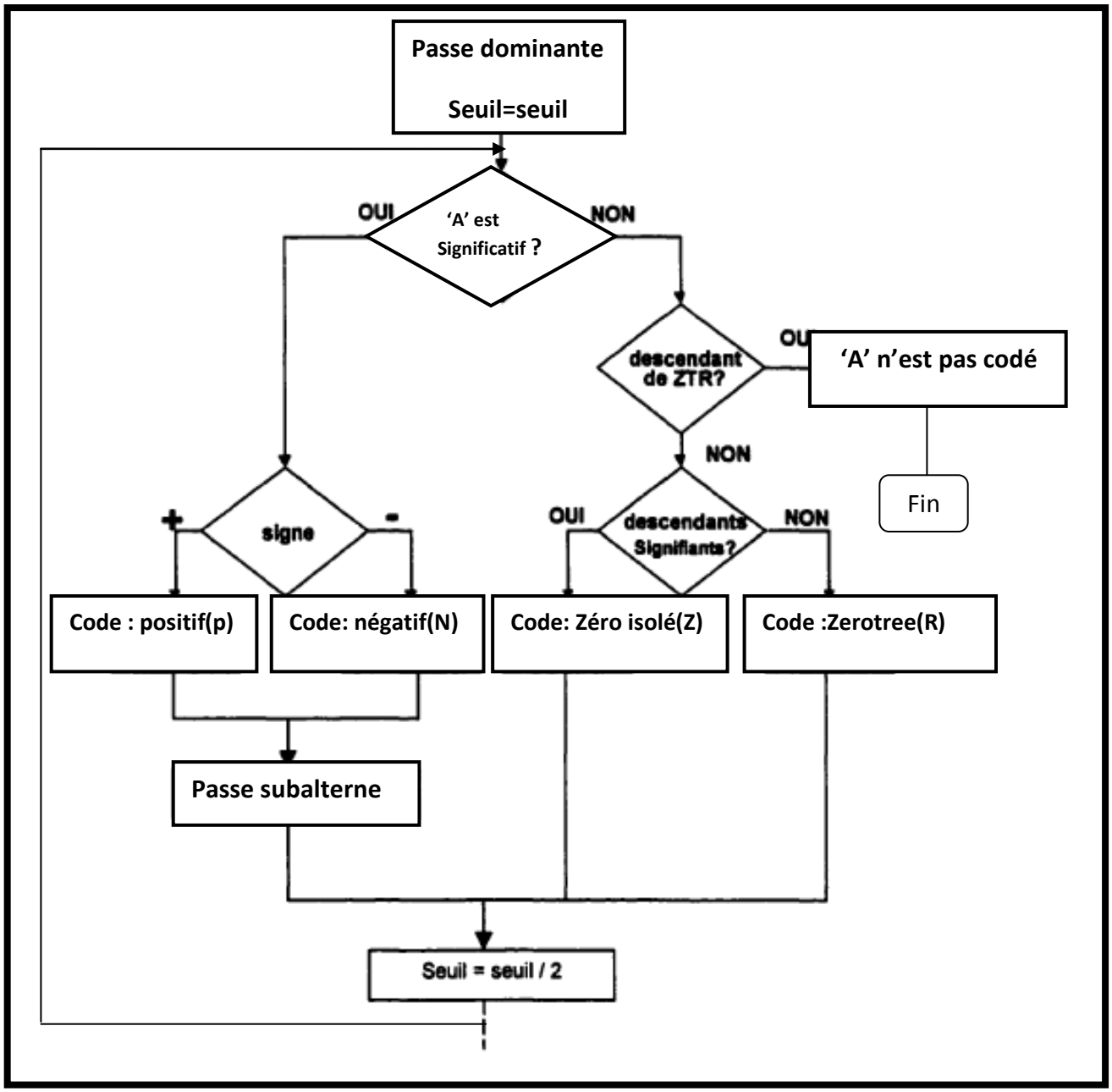


Figure III.3Organigramme représentant les étapes codage de EZW

III.1.3. Phase de décodage :

Pour qu'on puisse réaliser le décodage, il faut d'abord avoir le nombre de niveaux de décomposition par ondelettes, qui permettent aux décodeur de construire une matrice, dont la taille est la même que celle de l'image original.

Puis le codeur envoie la puissance associée au seuil de départ, pour que le décodeur puise calculer la valeur de ce seuil, ainsi les coefficients vont être reconstruit par rapport à ce dernier ; Comme illustre le tableau (2).

A la fin du parcours, le seuil est divisé par 2 et 1' algorithme reprend. Lorsqu'on veut mener une compression sans perte, et donc avoir une reconstruction idéale de l'image, il faut que le seuil atteigne la valeur de '1'.Mais dans le cas d'une compression avec perte, le seuil n'est pas forcer d'atteindre cette valeur, mais par contre, les coefficients vont être traités une deuxième fois, pour assurer plus de précision et éviter moins de pertes.

Les symboles	Le décodage
P	On ajoute la valeur du seuil au contenu de la case en cours.
N	On soustrait de la valeur du coefficient celle du seuil.
R	On ignore complètement les descendants de ce coefficient.
Z	On ne peut pas ignorer l'arborescence de ce coefficient, vu qu'il existe au moins un coefficient significatif.

Tableau III.2 : le protocole de décodage.

III.1.4. Exemple d'Application de l'algorithme EZW :

L'exemple suivant illustre le codage appliqué sur une matrice de 8*8 coefficients à trois niveaux de décomposition.

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	2
−5	9	−1	47	4	6	−2	2
3	0	−3	2	3	−2	0	4
2	−3	6	−4	3	6	3	6
5	11	5	6	0	3	−4	4

Figure III.4 : Exemple de matrice de coefficients

➢ **La première passe dominante :**

✓ On calculele seuil T0, en appliquant l'équation(III-1).

Le coefficient dont la valeur (63) est la valeur maximal de la matrice, ce qui donne T_0=32.

✓ Puis on parcourt les coefficients avec Morton Scan, comme c'est illustré par la figure (4) :

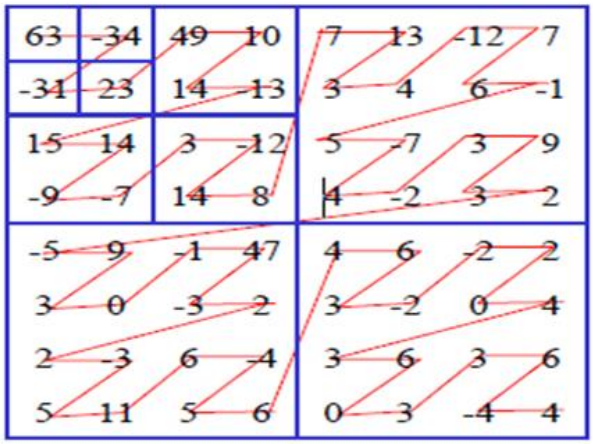


Figure III.5: Parcourt des coefficients de la matrice avec Morton Scan

✓ On compare chaque coefficient avec la valeur (32) et on leur applique les symboles décrits auparavant, ce qui donne le code suivant :
D1=PNZRPRRRRZRRRRRRRPRR

➢ **La première passe subalterne :**

✓ On traite les coefficients qui sont codés 'P' ou 'N', afin d'avoir plus de précision. Pour ce faire, on les compare au nouveau seuil T_s, qui est déterminé par l'équation (III.3).
Sachant que : $32=2^5$ => $T_{s1}=3*2^4=48$.
On a seulement quatre coefficients à comparer (3P et 1N), ce qui donne le code suivant :

S1= 1010

Pour plus de compréhension, on va prendre quelques coefficients pour expliquer en détail la manière dont on leurs a attribué ces codes :

- le coefficient '63' est supérieur à T_0et comme il est positif, donc on le code 'P'.

À la deuxième passe, 63>T_{s1}, donc il est codé par le bit '1'.

- le coefficient négatif '-34' dont la valeur absolue est supérieur à T_0 et comme il est négatif, donc on le code 'N'.

À la deuxième passe, '-34' <T_{s1}, donc il est codé par le bit '0'.

- le coefficient négatif '-31' dont la valeur absolue est inférieure à T_0 et comme il possède un descendant dont la valeur est supérieure au seuil (47), il est codé 'Z'.

La deuxième passe n'est pas recommandée, vu qu'il n'est pas 'N' ou 'P'.

- le coefficient '23' ainsi que tous ses descendants sont insignifiants à T_0, donc on lui attribue le code 'R' et tous ses descendants ne seront pas traités pendant cette passe.

La deuxième passe n'est pas nécessaire.

- ✓ Puis on calcule un nouveau seuil $T_{n=1,2,...}$ en appliquant la relation (III.2).
- ✓ Puis on refait les étapes précédentes jusqu'à $T_n=1$.

➢ **Les résultats obtenus pour les autres passes sont :**

T_2 =16, T_{s2}=24

D2: ZRNPRRRRRRRR

S2: 100110

T_3=8, T_{s3}=12

D3 : ZZZZZPPNPPNRRNNPRPRRNRRRRRRRPRRRPRRRRRRRRRPRRRRRRRRRRR

S3: 10011101111011011000

T_4=4, T_{s4}=6

D4: ZZZZZZZRZRZNZZZZPRRPRPPRPNPRNRRRRRPRPNPPPPRRRRRPRPRRRPNP

S4: 1101111101100100000111011010000100101011100

T_5= 2, T_{s5}=3

D5:ZZZZZRZZZZZRPZZZRRPRRRRNPRPPRRPRRRNPPNRRRRPNNPRRPRRPPRRR

S5: 10111100110100010111110101101100100000000110110110011000111

T_6=1

D6: ZZZRRZRRRZRRRRRNNRRR

On remarque que pour T_6=1, on a seulement effectué la passe dominante. Ceci est expliqué par le code D6, contient que des 'Z' et ' R', donc ces coefficients n'ont pas l'accès à la passe subalterne. Autrement dit, il y a aucune information intéressante à extraire.

A ce stade de codage, on est confronté à la version originale de l'image (approximation et détails), donc la reconstruction est parfaite et il n'y a aucun détail à rajouter.

III.2. Algorithme de codage SPIHT [14] [23] [24] :

SPIHT (Set Partionning In Hierarchical Trees) est un algorithme proposé par A. Said et W.A. Pearlman basé sur la transformation en ondelettes discrètes (DWT).La différence essentielle entre EZW et SPIHT est la façon dont les coefficients des arbres sont construits, triés et découpés. Ainsi la structure même des arbres de zéros est différente. Dans l'EZW, un arbre de zéros est défini par un coefficient racine et ses descendants ont tous la valeur zéro à l'intérieur d'un plan de bits.

SPIHT utilise lui deux types d'arbres de zéros :

- le premier (type A) consiste en une simple racine ayant tous ses descendants à '0' pour un plan de bits donné. Cela diffère un peu des arbres de zéros de l'EZW du fait que la racine elle-même n'a pas besoin d'être non significative. En fait, bien que l'arbre de zéros soit spécifié par les coordonnées de la racine, cette dernière n'est pas incluse dans l'arbre.
- Le second type d'arbre (type B) est similaire au type A, mais exclut les quatre enfants de la racine. Les arbres de type B contiennent uniquement les petits-enfants, arrières petits-enfants,...de la racine.

Une autre différence est l'ordre de traitement des coefficients qui est dépendante des données. Alors que les coefficients sont traités en zigzag dans chaque sous-bande pour l'EZW. Le système de liste du SPIHT laisse l'ordre entièrement dépendant des données. Les coefficients sont traités selon leur position dans les listes. La définition des arbres de zéros comme on vient de le voir est sensiblement différente car SPIHT considère deux types d'arbres de zéros : le type A (arbre de degré 1) et le type B (arbre de degré2). Il est à noter que dans les deux cas, rien n'est dit sur la valeur du coefficient à la racine qui peut être significatif. Pour caractériser les relations parent/enfant dans les sous-bandes. Les ensembles de coordonnées utilisés sont les suivants, et illustrés à la figure(5) :

- O(i, j) : Ensemble des coordonnées de tous les enfants du nœud (i, j).Il s'exprime de la même façon que celui de l'EZW.
- D(i, j) : Ensemble des cordonnées de tous les descendants du nœud (i, j) (type A d'arbres de zéros).
- L(i, j)=D(i, j) – O(i, j) : Ensemble des descendants à l'exception des enfants(Type B d'arbre de zéros).

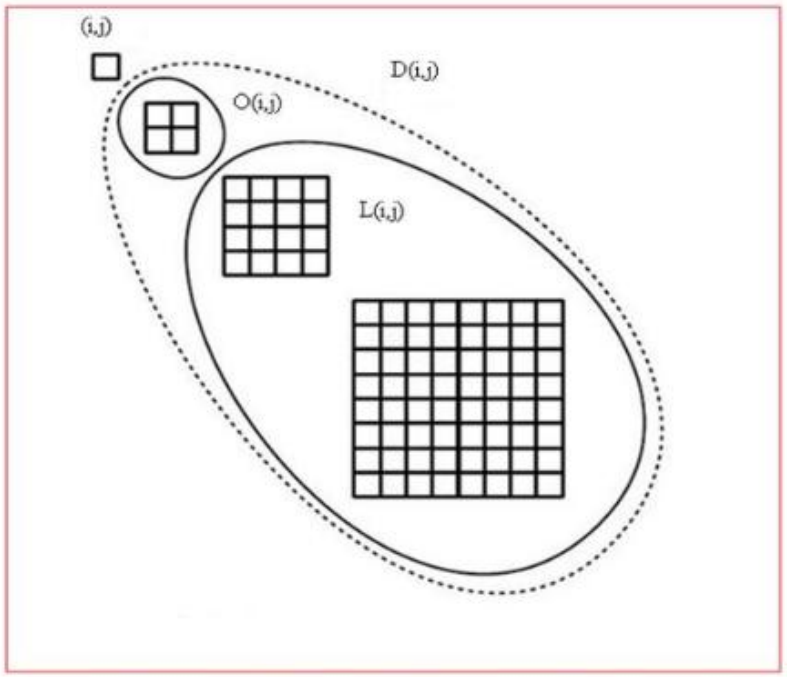


Figure III.6 : Terminologie SPIHT pour les descendants

III.2.1. Fonctionnement du SPIHT :

Pour réaliser pratiquement le codage, SPIHT stocke l'information de signification dans trois listes ordonnées :

- La Liste des Coefficients Significatifs (LCS),
- La Liste des Coefficients Non significatifs (LCN),
- La Liste des Ensembles Non significatifs (LEN)

Ce sont des listes de coordonnées(i,j), dans la LCN et LCS, ils représentent des coefficients et dans la LEN, ils représentent soit l'ensemble D(i,j) (une entrée de type A) ou l'ensemble L(i,j) (une entrée de type B). La LCN contient les coordonnées de coefficients qui étaient insignifiants dans la précédente passe. Dans le passage de la passe courante, ils sont testés, et ceux qui sont significatifs sont déplacés vers la LCS. Similairement, les ensembles de LEN sont évalué dans leur ordre d'entrée, et quand un ensemble est trouvé significatif, il est supprimé de cette liste puis est partitionné. Les nouveaux ensembles avec plus d'un élément sont ajoutés à la fin de LEN avec le type (A ou B), alors que les simples coefficients sont ajoutés à la fin de LCS ou LCN suivant leur signification. La liste LCS contient les coordonnées des coefficients qui seront visités dans la prochaine passe de raffinement. La passe de raffinement transmet le bit le plus significatif des entrées de LCS.

Nous allons donner maintenant l'algorithme de base du SPIHT 2D. Pour cela, nous définissons l'opérateur de signification σTn qui évalue la signification d'un sous-ensemble E pour un seuil donné Tn:

$$\sigma Tn(E) = \begin{cases} 1 \; si \; \exists w \; \in E : |w| \geq Tn \\ 0 \; si \; \forall w \; \in E : |w| < Tn \end{cases} \qquad \text{(III-4)}$$

Le seuil Tn pour le premier plan de bit est égale à 2^n où n =$|\log_2(wmax)|$. Tous les coefficients d'ondelettes sont scannés afin de trouver le maximum de w (i,j) après l'exécution de la transformée en ondelettes discrète. Pour le passage suivant, on divise le seuilTnpar2.

III.2.2. Algorithme SPIHT 2D :

1. Initialisation :

   Sortie$n = |\log_2(wmax)| \Leftrightarrow T_n = 2^{|\log_2(wmax)|}; LCS = \emptyset; LCN = \{(i,j) \in LL\}$.

   LEN contient les mêmes coefficients que LCN excepté ceux qui n'ont pas de descendants.

