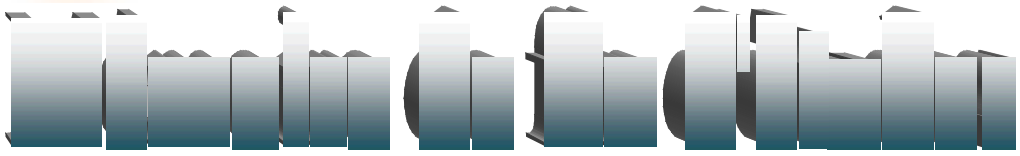
REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
Ministère de l'enseignement supérieur et de la recherche scientifique
Université Mouloud Mammeri de Tizi-Ouzou
Faculté de Génie Electrique et d'Informatique
Département d'Electronique



En vue de l'obtention du diplôme d'ingénieur d'état en électronique
Option : contrôle

Thème

Compression d'images fixes par classification de régions en associant les ondelettes et les fractales

Proposé et dirigé par :

M^me^ HEMDANI NAIMA

Réalisé par:

M^elle^ : NAIT AMARA NAIMA
M^elle^ : CHOUCHANE RADIA

Promotion: 2008

Remerciements

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

Toute notre reconnaissance et notre gratitude à M^r^ LAZRI MOURAD et M^elle^ CHILLALI KARIMA pour leurs aides précieux qu'ils nous ont apportés afin de mener à bien 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 aidé et encouragé pour l'aboutissement de ce travail, trouve ici l'expression de notre sincère reconnaissance.

Dédicaces

Je dédie ce modeste travail à:

- *Mes très chers parents*
- *Mes très chers frères:Ameziene et Arabe*
- *Ma très chère soeur Djidjiga son époux et ces enfants:Sabrina,yacine et salim*
- *Mes très chères soeurs: Saida et Kahina*
- *Mes très chers grands parents*
- *Mes très chers oncles et tentes et leurs familles*
- *Mes cousines et cousins*
- *Mes camarades et amies: kahina, Aldjia, Houria, Fadila,RADIA*
- *Toute la promotion2007/2008*

NAIMA

Sommaire

chapitre II : La théorie des ondelettes

ChapitreIII : Théorie des fractales

Chapitre v : La méthode adoptée

Chapitre V : tests et résultats

Introduction générale

Le développement de l'informatique a engendré des applications qui mettent en jeu la transmission et l'archivage de données de plus en plus volumineuses, ce qui justifie le recours à la compression.

Comprimer les données revient tous simplement à éliminer la redondance. Si l'on veut augmenter la vitesse de consultation des données ou minimiser l'espace de stockage des documents numérisés, surtout dans le cas des images, il est nécessaire de recourir à des systèmes de compression appropriés. Deux sortes de techniques permettent la compression des images : les méthodes réversibles, c'est-à-dire sans pertes, qui conduisent à des faibles taux de compression et celles appelées irréversibles, c'est-à-dire avec pertes, qui permettent de compresser fortement les images.[19]

Les méthodes de compression d'images les plus utilisées dans la littérature sont celles avec pertes (irréversible), citons la quantification scalaire ou vectorielle, le codage fractale, les codeurs basés sur les transformations.

Quant aux méthodes par transformations à fort taux de compression, les plus efficaces sont basées sur de puissant outils, tels que la transformée en cosinus discrète (la base de JPEG), la transformée en ondelettes (la base de JPEG2000).

Les méthodes fractales subit une forte compression, mais avec des pertes qui se traduisent par des distorsions plus ou moins perceptibles à l'œil nu lors de la décompression et un temps de calcul lors de l'étape de codage est assez important. [4]

Notre travail consiste alors à développer une méthode hybride basée sur le codage fractale dans le domaine transformé des ondelettes. Pour cela, nous avons organisé ce mémoire en cinq chapitres :

- Le premier chapitre porte sur des notions et généralités sur la compression d'images.
- Le second chapitre est consacré à la transformée en ondelettes et à la notion d'analyse multiresolution.
- Le troisième chapitre concerne la théorie des fractales, qui constitue un domaine impressionnant, parce qu'il est en relation directe avec la nature dans laquelle nous vivons. Ce domaine est caractérisé par des ressemblances typiques de ses

composantes d'une façon systématique, ceci peut être exploité dans la compression d'images.