2. Passe de signification :

   2.1 Pour chaque $(i,j) \in$ LCN faire :

   2.1.1 Sortie σTn(i,j)

   2.1.2 siσTn(i,j)=1 alors mettre (i,j) dans LCS et coder le signe de w(i,j).

   2.2 Pour chaque $(i,j) \in$ LEN faire :

   2.2.1 Si l'entrée est de type A

   a) Sortie σTn$\big(D(i,j)\big)$

   b) si σTn$\big(D(i,j)\big) = 1 \; alors$:

   - Pour chaque $(l,m) \in \big(O(i,j)\big)$ faire :
     - Sortie σTn(l,m)
     - Si σTn$(l,m) = 1$ alors mettre (l,m) dans LCS et coder le signe de w(l,m).
     - Si σTn$(l,m) = 0$ alors mettre (l,m) à la fin de LCN

   - Si $L(i,J) \neq \emptyset$ alors mettre (i,j) à la fin de LEN comme une entrée de type B et aller à 2.2.2 sinon supprimer (i,j)de LEN.

   2.2.2 Si l'entrée est de type B

   a) Sortie σTn$\big(L(i,j)\big)$

b) si σTn$\big(L(i,j)\big) = 1\ alors$:

- Mettre chaque $(l,m) \in \big(O(i,j)\big)$à la fin de LEN comme une entrée de type A.
- supprimer (i,j)de LEN

3. Passe de raffinement :

Pour chaque coefficient $(i,j) \in$ LCS à l'exception de ceux ajoutés au cours de la dernière passe. Ecrire le n^{ième} bit le plus significatif de $|w(i,j)|$.

4. Modification du pas de quantification :Tn ← Tn$/2$ et aller à l'étape 2.

Pour obtenir l'algorithme de décodage, il suffit simplement de remplacer le mot Sortie par Entrée dans l'algorithme précédent. De plus, le décodeur exécute une tâche supplémentaire en modifiant l'image reconstruite. Pour un seuil Tn donné, quand un coefficient est déplacé dans la LCS, il est évident que Tn<w (i, j) < 2Tn. Ainsi, le décodeur utilise cette information plus le bit de signe juste après l'insertion dans la LCS pour mettre (i, j)= ±1.5Tn. De manière identique, pendant la passe de raffinement le décodeur ajoute ou soustrait Tn/2 à (i, j) quand on reçoit les bits de la représentation binaire de |w (i,j)|. De cette manière, la distorsion baisse à la fois pendant les deux passes. Enfin, on notera que contrairement à EZW, SPIHT produit directement des symboles binaires. Ainsi, un codeur arithmétique n'est pas obligatoire même s'il est souvent implanté pour améliorer les performances de codage. L'ensemble des améliorations proposées dans SPIHT par rapport à EZW, en fait la référence des méthodes de codage inter-bande (c.à.d. Celles qui utilisent les redondances inter-échelles entre les sous-bandes pour coder les coefficients d'ondelettes).

III.2.3. Exemple d'application de l'algorithme SPIHT [1]:

Nous allons maintenant voir le déroulement du SPIHT sur la même matrice de décomposition en TO-2D donnée par SHAPIRO.

26	6	13	10
-7	7	6	4
4	-4	4	-3
2	-2	-2	0

Figure III.7 : Exemple de matrice sur le codeur SPIHT

Comme on l'a vu précédemment, l'algorithme SPIHT maintient trois listes de coefficients : la liste des coefficients significatifs LCS, la liste des coefficients non significatifs LCN et la liste des ensembles non significatifs LEN. La fonction σTn (i, j) est égale à '0' si tous les descendants de (i, j) sont en dessous du seuil Tn (arbre de zéros) et '1' dans le cas contraire.
SPIHT considère deux types d'arbres de zéros : le type A où tous les descendants ne sont pas significatifs et le type B où tous les descendants, à l'exception d'au moins un des enfants, n'est pas significatif. Le déroulement de l'algorithme SPIHT sur les données de la figure III.6 et ses différentes étapes sont décrits ci-dessous :

- **Initialisation**

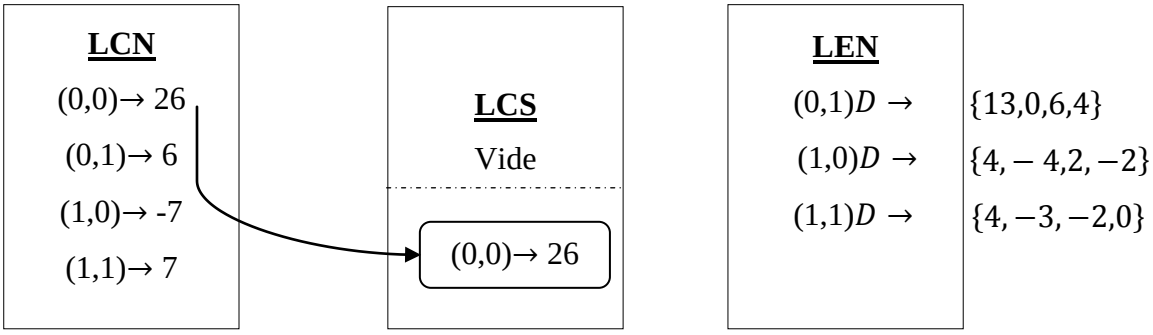
$\boldsymbol{N = log_2(26) \rightarrow T = 16}$

LCN
(0,0)→ 26
(0,1)→ 6
(1,0)→ -7
(1,1)→ 7

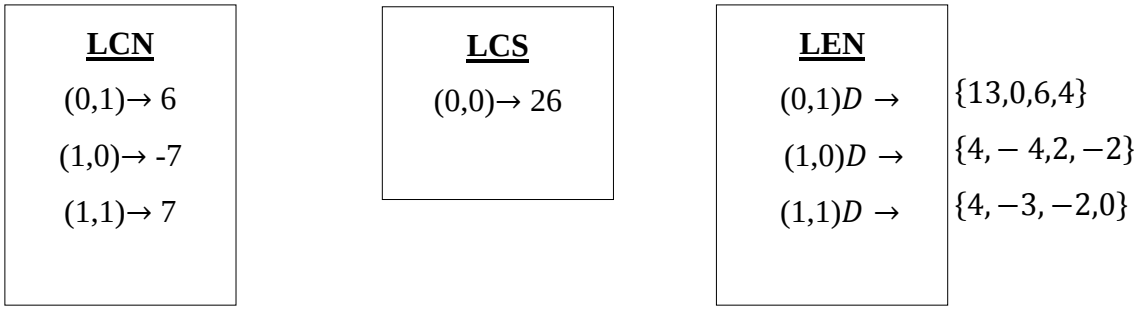
LCS
vide

LEN
$(0,1)D \rightarrow \{13,0,6,4\}$
$(1,0)D \rightarrow \{4,-4,2,-2\}$
$(1,1)D \rightarrow \{4,-3,-2,0\}$

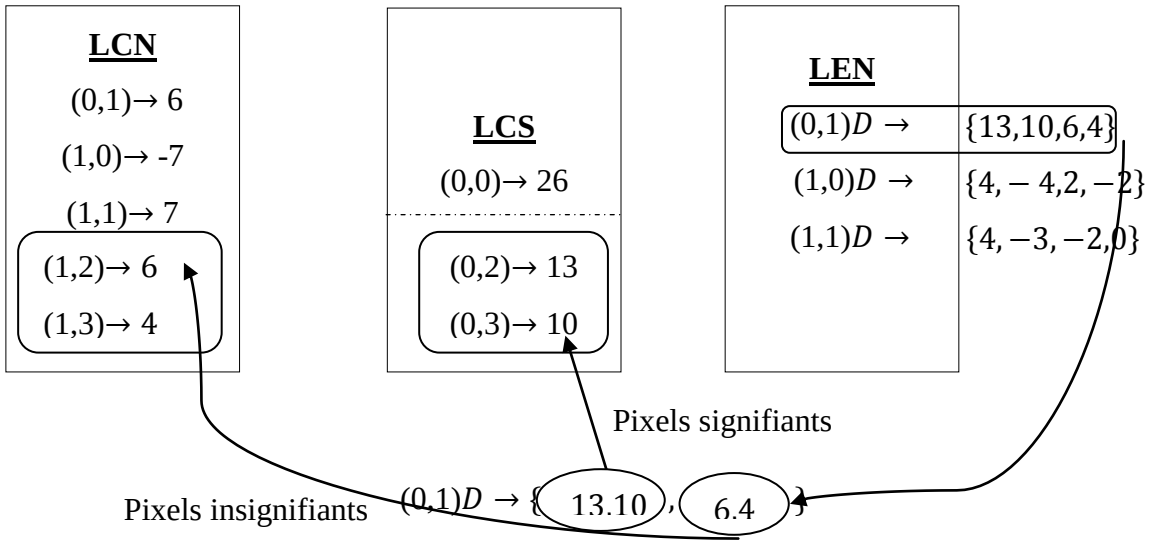
- **Pendant la première passe de tri ($T = 16$)**



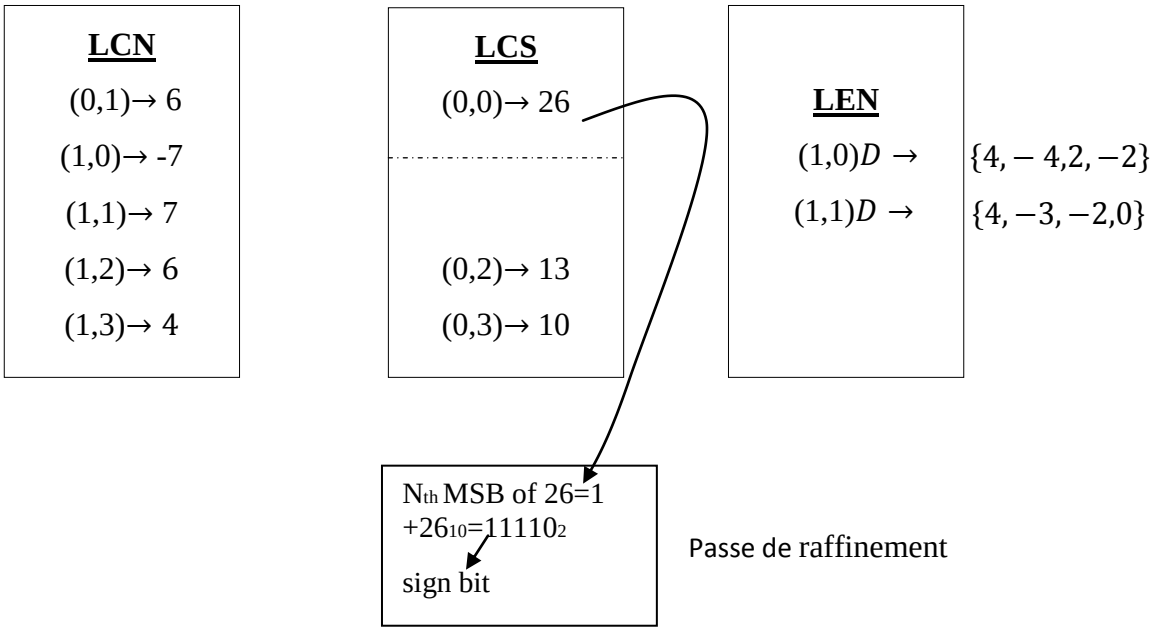
- **Après la première passe de tri ($T = 16$)**



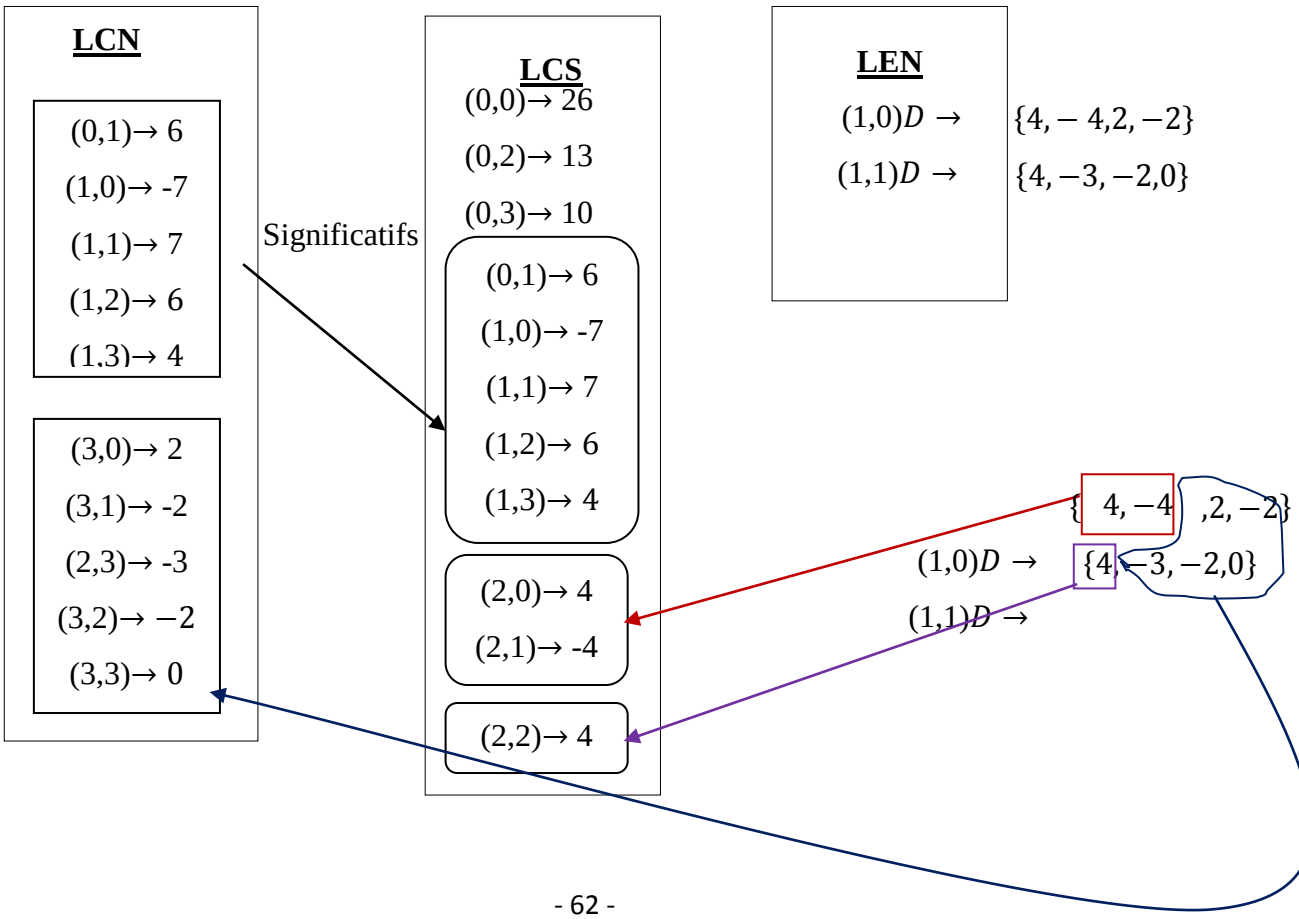
- **Pendant la seconde passe de tri (T=8)**



- **Après la seconde passe de tri (T=8)**



- **Pendant la troisième passe de tri (T=4)**



- **Après la troisième passe de tri (T=4)**

LCN

(3,0)→ 2

(3,1)→ -2

(2,3)→ -3

(3,2)→ −2

LCS

(0,0)→ 26

(0,2)→ 13

(0,3)→ 10

(0,1)→ 6

(1,0)→ -7

(1,1)→ 7

(1,2)→ 6

(1,3)→ 4

(2,0)→ 4

(2,1)→ -4

(2,2)→ 4

LEN

Vide

III.3.Algorithme de codage EBCOT [22] :

Le codeur EBCOT (Embedded Block Coding with Optimized Truncation) est issu des travaux de TAUBMAN. Mais, n'appartient pas à la famille des schémas de codage à arbre de zéros. Le codeur EBCOT est la référence pour la compression d'images à base de la transformée en ondelettes, adopté par le standard JPEG2000.

L'algorithme EBCOT est l'évolution de l'algorithme LZC (Layered Zero Coding).C'est un algorithme à codage progressif pour les images fixes. Il permet un codage emboîté, grâce à la décomposition en ondelettes d'une image. Son originalité réside dans le ré-ordonnancement du train binaire final, destiné à présenter une progressivité optimale du point de vue débit/distorsion.

III.3.1) Description de l'algorithme :

Le codeur EBCOT est un codeur par blocs, c'est-à-dire qu'il code les coefficients d'ondelettes bloc par bloc. En général, les blocs sont de taille 64 × 64. La portion de code-bloc que nous allons utiliser dans notre exemple est présentée sur la figure (7). On fait l'hypothèse que ce bloc est dans une sous-bande LH. Les nombres représentent les valeurs des coefficients d'ondelettes après la quantification. Le parcours des coefficients est effectué par colonnes de 4 coefficients, et bande par bande, comme illustré sur la figure (7), où la première ligne de coefficients d'ondelettes appartient à la bande précédente et la dernière à la bande suivante.

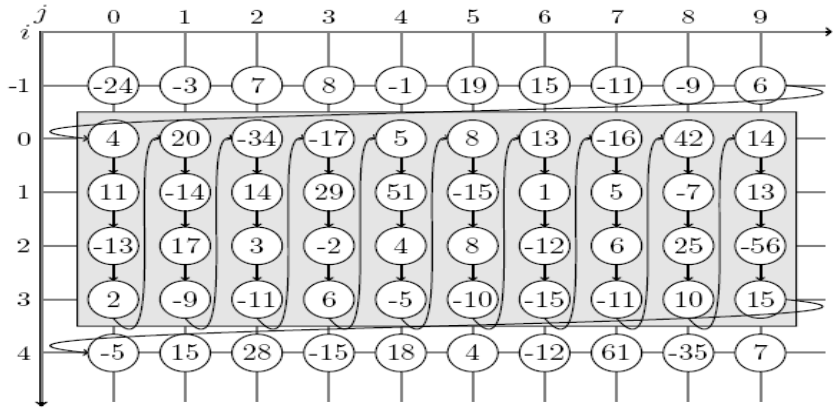


Figure III.8 : Portion de code-bloc utilisée dans notre exemple.

Le coefficient le plus fort en valeur absolue dans ce code-bloc vaut 61 (en position (4,7)). Il y a donc 6 plans de bits en amplitude dans ce code-bloc ($2^6 = 64$). On les numérote de 0 à 5, du plan de bits le moins significatif au plan de bits le plus significatif. Ce nombre de plans de bits est inclus dans le train binaire émis afin de signaler au décodeur à partir de quel plan de bits le codage est effectué.

Le codeur contextuel EBCOT est un codeur par plan de bits. Sur chaque plan de bits, trois passes sont utilisées : Passe de signification (significance pass), Passe d'affinage (refinement pass) et Passe de nettoyage (cleanup pass). Tous les plans de bits (à partir du plan de bits non nul) seront codés par les trois passes successives à l'exception du premier plan de bits rencontré signifiant. Un bit est dit signifiant s'il est égal à '1' et insignifiant s'il est égal à '0', qui sera codé seulement avec la passe cleanup. Chaque bit d'un plan de bits est codé une seule fois par l'une des trois passes.

Il existe quatre primitives de codage utilisées par ces passes :

- Primitive RL (Run-Lengh) :

Elle prend la valeur 0 si aucun coefficient n'est significatif connu et n'est de voisin significatif connu (mode Run-Lengh) et passe à un 1 dès que le codeur rencontre un coefficient significatif (sortie du mode Run-Lengh) et émit un symbole dans le contexte UNI pour coder la position du coefficient signifiant.

- Primitive ZC (Zero Coding) :

Permet de coder les bits des coefficients à la sortie du mode Run-Lengh. Elle contient 9 contextes qui prennent en compte la signifiance connue des huit coefficients voisins du bit à coder. Ces contextes sont définis dans le tableau 5.

- Primitive SC (Sign Coding) :

Permet de coder le signe du coefficient signifiant. Cinq contextes sont associés à cette primitive et sont décrits dans le tableau 4.

- Primitive MR (Magnitude Refinement)

Elle permet de coder la valeur du coefficient signifiant. Trois contextes sont associés à cette primitive et sont décrits par le tableau 8.

a) **Codage du premier plan de bits**

Dans le premier plan de bits (n = 5) représenté sur la figure(8), seule la passe de nettoyage est utilisée. En effet, la passe de propagation de la signifiance ne peut être utilisée parce qu'aucun coefficient d'ondelettes n'est signifiant avant ce plan de bits. Comme il n'y a pas de coefficient signifiant connu, on ne peut pas propager le codage de la signifiance autour des coefficients signifiants. Pour la même raison, la passe d'affinage de l'amplitude ne peut être utilisée. En effet, il faut d'abord trouver les coefficients signifiants pour ensuite pouvoir affiner leur amplitude dans les plans de bits moins significatifs. La seule passe utilisée pour coder le premier plan de bits est donc la passe de nettoyage. Cette passe est nommée ainsi, parce qu'elle sert à coder tous les bits qui n'ont pas été codés par la passe de propagation de la signifiance ou la passe d'affinage de l'amplitude.

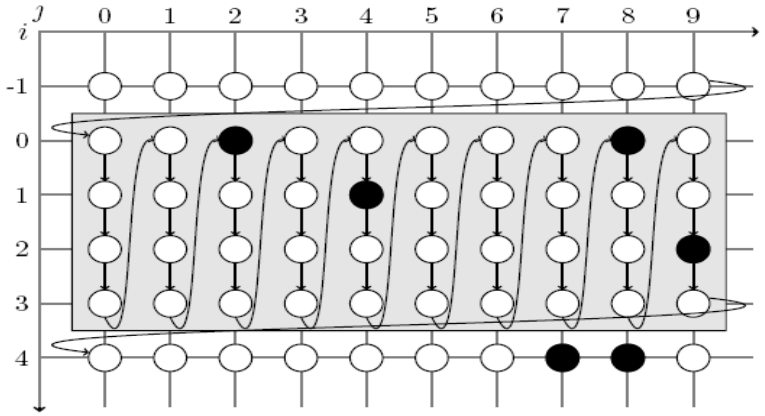


Figure III.9 : Plan de bits *n* = 5.

b) **Passe de nettoyage**

On commence par le codage de la colonne *j* = 0. Avant le codage de cette colonne, on ne connaît pas de coefficient signifiant dans cette colonne. De plus, on ne connaît pas de coefficients voisins à cette colonne qui soient signifiants. Ces deux conditions réunies, le codeur passe en mode *Run-Length*. Le codeur parcourt alors la colonne *j* = 0. Au plan de bits *n* = 5, aucun coefficient n'est significatif dans cette colonne. Le codeur émis donc un symbole '0' dans le contexte (RL) pour signaler une colonne de zéros.

De même, sur la colonne *j* = 1, avant le codage, il n'y a pas de coefficient signifiant connu, ni de voisin signifiant *connu*. On reste donc dans le mode *Run-Length*. Comme aucun coefficient de cette colonne n'est significatif au plan de bits *n* = 5, le codeur émit un symbole '0' dans le contexte (RL).

Pour le codage de la colonne *j* = 2, le codeur est toujours dans le mode *Run-Length* car les deux conditions sont réunies. Cependant, le coefficient (0,2) est signifiant au plan de bits *n* = 5. Le codeur émit alors le symbole '1' dans le contexte (RL) pour signaler la sortie du mode *Run-Length*. De plus, le codeur code la position du premier coefficient signifiant de la colonne. Il émit les symboles'00' dans le contexte (UNI). Cela permet de signaler au décodeur la position exacte de sortie du mode Run-Length. Le signe de chaque coefficient d'ondelettes est codé dès que le coefficient a été trouvé signifiant. Il faut maintenant coder le signe du coefficient (0,2).

$\bar{\chi}^h$	$\bar{\chi}^v$	κ^{SC}	$\hat{\chi}$
1	1	SC4	1
1	0	SC3	1
1	-1	SC2	1
0	1	SC1	1
0	0	SC0	1
0	-1	SC1	-1
-1	1	SC2	-1
-1	0	SC3	-1
-1	-1	SC4	-1

Tableau III.3 : Prédiction du signe $\hat{\chi}$et contextes κ^{SC} de la primitive SC

Pour le codage du signe, le codeur EBCOT utilise une prédiction basée sur les signes des quatre coefficients voisins. Les prédictions sont écrites dans le tableau 5 pour les sous-bandes LL, LH ou HH. Les valeurs $\bar{\chi}^h$ et $\bar{\chi}^v$ représentent le signe des deux voisins horizontaux et verticaux respectivement. Ces quantités sont inversées dans le tableau de prédiction pour le codage des sous-bandes HL. Une valeur $\bar{\chi}$vaut 1 si les deux voisins sont positifs ou si un voisin est positif et l'autre n'est pas encore signé, 0 si les deux voisins ne sont pas encore signés ou de signes opposés, et -1 si les deux voisins sont négatifs ou si un voisin est négatif et l'autre n'est pas encore signé. La valeur $\hat{\chi}$représente la prédiction du signe du coefficient codé. Si cette prédiction est exacte, le symbole '0' est émis dans le contexte SC respectif, sinon le symbole '1' est émis dans ce même contexte.

Dans le cas du coefficient (0,2), comme il n'y a aucun voisin signé, on est dans le contexte (SC0). Le signe prédit est donc positif. Cette prédiction étant incorrecte, le symbole '1' est émis dans le contexte (SC0).

Dans la colonne j = 2, il reste à coder les symboles (1,2) à (3,2). Comme on est sorti du mode *Run-Length*, c'est la primitive ZC qui est utilisée. Neuf contextes, notés ZC0 à ZC8, sont utilisés dans cette primitive. Ces contextes prennent en compte la signifiance connue des huit coefficients voisins du bit à coder. Ils sont définis dans le tableau 5.Les valeurs κ^h,κ^v et κ^d

représentent respectivement le nombre de coefficients voisins horizontaux, verticaux, et diagonaux qui sont déjà signifiants au moment du codage.

Sous-bandes LL et LH			Sous-bandes HL			Sous-bandes HH		
κ^h	κ^v	κ^d	κ^h	κ^v	κ^d	κ^d	$\kappa^h + \kappa^v$	κ^{ZC}
0	0	0	0	0	0	0	0	ZC0
0	0	1	0	0	1	0	1	ZC1
0	0	≥ 2	0	0	≥ 2	0	≥ 2	ZC2
0	1	x	1	0	x	1	0	ZC3
0	2	x	2	0	x	1	1	ZC4
1	0	0	0	1	0	1	≥ 2	ZC5
1	0	≥ 1	0	1	≥ 1	2	0	ZC6
1	≥ 1	x	≥ 1	1	x	2	≥ 1	ZC7
2	x	x	x	2	x	≥ 3	x	ZC8

Tableau III.4 : Contextes κ^{ZC} de la primitive ZC. «x» signifie n'importe quelle valeur.

Le coefficient (1,2) n'a qu'un seul coefficient voisin signifiant, le coefficient (0,2). De plus, le coefficient (1,2) n'est pas signifiant au plan de bits $n = 5$. Le symbole '0' est donc émit dans le contexte (ZC3). Les coefficients (2,2) et (3,2) n'ont pas de voisins signifiants et ne sont pas signifiants au plan de bits $n = 5$. Deux symboles '0' sont donc codés dans le contexte (ZC0).

Le codeur s'apprête maintenant à coder la colonne $j = 3$. Comme le coefficient (0,2), voisin à cette colonne, est connu signifiant au plan de bits $n = 5$, le mode *Run-Length* n'est pas activé. C'est la primitive ZC qui est utilisée pour coder les quatre bits de cette colonne. Les symboles codés sont '0(ZC5)', '0(ZC1)', '0(ZC0)', et '0(ZC0)'.

Il n'y a pas de coefficient voisin de la colonne $j = 4$ connu signifiant. De plus, aucun des coefficients de cette colonne n'est connu signifiant. Le codeur entre donc dans le mode *Run-*

Length. Comme le coefficient (1,4) est signifiant au plan de bits n = 5, le codeur émit le symbole '1' dans le contexte (RL) et signale la position du premier coefficient signifiant de la colonne en codant sa position par les symboles '01' dans le contexte (UNI). Le signe de ce coefficient est ensuite codé. Le coefficient (1,4) est positif. Le contexte est (SC0). La prédiction d'un signe positif est correcte. Le symbole '0' est donc codé dans le contexte (SC0). Les deux bits suivant sont codés par la primitive ZC. Les symboles codés sont '0(ZC3)' et '0(ZC0)'.

Le reste du codage du plan de bits n = 5 par la passe de nettoyage est décrit dans le tableau 5.

	Symboles codés et contextes	Commentaires
j = 0	0(RL)	Run-Length.
j = 1	0(RL)	Run-Length.
j = 2	1(RL)	Run-Length jusqu'au coefficient (0,2).
	00(UNI)	Position du premier coefficient signifiant.
	1(SC0)	Le signe prédit par le contexte (SC0) est incorrect.
	0(ZC3) 0(ZC0) 0(ZC0)	Reste de la colonne codé par la primitive ZC.
j = 3	0(ZC5) 0(ZC1) 0(ZC0) 0(ZC0)	Pas de Run-Length parce que le coefficient (0,2) est signifiant.
j = 4	1(RL)	Run-Length jusqu'au coefficient (1,4).
	01(UNI)	Position du premier coefficient signifiant.
	0(SC0)	Le signe prédit par le contexte (SC0) est correct.
	0(ZC3) 0(ZC0)	Reste de la colonne codé par la primitive ZC.
j = 5	0(ZC1) 0(ZC5) 0(ZC1) 0(ZC0)	Pas de Run-Length parce que le coefficient (1,4) est signifiant.
j = 6	0(RL)	Run-Length.
j = 7	0(RL)	Run-Length.
j = 8	1(RL)	Run-Length jusqu'au coefficient (0,8).

	00(UNI)	Position du premier coefficient signifiant.
	0(SC0)	Le signe prédit par le contexte (SC0) est correct.
	0(ZC3) 0(ZC0)	Reste de la colonne codé par la primitive ZC.
	0(ZC0)	
j = 9	0(ZC5)	Pas de Run-Length parce que
	0(ZC1)	le coefficient (0,8) est signifiant.
	1(ZC0)	Un coefficient signifiant dans le contexte (ZC0).
	1(SC0)	Le signe prédit par le contexte (SC0) est incorrect.
	0(ZC3)	

Tableau III.5 : Codage du plan de bits n = 5 par la passe de nettoyage.

c) Codage du second plan de bits

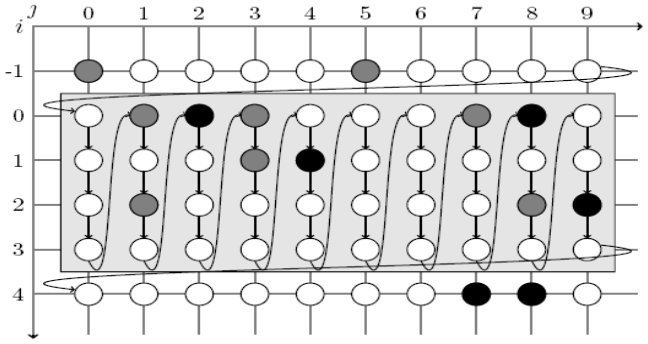


Figure III.10 : Plan de bits *n* = 4.

Le second plan de bits (*n* = 4) est représenté sur la figure9. Les coefficients signifiants au précédent plan de bits codé (plus significatifs) sont représentés en noir. Les coefficients trouvés signifiants au plan de bits *n* = 4 sont représentés en gris. Sur ce second plan de bits, les trois passes de codage sont utilisées : la passe de propagation de la signifiance est d'abord utilisée pour coder les bits voisins des coefficients déjà identifiés signifiants au plan de bits *n* = 4 ou aux plans de bits plus significatifs. En effet, les forts coefficients d'ondelettes sont souvent agglomérés. C'est pourquoi on commence par rechercher des coefficients signifiants autour des coefficients les plus forts déjà connus. Ensuite, la passe d'affinage de l'amplitude est utilisée pour affiner l'amplitude des coefficients qui étaient déjà signifiants aux plans de bits précédemment codés. Enfin, la passe de nettoyage est utilisée pour coder les bits restant.