- Au quatrième chapitre, nous avons exposé la méthode adoptée en résumant le principe du schéma de compression/décompression que nous avons élaboré.
- Le cinquième chapitre de ce mémoire représente les résultats obtenus par l'association de la transformée en ondelettes et les fractales.

Enfin, nous terminerons par une conclusion générale, en présentant une synthèse sur notre travail ainsi que les perspectives et les directions de recherches.

Introduction

La représentation des images fixes est un des éléments des applications multimédias, comme 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 (par exemple sur une disquette), et encore plus pour être transmise sur des réseaux bas ou moyen débit. La question qui se pose, comment peut on gagné en vitesse lors de la transmission, et en capacité de stockage ?

En réponse à cette question des méthodes de compressions d'images sont développées. Nous allons les détailler dans ce présent chapitre.

Une image, telle qu'elle est définie à l'échelle de l'observation visuelle (une représentation bidimensionnelle d'objet tridimensionnel de nature diverse) est inexploitable par les équipements audiovisuels ou informatiques, ce qui exige, donc ; sa numérisation.

- **Types d'images numériques [1,20]**

On distingue **deux types** d'images à la composition et au comportement différent :

Images matricielles et **les images vectorielles.**

- **Image matricielle (ou image bitmap)**

Elle est composée comme son nom l'indique d'une **matrice (tableau)** de points à plusieurs dimensions, chaque dimension représentant une dimension spatiale (hauteur, largeur, profondeur), temporelle (durée) ou autre (par exemple, un niveau de résolution).

- **Images 2D**

Dans le cas des images à deux dimensions (le plus courant), les points sont appelés **pixels**. Ce type d'images s'adapte bien à l'affichage sur écran informatique (lui aussi orienté pixel) ; il est en revanche peu adapté pour l'impression, car la résolution des écrans informatiques, généralement de 72 à 96 **ppp** (« points par pouce », en anglais *dots per inch* ou *dpi*) est bien inférieure à celle atteinte par les imprimantes, au moins 600 ppp aujourd'hui. L'image imprimée, si elle n'a pas une haute résolution, sera donc plus ou moins floue ou laissera apparaître des pixels carrés visibles.

Images 2D + t (vidéo), images 3D, images multi-résolution

- Lorsqu'une image possède une composante temporelle, on parle **d'animation.**
- Dans le cas des images à trois dimensions les points sont appelés des ***voxels.*** Ils représentent un **volume.**

Ces cas sont une généralisation du cas 2D, la dimension supplémentaire représentant respectivement le temps, une dimension spatiale ou une échelle de résolution.

Images vectorielles [20]

Le principe est de représenter les données de l'image par des formules **géométriques** qui vont pouvoir être décrites d'un point de vue **mathématique**. Cela signifie qu'au lieu de mémoriser une mosaïque de points élémentaires, on stocke la succession d'opérations conduisant au tracé. Par exemple, un dessin peut être mémorisé par l'ordinateur comme « *une droite tracée entre les points (x1, y1) et (x2, y2)* », puis « *un cercle tracé de centre (x3, y3) et de rayon 30 de couleur rouge* ». L'avantage de ce type d'image est la possibilité de l'agrandir indéfiniment sans perdre la qualité initiale, ainsi qu'un faible encombrement. L'usage de prédilection de ce type d'images concerne les schémas qu'il est possible de générer avec certains logiciels de **DAO** (Dessin Assisté par Ordinateur) comme **Auto CAD.** Ce type d'images est aussi utilisé pour les animations **Flash**, utilisées sur Internet pour la création de bannières publicitaires, l'introduction de **sites web**, voire des sites web complets. Étant donné que les moyens de visualisation d'images actuels comme les **moniteurs d'ordinateur** reposent essentiellement sur des images matricielles, les **descriptions vectorielles (Fichiers)** doivent préalablement être converties en **descriptions matricielles** avant d'être affichées comme images.