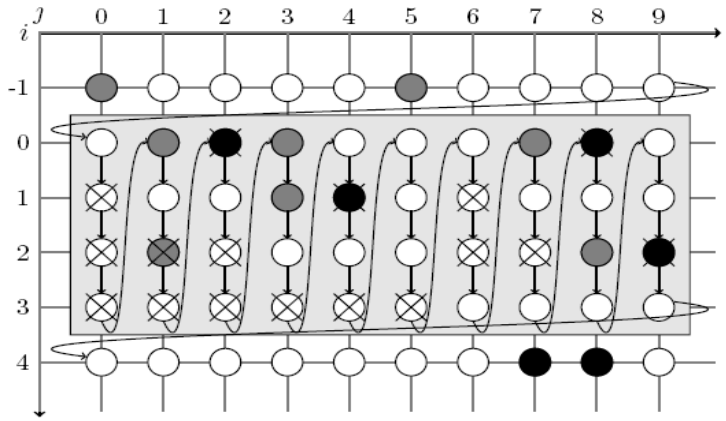


Figure III.11 : Passe de propagation de la signifiance sur le plan de bits *n* = 4.

c-1) Passe de propagation de la signifiance

Lors de la lecture du plan de bits *n* = 4, on trouve que le coefficient (0,0) a un voisin déjà signifiant : le coefficient (-1,0) qui est devenu signifiant au plan de bits n=4. Le bit à la position (0,0) est donc codé dans la passe de propagation de la signifiance. Dans la passe de propagation de la signifiance, seule la primitive ZC est utilisée. Le symbole émit pour coder le bit (0,0) est donc '0' dans le contexte (ZC3).

Les coefficients (1,0) à (3,0) n'ont pas de voisins signifiants connus. Ils ne sont donc pas codés par la passe de propagation de la signifiance. Ces coefficients sont marqués d'une croix sur la figure (10).

Comme le coefficient (0,2) est signifiant, les bits aux positions (0,1) et (1,1) sont codés. Le bit (0,1) est signifiant au plan de bit *n* = 4. Le symbole 1(ZC6) est émis. Le signe du coefficient d'ondelette est ensuite codé par le symbole 1(SC3) car le signe prédit est négatif alors que le coefficient codé est positif. Le symbole 0(ZC3) est ensuite émit pour coder le bit (1,1).

Les coefficients (2,1) et (3,1) ne sont pas codés. Le coefficient (0,2) n'est pas codé non plus car il était déjà signifiant au plan de bits précédent. Le codeur émit le symbole 0(ZC3) pour coder le bit (1,2). Les coefficients (2,2) et (3,2) ne sont pas codés.

Le reste du codage du plan de bits *n* = 4 par la passe de propagation de la signifiance est décrit dans le tableau 6.

	Symboles codés et contextes	Commentaires
$j = 0$	0(ZC3)	Codage du coefficient (0,0).
$j = 1$	1(ZC6)	Codage du coefficient (0,1).
	1(SC3)	Le signe prédit dans le contexte (SC3) est incorrect.
	0(ZC3)	Codage du coefficient (1,1).
$j = 2$	0(ZC3)	Codage du coefficient (1,2).
$j = 3$	1(ZC6) 0(SC3)	Codage du coefficient (0,3).
	1(ZC7) 0(SC2)	Codage du coefficient (1,3).
	0(ZC3)	Codage du coefficient (2,3).
$j = 4$	0(ZC7) 0(ZC3)	Codage des coefficients (0,4) et (2,4).
$j = 5$	0(ZC3) 0(ZC5) 0(ZC1)	Codage des coefficients (0,5) à (2,5).
$j = 6$	0(ZC1) 0(ZC1)	Codage des coefficients (0,6) et (3,6).
$j = 7$	1(ZC5) 1(SC3)	Codage du coefficient (0,7).
	0(ZC3) 0(ZC3)	Codage des coefficients (1,7) et (3,7).
$j = 8$	0(ZC3)	Codage du coefficient (1,8).
	1(ZC5) 1(SC3)	Codage du coefficient (2,8).
	0(ZC4)	Codage du coefficient (3,8).
$j = 9$	0(ZC5) 0(ZC3) 0(ZC3)	Codage des coefficients (0,9), (2,9), et (3,9).

Tableau III.6 : Codage du plan de bits $n = 4$ par la passe de propagation de la signifiance.

c.2) Passe d'affinage de l'amplitude

Dans la passe d'affinage de l'amplitude, seule la primitive de codage MR est utilisée. Cette primitive code les bits des coefficients déjà signifiants aux plans de bits précédents en utilisant les trois contextes du tableau 8notés MR0 à MR2. La variable $\tilde{\sigma}$vaut 0 si c'est la première fois que la passe d'affinage de l'amplitude est appliquée au coefficient codé, sinon la variable $\tilde{\sigma}$vaut 1. Les variables κ^h et κ^v sont les mêmes que pour la primitive de codage ZC.

$\tilde{\sigma}$	$\kappa^h + \kappa^v$	κ^{MR}
0	0	MR0
0	$\neq 0$	MR1
1	x	MR2

Tableau III.7 : Contextes κ^{MR} de la primitive MR.

Les coefficients qui étaient signifiants dans les plans de bits précédemment codés ont leur amplitude affinée dans cette passe. Le symbole 0(MR1)' est donc émis pour affiner l'amplitude du coefficient (0,2). De mêmes, les symboles '1(MR1)', '0(MR1)', et '1(MR1)' sont émis pour affiner l'amplitude des coefficients respectivement en positions (1,4), (0,8), et (2,9).

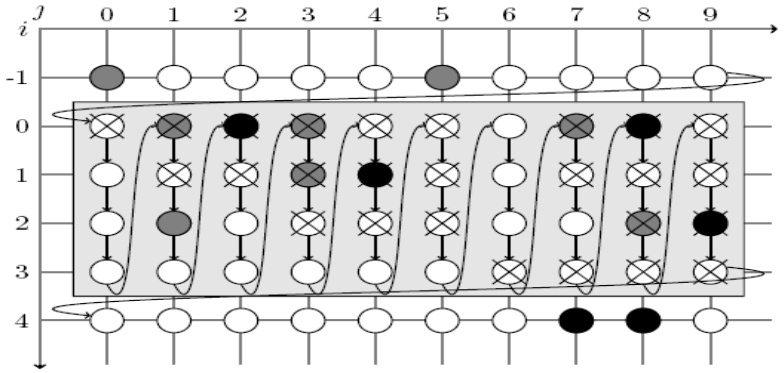


Figure III.12 : Passe de nettoyage sur le plan de bits *n* = 4.

c.3) Passe de nettoyage

La passe de nettoyage est utilisée pour coder les bits qui n'ont pas été codés par les deux passes précédentes. Ce sont les bits qui ne sont pas marqués d'une croix sur la figure(11). Le codage est décrit dans le tableau 8.

	Symboles codés et contextes	Commentaires
j = 0	0(ZC1) 0(ZC0) 0(ZC0)	
j = 1	1(ZC0) 0(SC0) 0(ZC3)	Codage du coefficient (2,1).
j = 2	0(ZC6) 0(ZC1)	
j = 3	0(ZC0)	
j = 4	0(ZC0)	
j = 5	0(ZC0)	
j = 6	0(ZC6) 0(ZC1) 0(ZC0)	
j = 7	0(ZC5)	

Tableau III.8 : Codage du plan de bits *n* = 4 par la passe de nettoyage.

d-Suite et fin du codage

Dans la suite, le déroulement du codage est similaire jusqu'au plan de bits *n* = 0. Le codeur arithmétique MQ code chaque bit émit en utilisant des probabilités d'apparitions différentes pour chaque contexte. Ces probabilités conditionnées par les contextes sont ajustées à chaque bit codé. Ce processus de codage des codes-blocs constitue la première partie du codeur EBCOT. Elle est nommée Tier1. Dans Tier2, la seconde partie du codeur, les morceaux de trains binaires, issus du codage des plans de bits des codes-blocs dans Tier1, sont organisés de façon à optimiser le train binaire final en termes de débit-distorsion. Ce processus est nommé PCRD-OPT (Post-Compression Rate-Distorsion Optimisation).

CONCLUSION :

Le but de la compression d'images est bien de modifier la représentation initiale des données, pour qu'elles occupent moins de place. Cette nouvelle représentation sera décodée durant une procédure de décompression pour reconstruire l'image. Plusieurs algorithmes de compression utilisant les ondelettes ont été proposés, dont les plus utilisés sont EZW, SPIHT et EBCOT. En remarque que chaque codeur a ses inconvénients et ses avantages, ainsi il n'y a pas de codeur universel utilisé dans l'opération de compression ; On doit donc changer d'algorithme pour chaque méthode de compression et la qualité des images reconstituées peut être améliorée en utilisant progressivement plus de coefficients d'ondelettes. Notons aussi que la séquence de transmission des coefficients d'ondelettes dépend des résultats désirés. Pour avoir une image reconstituée qui ressemble à l'image originale, il faut transmettre au début des coefficients d'ondelettes importants qui contribuent à faire apparaître sur l'image reconstituée les informations de hautes fréquences comme les bords et par la suite, des coefficients d'ondelettes de moindre importance qui à leur tour contribuent à faire apparaître sur l'image reconstituée les informations de basses fréquences comme des zones de niveaux de gris constants.

Introduction

Pour une évaluation pratique des performances de compression des trois algorithmes (EZW, SPHIT et EBCOT) sur les quels est basée notre étude comparative, nous les avons appliqués sur trois images de références très utilisées dans lacompression d'images. Il s'agit des images Lena, Goldhill et Peppers. Par ailleurs, nous avons décrit brièvement les étapes du processus de compression et décompression d'images basé sur la transformée en ondelettes.Ainsi que les résultats obtenus par l'application de chacun des trois algorithmes sur les trois images de références choisies.

Les paramètresles plus utilisés dans le domaine de la compression des images ; témoignés de la qualité des images ainsi que leurs tailles mémoire occupée sont le rapport signal sur bruit crête (PNSR) et le Taux de compression Rc (Bpp : Bits par pixel).

IV.1.Processus de compression et décompression d'images paralgorithmes de codage par plan de bits [17] :

Le processus de compression etde décompression d'images fixes donné dans la littérature suit les étapes résumées par le schéma synoptique suivant :

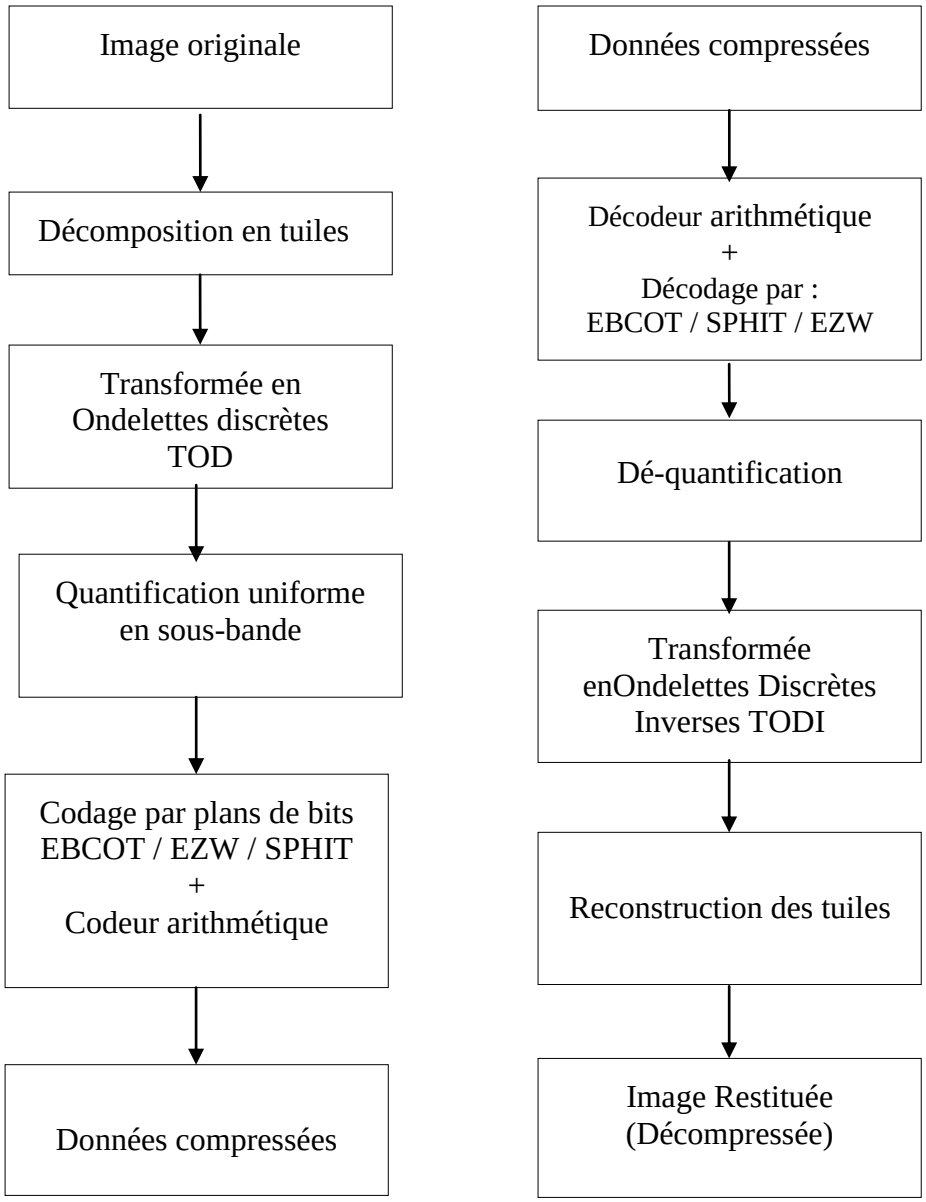


Figure.IV.1 : Schéma Général desétapes de la compression et de la décompression.

IV.1.1 Le partitionnement en tuiles :

Chaque image est divisée en plusieurs rectangles de même taille 16x16 ou 32x32 ou 64x64 … (sauf exception faite pour les zones situées aux extrémités de l'image) que l'on appelle Tuiles (figure.IV.2) et qui ne se recouvrent pas. Les Tuiles seront ensuite chacune codée séparément avec leurs propres paramètres. Ce partitionnement est particulièrement utile dans les applications qui possèdent des ressources mémoire limitées.

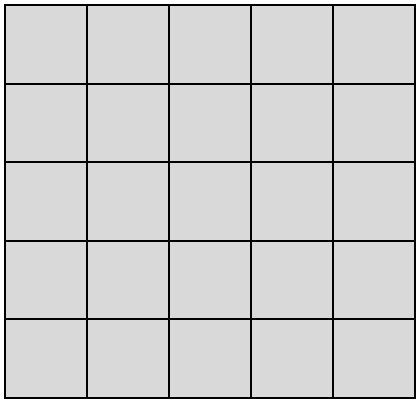


Figure.IV.2 : Exemple de partitionnement en tuiles.

IV.I. 2. Transformée en ondelettes(TOD) :

La TOD à deux dimensions est basée sur l'algorithme pyramidal de S. MALLAT, elle est réalisée à partir d'une décomposition unidimensionnelle appliquée successivement sur les lignes puis sur les colonnes de l'image.

Après l'application d'un banc de filtre (H, G), on obtient une nouvelle image divisée en quatre sous-bandes à une résolution telle que représentée par la figure suivante :

	HH		GH
A_j Image approximée		D^1_j Détails verticaux	
	HG		GG
D^2_j Détails horizontaux		D^3_j Détails diagonaux	

Figure.VI.3: Image après TOD

Il est claire qu'après l'application de la TOD sur l'image, on obtientune décomposition de cette dernière en sa composante principale (HH) (faible résolution) et trois composantesde détails représentant les détails horizontaux (GH), verticaux(HG) et diagonaux(GG).

On applique à l'image originale une TOD, telle qu'on obtient une sous-bande approximative (bande lissée) et des sous-bandes de détails (hautes fréquences). On peut réappliquer la TOD sur la sous-bande approximée autant de fois que nécessaire selonle niveau de résolution qui nous permet d'obtenir la meilleure reconstruction possible d'images.En général, le niveau de résolution estau moins égal à 3.

IV.1. 3. Transformée en ondelettes Inverse TODI :

Cette étape représente l'opération inverse de la transformée en ondelettes. La transformée inverse a pour but de reconstruire l'image originale à partir des sous bandes détails et de l'image de basse résolution. La reconstruction est donc obtenue par la somme du filtrage passe bas (h) des coefficients d'approximations et du filtrage filtre passe haut (g) des coefficients de détails auxquels sont intercalés des zéros.

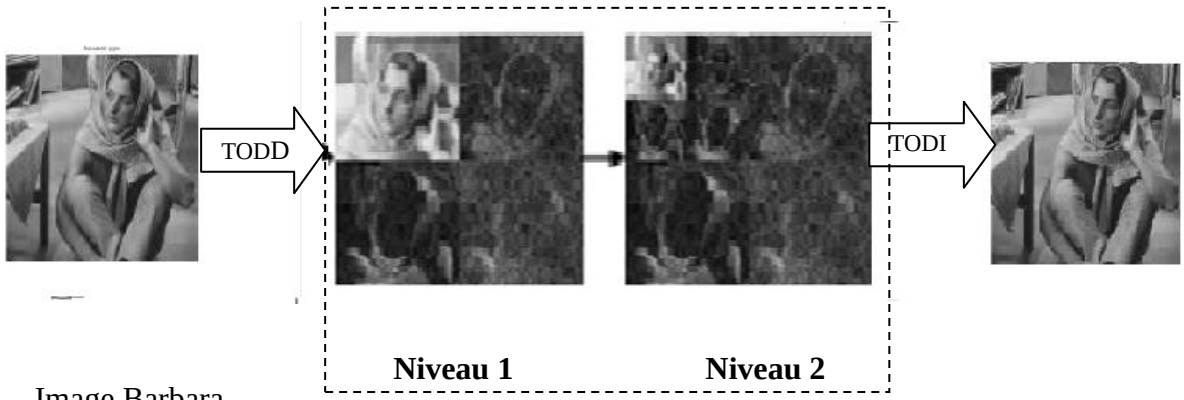


Figure.VI.4 : Application de la TOD et TODI sur l'image Barbara

- **Choix de l'ondelette et du niveau de décomposition :**

La qualité de l'image décompressée par ondelettes dépend essentiellement du choix de l'ondelette utilisée. 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. Dans la littérature, plusieurs études réalisées sur des bases d'ondelettes réelles et entières ont abouti aux conclusions suivantes :

- 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 qui donnent des résultats meilleurs.
- Pour la compression avec pertes, la base bi-orthogonale de Daubechies 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 donne de bons résultats pour des images de paysage

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 pour les ondelettes bi-orthogonales, cinq niveaux de décomposition pour compacter le maximum d'énergie dans la sous-bande d'approximation.

IV.1. 4. Quantification / Dé-quantification :

La transformation en ondelettes est suivie d'une quantification scalaire permettant de réduire la dynamique des coefficients. Cette opération est destructive, sauf si le pas de quantificationest égal à 1 (ce qui signifie qu'aucune quantification n'est appliquée). Mathématiquement, la fonction de quantification est donnée par l'équation suivante :

$$q_b(u,v) = signe\big(a_b(u,v)\big) \times \frac{(|a_b(u,v)|)}{\Delta_b} \qquad \text{(IV-1)}$$

Avec$q_b(u,v)$représentent les indices de quantification obtenus et $a_b(u,v)$ sont les coefficientsd'ondelettes à quantifier de la sous-bande b et Δ_b est le pas de quantification en fonction de ladynamique et du type de la sous-bande.

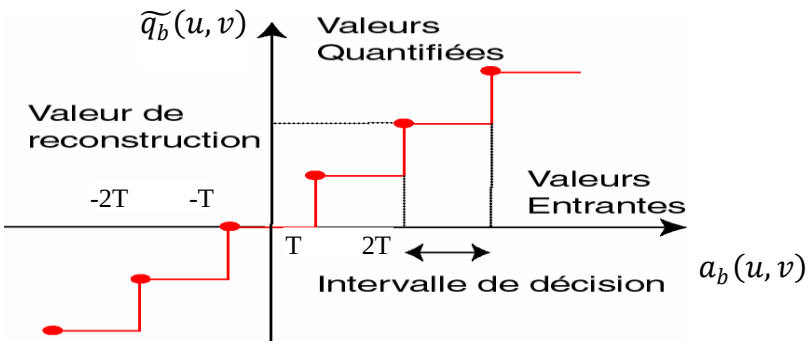


Figure.IV.5 :Exemple de quantification uniforme à intervalles égaux.

- **Règle de Dé-quantification :**

La valeur de dé-quantification se calcule comme suit :

$$\widetilde{q_b}(u,v) = \left[q_b(u,v) + r \times signe\left(q_b(u,v)\right)\right] \times \Delta_b \qquad \text{(IV-2)}$$

Où : $\widetilde{q_b}(u,v)$ est la valeur reconstruite, $q_b(u,v)$ est l'index de quantification, Δ_b est le pas dequantification,$signe(q_b(u,v))$ dénote le signe de $q_b(u,v)$et r détermine le biais lié à lareconstruction et la valeur de r n'est pas normalisée par le standard JPEG2000.

A noter que la valeur de « r » dépend du signe de

$$q_b(u,v) : \begin{cases} r = 0{,}5 \quad quand \quad signe(q_b(u,v)) = +1 \\ r = 0{,}375 \quad quand \quad signe(u,v)) = -1 \end{cases}$$

IV.1.5. Codage en plans de bits [22]:

Il s'agit ici de réduire la redondance inter-coefficients. Le principe d'un codage en plans de bits consiste en ladécomposition d'une image à plusieurs niveaux en une série d'images binaires, puis chaque image binaireest compressée.

.

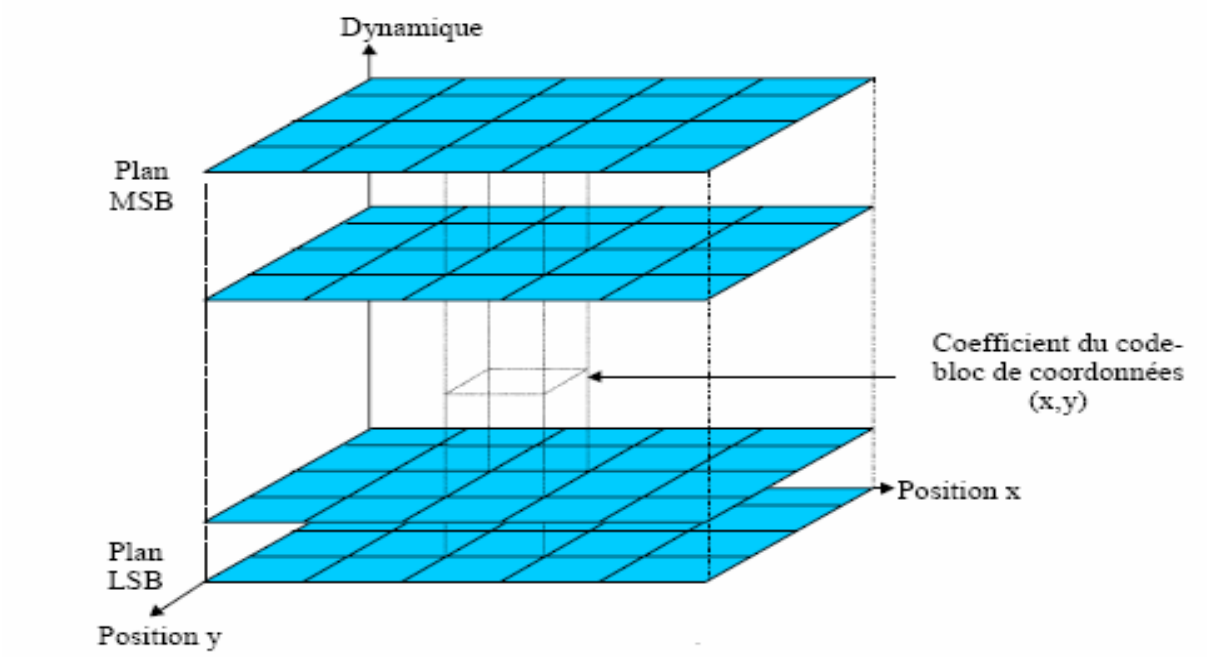
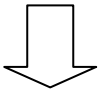


Figure.IV.6 : Représentation d'un plan de bits.

Pour mieux illustrer la représentation d'un code-bloc en plans de bits, nous donnons unexemple pratique comme le montre la figure.IV.7 ; Le code-bloc utilisé est un bloc de3 x 3 coefficients et sa représentation en plans de bits est faite sur trois plans de bits.

3	-1	7
4	-5	2
6	1	-2



Plan de bits MSB

0	0	+1
+1	-1	0
+1	0	0

+1	0	+1
0	0	+1
+1	0	-1

Plan de bits LSB

+1	-1	+1
0	+1	0
0	+1	0

Figure.IV.7 : Représentation d'un code-bloc de 3x3 coefficients en trois plans de bits.

Dans le codage par plans de bits, on a utilisé trois algorithmes qui sont détaillés dans le troisième chapitre (EZW, SPHIT et EBCOT) suivi d'un codage arithmétique.

Pour le décodage, ça dépend du type de codeur utilisé lors du codage suivi d'un décodage arithmétique.

IV.1. 6 Le codage arithmétique :

Le codage arithmétique est prouvé performant, d'où son introduction dans la norme JPEG2000.

- La procédure de codage arithmétique est la suivante :
  - Calculer la probabilité associée à chaque symbole dans la chaîne à coder.
  - Associer à chaque symbole un sous intervalle proportionnel à sa probabilité, dans l'intervalle [0.1] (l'ordre de rangement des intervalles sera mémorisé car il est nécessaire au décodeur)
  - Initialiser la limite inférieure de l'intervalle de travail à la valeur 0 etla limite supérieure à la valeur 1.
  - Tant qu'il reste un symbole dans la chaîne à coder:

• largeur = limite supérieure - limite inférieure.

• limite inférieure =limite inférieure + largeur * (limite basse dusous intervalle du symbole).

• limite supérieure = limite inférieure + largeur * (limite haute dusous intervalle du symbole).

- La limite inférieure code la chaîne de manière unique.

On remarque que le premier symbole de la chaîne fixée est le premier chiffreaprès la virgule du code final.

- Décrivons le principe du codage et de décodage sur un exemple :

Considérons le codage du message « ATLAS ». Les probabilités des caractères sont les suivantes :

Caractère	Probabilité
A	1/10
L	1/5
S	1/10
T	1/10

Etape 1

Dans l'intervalle général de probabilité [0,1], chaque symbole se voit affecter un intervalle de probabilité (la manière d'affecter cet intervalle n'ayant pas d'importance).

Caractère	Probabilité	Intervalle
A	1/10	$0{,}10 \leq r < 0{,}20$
L	1/5	$0{,}20 \leq r < 0{,}40$
S	1/10	$0{,}40 \leq r < 0{,}50$
T	1/10	$0{,}50 \leq r < 0{,}60$

Etape 2

Le codage s'effectue progressivement à partir de la première lettre dumessage. Puisque c'est A, le codage du message doit être compris entre 0,10 et0,20. La deuxième lettre est T; dans le sous intervalle 0,10 à 0,20, le codage sera compris entre 50 et 60 %. Autrement dit, à ce stade « AT » sera compris entre [0,10+(0,20-0,10)x0,50] et [0,10+(0,20-0,10)x0,60], soit entre 0,15 et 0,16.

Au niveau de la troisième lettre L, il doit être compris entre [0,15+(0,16-0,15)x0,20] et [0,15+(0,16-0,15)x0,40] soit entre 0,152 et 0,154.

En continuant, on arrive au tableau suivant :

Caractère successif	Limite inférieure	Limite supérieure
A	0,1	0,2
T	0,15	0,16
L	0,152	0,154
A	0,1522	0,1524
S	0,15228	0,1523

Etape 3

Le codage du message est constitué par la dernière limite inférieure : 0,15228.

- **Procédure de décodage arithmétique :**

Le décodage s'effectuera de manière unique à partir de ce nombre, suivant le mécanisme inverse. Puisque 0,15228 est compris entre 0,1 et 0,2 la première lettre du message est A. On peut maintenant soustraire la limite inférieure correspondant à A, soit 0,10 ce qui donne 0,05228 et diviser ce résultat par la longueur de l'intervalle correspondant à A, soit 0,10 ce qui donne : 0,5228. Ce nombre étant compris entre 0,5 et 0,6, la deuxième lettre est T. En continuant, on trouve dans l'ordre tous les symboles du message.

Nombre codé	Limite inférieure	Limite supérieure	Symbole
0,15228	0,1	0,2	A
0,5228	0,5	0,6	T
0,228	0,2	0,4	L
0,14	0,1	0,2	A
0,4	0,4	0,5	S

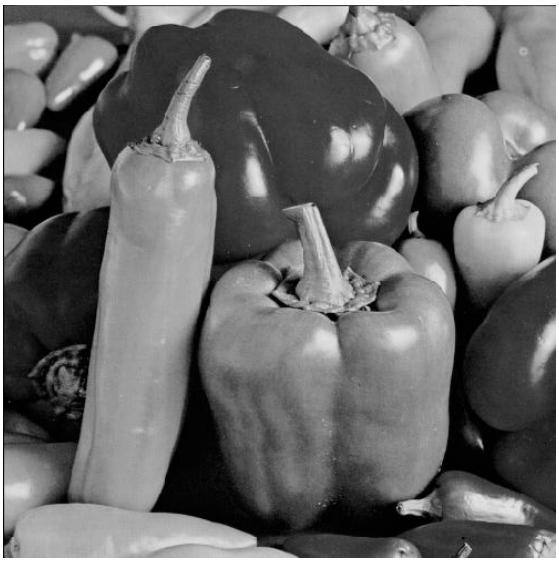
Cette technique se montre un peu plus lente que celle de Huffman mais elle présente des taux de compression supérieurs.

IV.2. Tests et Résultats [24]:

IV.2.1. Images tests :

Notre étude comparative s'est concentrée sur trois images tests en niveaux de gris de références à savoir : LENA, GOLDHILL et PEPPERS, de dimension 512*512 codée chacune d'elles sur 8 bits.

(a)



(b)

(c)

Figure IV-7 Images testsutilisées
(a) Lena; (b)Peppers; (c) Goldhill

IV.2. 2. Paramètres d'évaluation :

Plusieurs paramètres nous permettent d'évaluer une image compressée à savoir,la qualité (PNSR), la distorsion (MSE), le taux de compression Tc ou Rc (Bpp : nombre de bits par pixel).Dans le cas de notre étude, nous avons opté pour le PNSR(en db) et le taux de compression Rc(en bpp).

IV.2. 3. Résultats :

Les résultats sont obtenus pour une décomposition en ondelettes à cinq niveaux en utilisant le filtre biorthogonal 9/7.

Pour évaluer les performances des trois algorithmes (EZW,SPHIT et EBCOT),nous avons utilisés les résultats du rapport signal sur PSNR(db) en fonction du taux de compression RC(bpp) obtenus après application de chacun de ces algorithmes sur trois images (Lena ;Goldhillet Peppers).

Ces résultats sont illustrés par le tableau suivant :

Images	Rc (Bpp)	PSNR (Db)		
		EZW	SPIHT	EBCOT(JPEG2000)
LENA	0,25	30,50	32,90	33,20
	0,5	***35,20***	***37,20***	***37,30***
	0,75	36,9	38,8	38,9
	1	38,60	40,40	**40,50**
GOLDHILL	0,25	28,20	29,80	29,90
	0,5	***31,60***	***33,10***	***33,20***
	0,75	33,2	34,8	34,9
	1	34,8	36,5	**36,6**
PEPPERS	0,25	30,30	32,50	32,70
	0,5	***35,00***	***35,80***	***35,90***
	0,75	36,15	37,05	37,25
	1	37,30	38,30	**38,60**

Tableau IV.1 : Résultats des PSNR (en db) en fonction du taux de compression (en bpp) des différents algorithmes appliqués sur les trois images tests.

En général, la compression se fait soit en bas débit (< 0.5) ou en moyen débit (~ 0.5) ou en haut débit(>0.5),selon l'objectif de la compression à faire.

Dans le domaine de la compression d'images, le critère le plus pertinent pour évaluer une méthode reste toujours le critère visuel. Seul l'œil humain et l'analyse que fait le cerveau,des images captés, peuvent juger de leurs qualités.

Nous allons présenter dans ce qui suit les images restituées par les trois algorithmes pour un moyen débit de 0.5 bpp.

(a) EZW : PSNR=35.3 db

(b) SPHIT : PSNR=37.2 db

(c) EBCOT (JPEG2000) : PSNR=37.3db

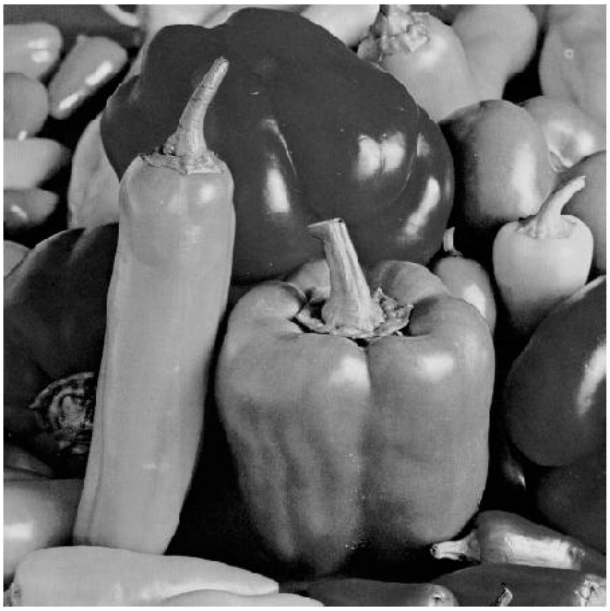
Figure IV.8. Images LENA restituées à Rc = 0.5 bpp

(a) EZW : PSNR=31.7db

(b) SPHIT : PSNR=33.1db

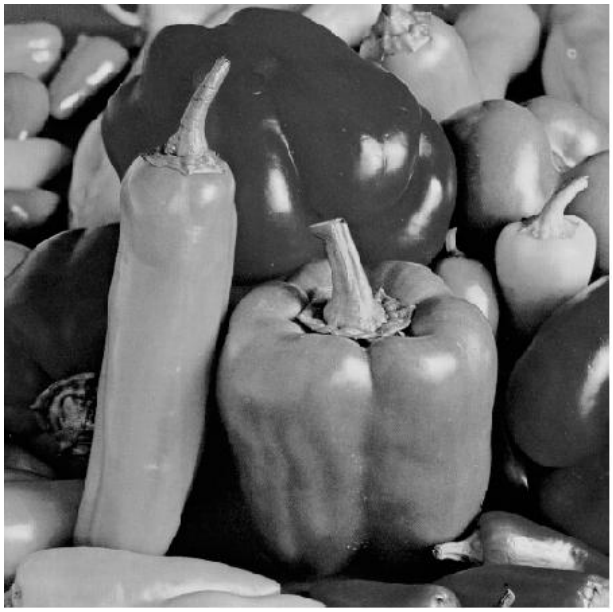
(c) EBCOT (JPEG2000) : PSNR=33.2db

Figure IV.9. Images Goldhill restituées à RC = 0.5 bpp



(a) EZW : PSNR=35db

(b) SPHIT : PSNR=35.80db



(c) EBCOT (JPEG2000) : PSNR=35.9db

Figure IV.10. Images Peppers restituées à RC = 0.5 bpp

(a) EZW : PSNR=35 db

(b) SPHIT: PSNR=35.80db

(c) EBCOT (JPEG2000) : PSNR=35.9db

Figure IV.11.Échantillons d'images Peppers restituées à Rc = 0.5 bpp avec un Zoom.

- **Synthèse des images restituées :**

D'après les figures présentées précédemment, nous avons remarqué que les images restituées par les trois codeurs sont de qualité appréciable ; avec une supériorité pour celles restituées par EBCOT (JPEG2000). Cependant, les images restituées par SPHIT sont proches de celles de EBCOT (JPEG2000) et que les images restituées par le codeur EZWsont de moindre qualitépar rapport à celles issues des deux autres codeurs (voir Figure IV.10,dans le cas des images PEPPERS zoomées où la différence de qualité entre les images restituées est plus visible).

Afin de mieux illustrer les résultats obtenus par les trois algorithmes sur les images tests, nous avons procédé aux tracés des histogrammes du PSNR (db) en fonction du taux de compression Rc(bpp), qui sont représentés ci-dessous :

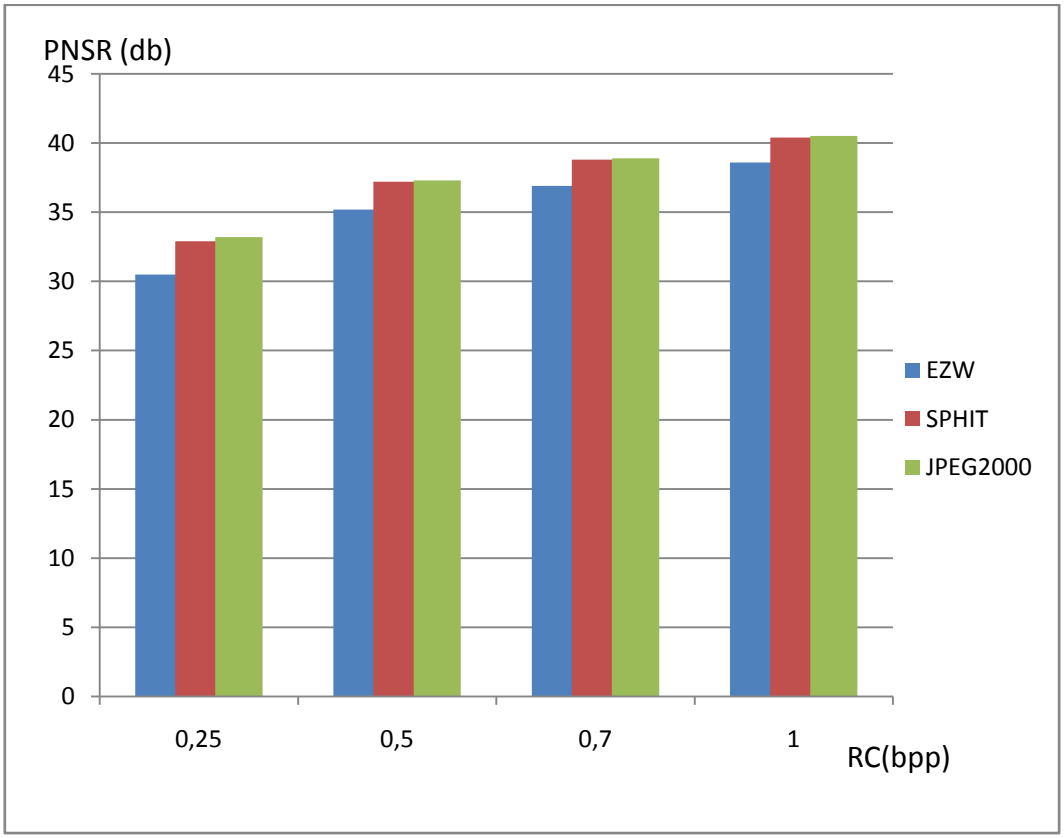


Figure.IV.12 :Histogramme PSNR (db) en fonction du taux de compression Rc(bpp) pour l'image LENA

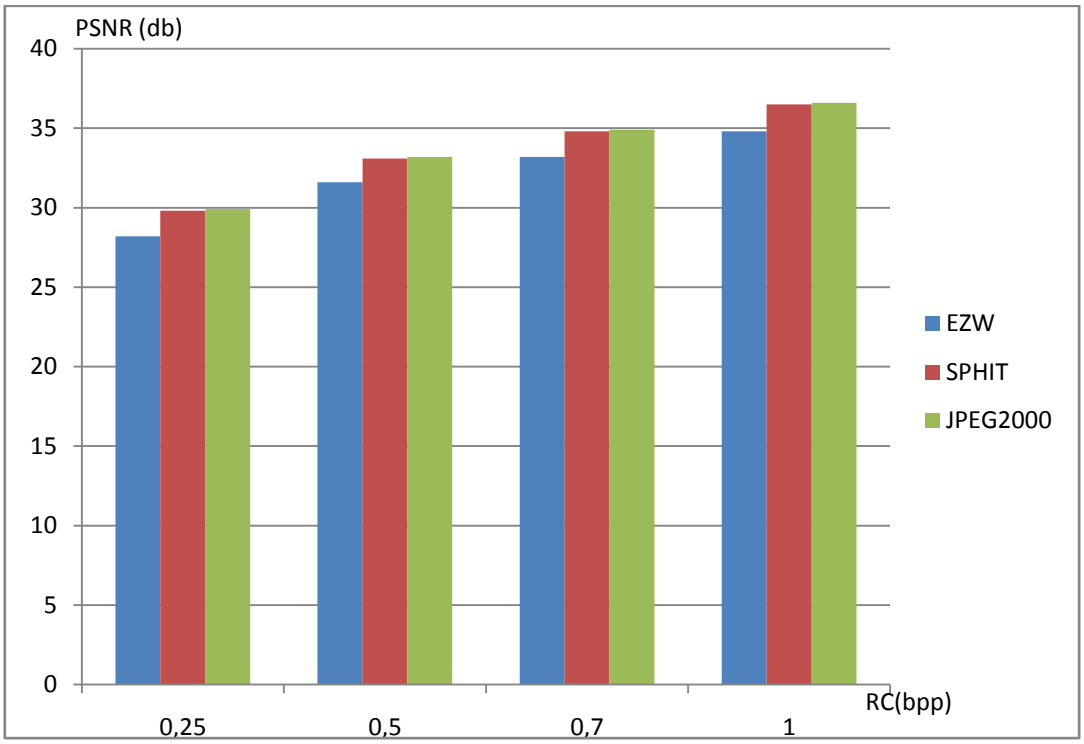


Figure.IV.13 : HistogrammePSNR (db) en fonction du taux de compression Rc(bpp) pour l'image GOLDHILL

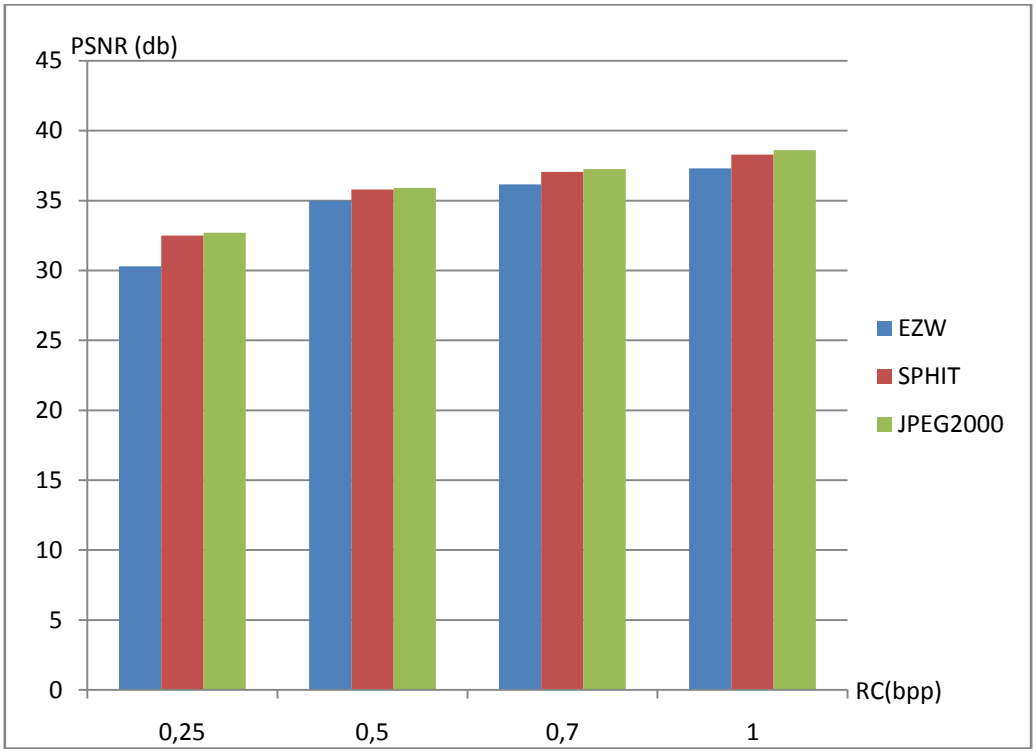


Figure.IV.14 : Histogramme PSNR (db) en fonction du taux de compression Rc(bpp) pour l'image PEPPERS

- **Interprétationsdes tracés d'histogrammes :**

A partir destracés d'histogrammes des trois images tests, nous avons noté que :

- ✓ Le PSNR diminue, quand le taux de compression augmente et ce pour les trois codeurs. Ceci s'explique par le fait que quand, le taux de compression est élevé, l'image est fortement quantifiée.Signifiant qu'une grande quantité d'information de hautes fréquences (détails) est perdue. Ce qui engendre une dégradation de l'image restituée, entrainant ainsi la diminution de la valeur du PSNR.
- ✓ Les histogrammes des deux codeurs SPHIT et JPEG2000 sont presque du même niveau avec un léger avantage pour le JPEG2000. Alors que, ceux du EZW sont nettement inférieurs pour les différents taux de compression et cela s'accentue pour les faibles taux de compression(RC< 0.5 bpp).

- **Synthèse des résultats selon la nature des trois images :**

Nous avons constaté que pour les trois codeurs, les meilleurs résultats obtenus du PSNR obtenus sont ceux de l'image Lena suivis par ceux de l'image Peppers et ceux de l'image Goldhill viennent en dernière position. Cela s'explique que l'image Goldhill possède beaucoup de textures différentes par rapport à Lena qui possède peu et Peppers qui possède que des objets uniformes.

Conclusion :

D'après les résultats obtenus, nous pouvons affirmer que le codeur EBCOT de la norme de compression JPEG2000 et SPIHT sont plus performants en terme de qualité d'images restituées (PSNR) que EZW et cela pour toute valeur de débit.

Pour le codeur EBCOT (JPEG2000) et SPIHT, la différence entre les images restituées est à peine visuellement perceptible. Il a fallu prendre un échantillon d'image restituée par les deux codeurs et après l'agrandissement qu'on arrive à percevoir la supériorité de la qualité des images restituées par EBCOT (JPEG2000).

CONCLUSION GENERALE

Dans ce travail, nous avons effectué une étude comparative entre les codeurs de compression d'images par plans de bits.Elle s'est portée particulièrement sur trois codeurs EZW, SPHIT et EBCOT, ce dernier est le cœur du standard de compression JPEG2000.
Après un état de l'art sur la compression d'images où nous avons exposé les différentes méthodes et standards existants et un aperçu sur la théorie des ondelettes et l'algorithme pyramidale de S.MALLAT. Nous avons effectué une étude détaillée sur les trois algorithmes, avec des exemples illustrant les étapes d'exécution de chacun des trois algorithmes.

Pour l'évaluation des performances de ces codeurs, nous avons étudié l'application de ces codeurs (EZW, SPIHT et EBCOT) sur trois images de références en niveau de gris de taille 512x512x8 bits (LENA,GODHILL et PEPPERS).

En analysant les différents résultats obtenus, nous avons noté une supériorité de EBCOT utilisé dans le schéma de compression JPEG2000 suivi de SPHIT et à la fin le codeur EZW.

La norme JPEG 2000 de par ses différentes parties, offre plusieurs fonctionnalités pour la compression des images fixes, que ça soit pour la compression avec ou sans pertes.Le standard JPEG2000 présente un excellent compromis qualité/débit/temps d'exécution surtout à bas et moyen débit et presque comparable au codeur SPIHT dans plusieurs cas. Mise à part, l'implémentation de codeur EBCOT présente une complexité dans les contextes qu'il utilise et nécessite des ressources plus puissantes en calcul, ce qui rend l'utilisation du standard JPEG2000 limitée à d'autres équipements comme les smartphones.

Notre étude aurait pu être élargie pour évaluer l'impact de l'application de ces codeurs par plans de bits sur des images couleurs, hyperspectrales et médicales,…

Notre travail nous a permis de s'initier au domaine de l'imagerie en général et celui de la compression en particulier où nous avons acquis un certain nombre de connaissances qui, sans doute nous servirons à l'avenir.

1.Coefficients des filtres d'ondelettes Orthogonales :

Désignation	**n**	$\sqrt{2} * h(n)$
Haar	0	+0.707106781187
	1	+ 0.707106781187
Daubechies 04	0	+ 0.4829629131445341
	1	+0.8365163037378079
	2	+ 0.2241438680420134
	3	-0.1294095225512604
Daubechies 06	0	+ 0.3326705529500825
	1	+ 0.8068915093110924
	2	+0.4598775021184914
	3	-0.1350110200102546
	4	-0.0854412738820267
	5	+ 0.0352262918857095
Daubechies 08	0	+ 0.2303778133088964
	1	+ 0.7148465705529154
	2	+ 0.6308807679398587
	3	-0.0278937694168599
	4	-0.1870348117190931
	5	+ 0.0308413818355607
	6	+ 0.0328830116668852
	7	-0.0105974017850690
Daubechies 10	0	
	1	+0,1601023979741929
	2	+0,6038292697971895
	3	+0,7243085284377726
	4	+0,1384281459013203
	5	-0,2422948870663823
	6	-0,0322448695846381
	7	+0,0775714938400459
	8	-0,0062414902127983
	9	-0,0125807319990820
	10	+0,0033357252854738

Daubechies 12	0 1 2 3 4 5 6 7 8 9 10 11	+0,1115407433591095 +0,4946238003984533 +0,7511339080210959 +0,3152503517091982 -0,2262646939654400 -0,1297668675672625 +0,0975016055873225 +0,0275228655303053 +0,0315820393174862 +0,0005538422011614 +0,0047772575109455 -0,0010773010853085
Désignation	**n**	$\sqrt{2} * h(n)$
Daubechies 14	0 1 2 3 4 5 6 7 8 9 10 11 12 13	+0,0778520540850037 +0,3965393194818912 +0,7291320908461957 +0,4697822874051889 -0,1439060039285212 -0,2240361849938412 +0,0713092192668272 +0,0806126091510774 -0,0380299369350104 -0,0165745416306655 +0,0125509985560086 +0,0004295779729214 -0,0018016407040473 +0,0003537137999715
Daubechies 16	0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15	+0,0544158422401072 +0,3128715909143166 +0,6756307362973495 +0,5833346836342139 -0,0158291052563853 -0,2840133429615824 +0,0004274845739124 +0,1287474266204863 -0,0173693010018090 -0,0440882539307971 +0,0139810279174001 +0,0087460940471063 -0,0048703529934520 -0,0003917403733770 +0,0006734494064506 -0,0001174767841248
Daubechies 18	0 1 2 3 4	+0,0380779473638778 +0,2438347746125858 +0,6048231236900955 +0,6572880780512736 +0,1331973858249883

	5	-0,2932737832791663
	6	-0,0968407832229492
	7	+0,1485407493381256
	8	+0,0307256814793385
	9	-0,0676328200613279
	10	+0,0002509471148340
	11	+0,0273616621236798
	12	-0,0047232047377518
	13	-0,0042815036824635
	14	+0,0018476468830563
	15	+0,0002303857635232
	16	-0,0002519631989427
	17	+0,0000393473203163
	18	
Désignation	**n**	$\sqrt{2} * h(n)$
Daubechies 20	0	+0,0266700579005473
	1	+0,1881768000776347
	2	+0,5272011889315757
	3	+0,6884590394534363
	4	+0,2811723436605715
	5	-0,2498464243271598
	6	-0,1959462743772861
	7	+0,1273693403357541
	8	+0,0930573646035547
	9	-0,0713941471663501
	10	-0,0294575366281399
	11	+0,0332126710593612
	12	+0,0036065535669870
	13	-0,0107331754833007
	14	+0,0013953517470688
	15	+0,0019924052951925
	16	-0,0006858566949564
	17	-0,0001164668551285
	18	+0,0000935886703202
	19	-0,0000132642028945

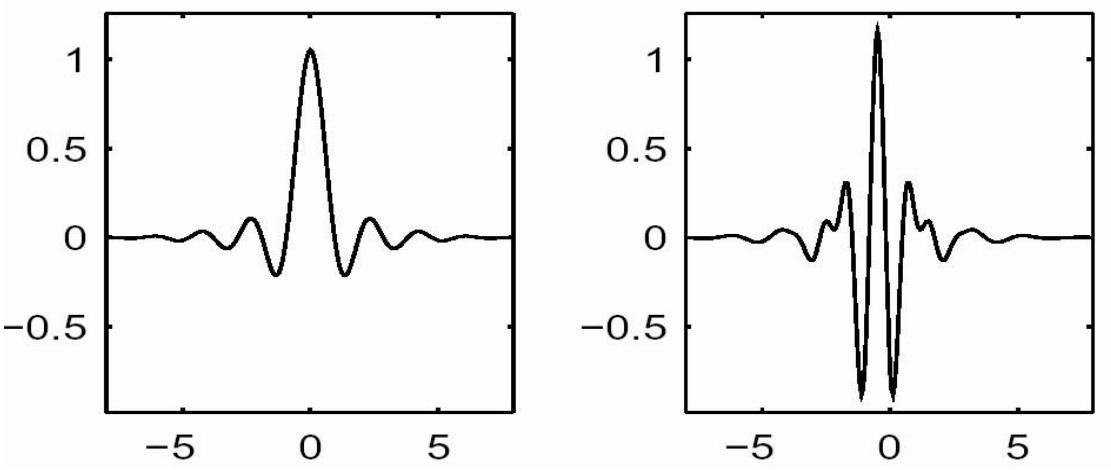


Figure A.1 Allure de l'ondelette de Meyer et de sa fonction échelle

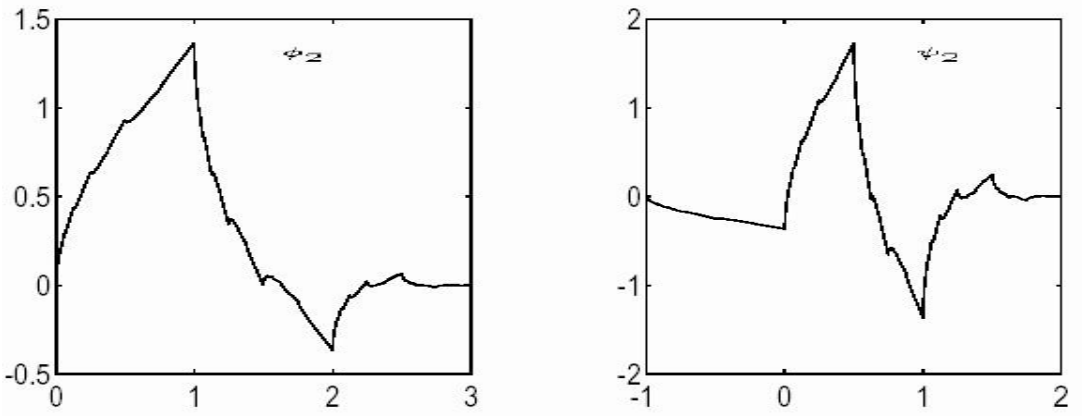


Figure A.2 Allure de l'ondeletette Daubechies 04 et de sa fonction échelle

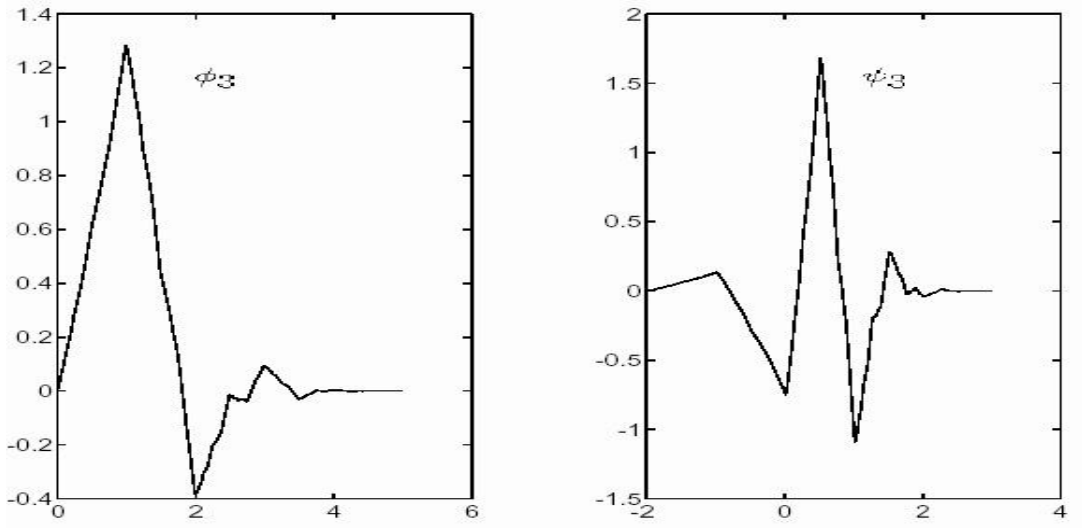


Figure A.3 Allure de l'ondelette Daubechies 06 et de sa fonction échelle

2.Ondelettes biorthogonales

Désignation	**n**	$h(n)$	$h\sim(n)$
5/3	-2 -1 0 1 2	-0.125 0.25 0.75 0.25 -0.125	0.25 0.5 0.25
5/7	-3 -2 -1 0 1 2 3	-0.05 0.25 0.60 0.25 _-0.05__	-0.010714285714 -0.053571428571 0.260714285714 0.607142857142 0.260714285714 -0.053571428571 -0.010714285714
9/7	-4 -3 -2 -1 0 1 2 3 4	0.026748757411 -0.016864118443 -0.078223266529 0.266864118443 0.602949018236 0.266864118443 -0.078223266529 -0.016864118443 _0.026748757411_	-0.045635881557 -0.028771763114 0.295635881557 0.557543526229 0.295635881557 -0.028771763114 -0.045635881557

Remarque

Seuls les filtres passe-bas sont donnés, les filtres passe-haut se déduisent en appliquant les formules suivantes:

— Pour les filtres orthogonaux:

$$g(n) = (-1)1-n\ h\ (1 - n)$$

Dans la pratique, on préfère utiliser la formule suivante qui donne un filtre causal ayant exactement le même support que $h(n)$:

$$g(n) = (-1)1-n\ h\ (L -1 - n)$$

— Pour les filtres biorthogonaux:

$$g(n) = (-1)1-n\ h\sim(1-n)$$

$$g\sim(n) = (-1)1-n\ h\ (1-n)$$

Les ondelettes biorthogonales comptes plusieurs familles, mais la plus utilisée est certainement la famille des ondelettes splines. Par ailleurs, les filtres définis par le standard JPEG2000 appartiennent à cette famille (9/7 par défaut pour la compression avec perte, 5/3 par défaut pour la compression sans perte, et 10/2 pour la compression sans perte).

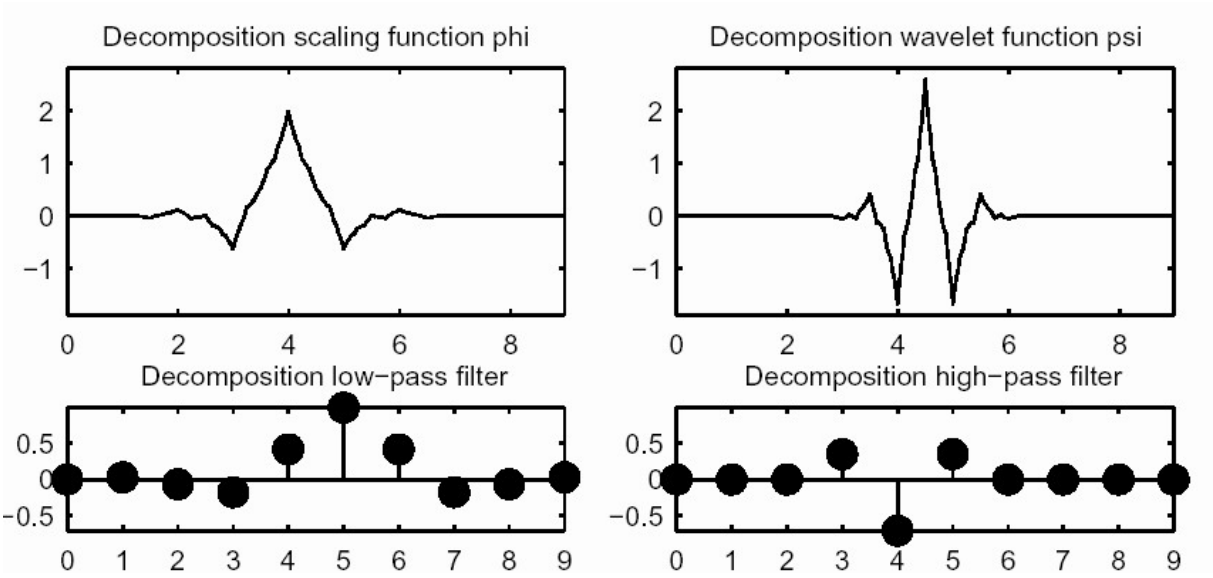


Fig. 1 Ondelettes biorthogonales 5/7: Ondelette, fonction échelle et filtres de décomposition.

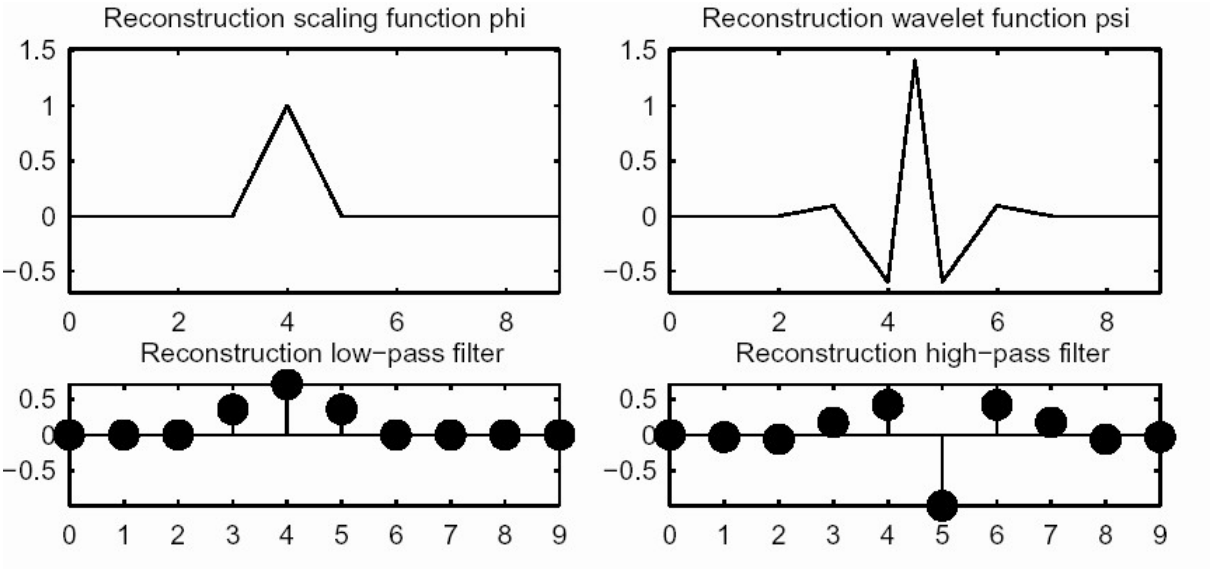


Fig. 2 Ondelettes biorthogonales 5/7: Ondelette, fonction échelle et filtres de reconstruction

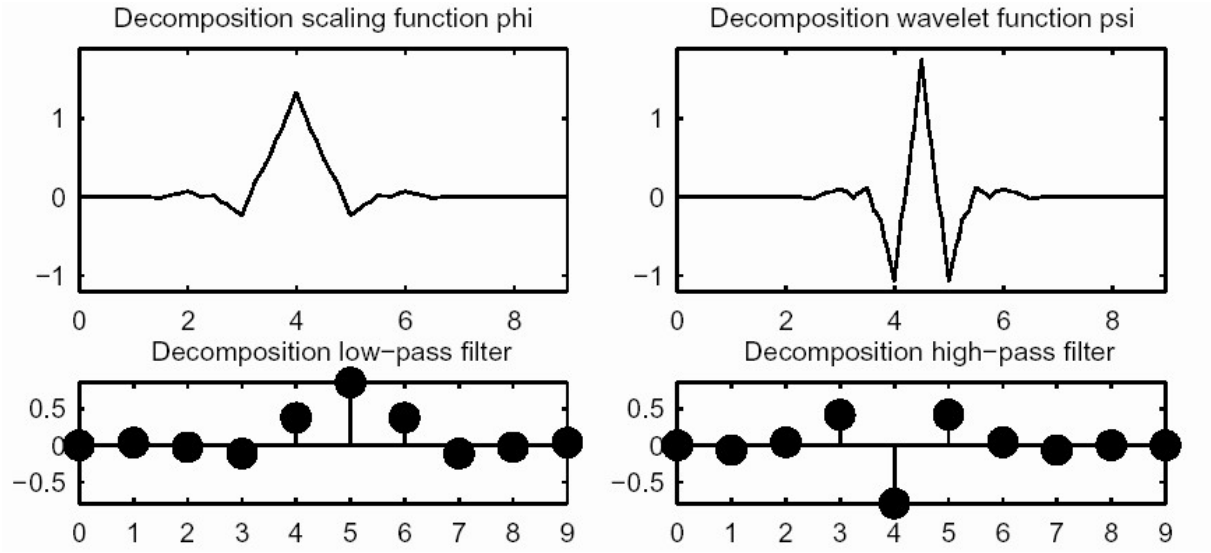


Fig. 3 Ondelettes biorthogonales 9/7: Ondelette, fonction échelle et filtres de décomposition

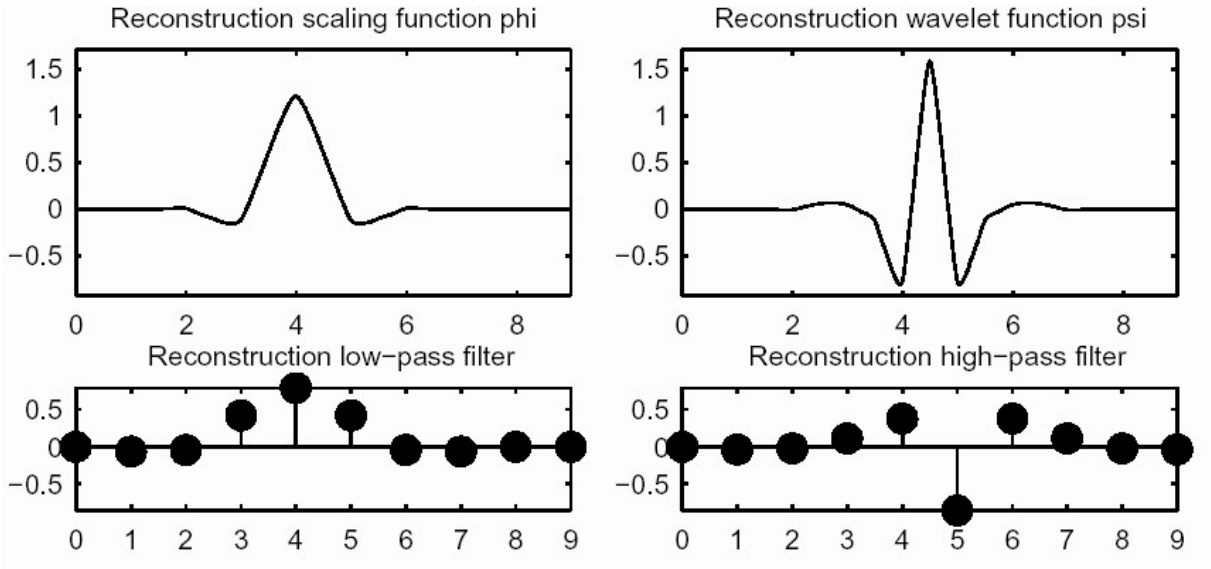


Fig. 4 Ondelettes biorthogonales 9/7:Ondelette, fonction échelle et filtres de reconstruction, Ces Figures sont tirées des fichiers de documentation de MATLAB

Arbres de zéros

Les arbres de zéros pour les coefficients d'ondelettes ont été développés pour permettre une exploitation complète de la notion de multi-résolution associée aux ondelettes. L'application de la transformée en ondelettes nous a fait ressortir la similarité de la localisation des coefficients signifiants entre les différentes sous-bandes. La propriété à l'origine des arbres de zéros est que si un coefficient est insignifiant dans une sous-bande, il est probablement in signifiant dans la sous-bande de plus haute fréquence. C'est cette idée qui a été exploitée par Shapiro avec EZW (Embedded Zerotree Wavelet) et qui a été reprise par A. SAID et W.PEARLMAN quelques années plus tard avec SPIHT (Set Partitionning in Hierarehical Tree) qui est une amélioration remarquable de EZW.

La définition d'un arbre de zéros varie selon les algorithmes. L'idée d'arbre de zéros de degré k a été développée par Y. Cho tel qu'un arbre de zéro de degré 0 est un arbre dont tous les coefficients sont nuls, un arbre de zéro de degré 1 est un arbre dont tous les coefficients sauf la racine sont nuls et un arbre de zéro de degré 2 est un arbre dont tous les coefficients sauf la racine et ses enfants directs sont nuls.EZW utilise des arbres de degré 0 alors que les arbres de degré 1 et 2 sont utilisés par SPHIT(voir les figures suivantes).

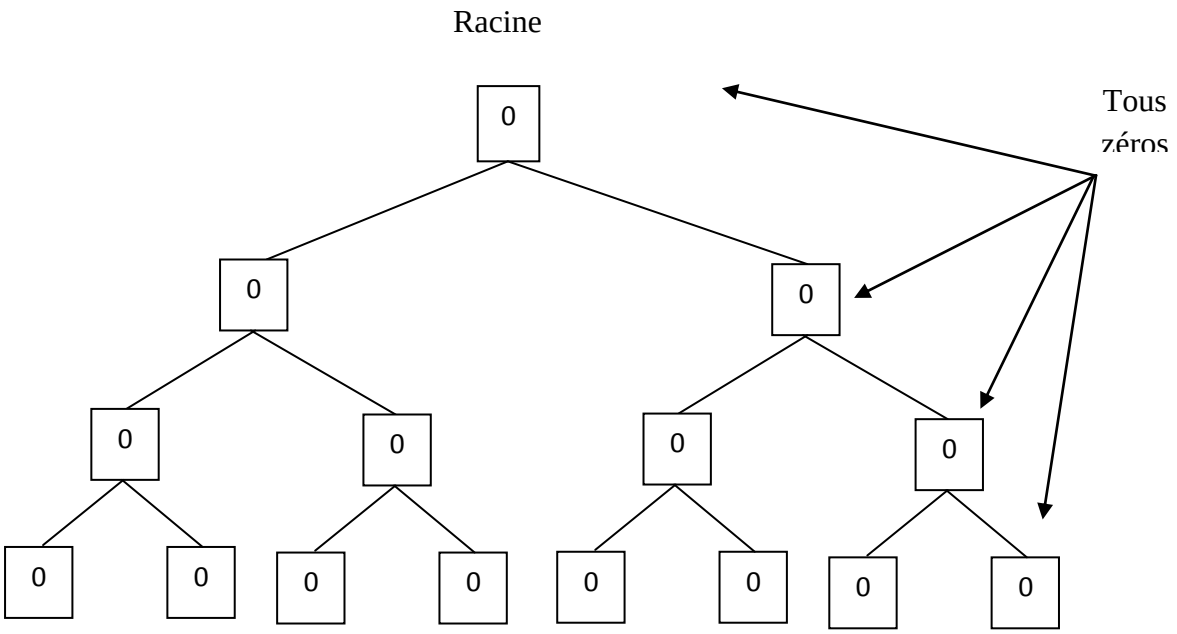


Fig.B1. arbre de zéros de degré 0.

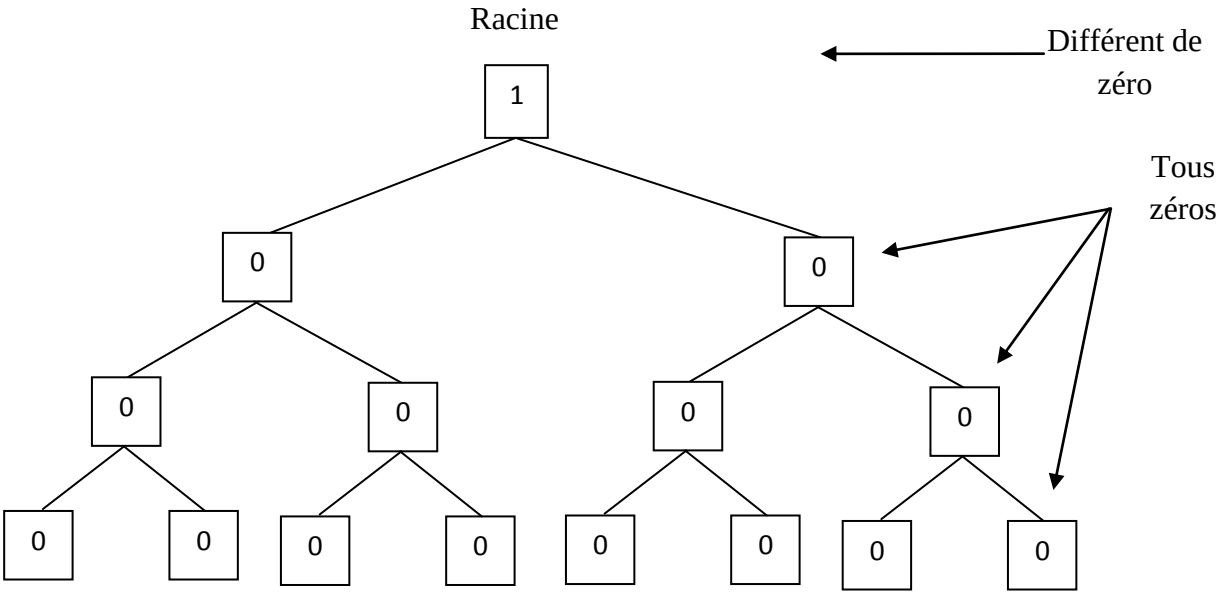


Fig.B2. arbre de zéros de degré 1.

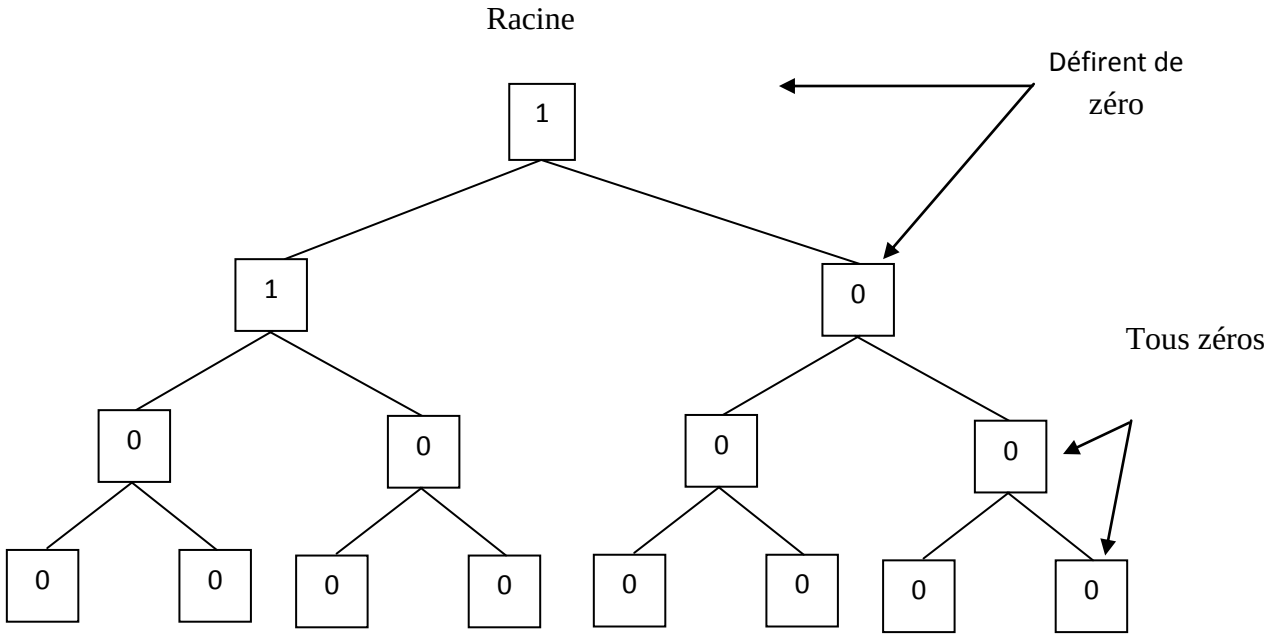


Fig.B3. arbre de zéros de degré 2.

Les méthodes basées sur les arbres de zéros présentent les avantages suivants :

- Corrélations inter-bandes bien exploitées.
- Représentation progressive de l'image.
- Codage avec perte vers codage sans perte.
- Faible complexité.
- Très bonnes performances de compression.

BIBLIOGRAPHIE

[1] : I. BOUKLIHACENE, « compression d'images médicales par Ondelettes de seconde génération » : Mémoire de Doctorat en Electronique biomédicale, Faculté de Technologie, Département de Génie Electrique et Electronique, UABB-Tlemcen,2014.

[2] :G.MERCIER, C.ROUX, G.MARTINEAU, "Technologies du multimédia", ENSTBretagne, Département ITI, France, 2003.

[3] : D. LINGRAND, "Introduction au traitement d'images", Edition Vuibert, Paris 2004.

[4] : X.WU, "Lossless compression of continuous-tone images via context selection, quantization and modeling", IEEE Transactions on Image Processing, vol. 6, 1997.

[5]: A. LEMPEL, J. ZIV, "Compression of two-dimensional data", IEEE transactions oninformation theory, vol. 32, no. 1, pages 2–8, January 1986.

[6]: Y. Q. SHI, H. SUN, "Image and video compression for multimedia engineering", 2nd Edition, CRC press, Taylor & Francis group, 2008.

[7]: K. SAYOOD, "Introduction to data compression", 3nd Edition, Morgan Kaufmann, 2006.

[8]: A.PENTLAND, "Fractal-Based Description of Natural Scenes", IEEE trans.Evolutionary Computing, vol. 6, n°. 6, pp.661–674, Nov. 1984.

[9]: S.YEA, S.CHO, W.A.PEARLMAN,"A wavelet –based two –stage near-lossless coder", Image Processing, IEEE Transactions on 15(11):3488-3500, Novembre 2006.

[10] : J. TAQUET, "Techniques avancées pour la compression d'images médicales", Thèse de doctorat en électronique, Université de Rennes1, 2011.

[11] : A. CZIHO, "Quantification vectorielle et compression d'image. Application à l'imageriemédicale", Thèse de doctorat, Université de Rennes1, Mai 1999.

[12]: S. MALLAT, "A wavelet tour of signal processing", Academic Press, San Diego,2nd Edition, 1999.

[13]: J.SHAPIRO, «Embedded Image Coding using Zerotree of Wavelet Coefficients", IEEE trans.Signal processing. Vol. 41, pp. 3445-3463, December 1993.

[14]: A. SAID, W. PEARLMAN, "A new, fast, and efficient image codec based on set partitioning in hierarchical trees", IEEE Transactions Circuits Syst. Video Technol., vol. 6, pp. 243-250, juin 96.

[15]: B.J. KIM, W. PEARLMAN, "An EmbeddedWavelet Video Coder Using Three-DimensionalSet Partitioning in Hierarchical Trees," IEEE Data Compression Conference, pp. 251-260, mars1997.

[16] : B. REMI, N. FRANÇOIS, « La compression JPEG », Rapport de Projet, Ecole nationale supérieured'ingénieurs de Caen et centre de recherche (ENSICAEN), Université de Caen Basse-Normandie, année 2005-2006.

[17] : N.SADOUN, « Compression d'images fixes par une méthode hybride », thèse de magistère en électronique, UMMTO, 2005.

[18] : S. LAKROUM, « Transmission progressive d'images fixes », mémoire d'ingénieur d'état en électronique, UMMTO, 2002.

[19]: I.DAUBECHIES, "Orthonormal Bases of Compactly Supported Wavelets",Comm.Pure on Applied Mathematics, vol.41, pp. 909-996, Nov.1988.

[20]: D.TAUBMAN, "Subband transforms and the DWT", Signal Processing 2,University of New South Wales Sydney Australia, 2002 (Support de cours).

[21]: S.MALLAT, "A theory for multiresolution signal decomposition: The wavelet representation", IEEE Trans. Pattern Analysis and Machine Intelligence, vol.11, n°.7, pp. 674-693, July 1989.

[22] :M. SAHIR,« Compression des images numériques par la technique des ondelettes », mémoire magister en électronique, Faculté de Technologie Département d'Electronique Université Ferhat Abbas-Sétif, 2011.

[23] :M. OUAHIOUNE,« compression d'images hyper spectrales par la transformée en ondelettes 3D »,mémoire magister en électronique, UMMTO, 2011.

[24] : A.OUAFI «Compression d'images avec pertes par codages imbriqués, *Proposition d'une optimisation de l'algorithme EZW* » thèse de doctorat en Electronique Faculte des Sciences et Technologie Département d'Electronique Université Mohamed Khider – Biskra.2009

Les Sites Web utilisés :

- http://www.map.toulouse.archi.fr/works/panoformation/imagenum/imagenum.htm.
- http://www.techniques-ingenieur.fr/base-documentaire.
- http://jj2000.epfl.ch.
- www.ee.nthu.edu.tw/~cwlin/courses/multimedia/notes/EBCOT.pdf] High performance scalable image compression with EBCOT, D.Taubma,2000
- http://d.xav.free.fr/ebcot/(Xavier Delaunay : Les trois passes de codage dans EBCOT expliquées sur un exemple détaillé).