I.1. Définition de la compression : [11,15]

La compression consiste à réduire la taille physique de blocs d'informations. 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, statistique. On évalue la performance d'une compression effectuée par :

- Le taux de compression
- L'entropie

- La qualité de reconstitution de l'image (mesure de distorsion)
- La rapidité du codeur et du décodeur (codec)

I.1.a. Taux de compression :

Sachant que, le but d'une compression est de minimiser la quantité d'informations nécessaires à la représentation d'une image, on définit le rapport de compression Rc tel que :

$$Rc = \frac{\text{Nbre de bits de l'image originale}}{\text{Nbre de bits de l'image compressée}}$$

Par conséquent, on peut définir la quantité Tc appelée taux de compression par :

$$Tc = \left(1 - \frac{1}{Rc}\right) \times 100$$

L'objectif d'une compression d'image est donc d'avoir un plut grand taux de compression «Tc» possible.

I.1.b. L'entropie :

On définit l'entropie (en bit) d'un point particulier P d'une image par :

$$H(p) = \sum_{n_i=1}^{m} p(n_i).I(n_i)$$

Avec :

n_i : Les niveaux de gris que peut revêtir le point P,

m : le nombre total de n_i,

p (n_i) : la probabilité d'apparition du niveau de gris n_i,

I (n_i) : l'information propre du niveau de gris n_i qui est définit par :

$$I(n_i) = \log_2 \frac{1}{P(n_i)}$$

D'où l'entropie sera :

$$H(p) = \sum_{n_i=1}^{m} P(n_i).\log_2 \frac{1}{P(n_i)}$$

$$H(p) = -\sum_{n_i=1}^{m} P(n_i).\log_2 p(n_i)$$

Donc :

L'entropie caractérise la quantité d'informations que contient une image .Toutefois, une image, dont tous les pixels ont la même valeur contient très peu d'informations, son entropie est faible. Par contre, une image, dont tous les pixels ont une valeur aléatoire contient beaucoup d'informations et son entropie est forte.

I.1.c. Mesure de la distorsion :

Pour évaluer numériquement la quantité de l'image reconstruite, il est nécessaire de contrôler ses distorsions, donc de les mesurer. Ces mesures se traduisent par des critères objectifs qui se sont qu'une approche de la quantité d'une image .Ces critères sont:

- L'erreur quadratique moyenne (MSE) :

$$MSE = \frac{1}{N}\sum_{n_i=1}^{N} (\hat{n}_i - n_i)^2$$

Avec :

n_i: Le niveau de gris de l'ieme point de l'image originale,

$\hat{n}_i$: Le niveau de gris de l'ieme point de l'image transformée (reconstruite),

N : le nombre total de points constituant chacune des images.

- Le rapport signal sur bruit crête (PSNR) en dB :

$$PSNR = 10\log_{10} \frac{N_{dGmax}^{\ 2}}{MSE}$$

Avec :

N_{dGmax} : le niveau de gris maximum et l'exemple couramment utilisé en télévision numérique est N_{dGmax} =255 donc :

$$PSNR = 10 \log_{10} \frac{(255)^2}{MSE}$$

Etant donné que, l'image reconstruite doit s'apprécier visuellement, on peut l'améliorer en calculant l'image différence ''I_{DIFF}'' entre l'image originale et l'image reconstruite qui est définit par :

$$I_{Diff} = 2(n_i - \hat{n}_i) + 128$$

Ou le facteur 2 est adopté pour rehausser la dynamique de l'image, et 128 pour rendre l'erreur I_{DIFF} positive (pour des raisons de visualisation). Cette image différence devrait être uniforme pour une image parfaitement reconstruite .Lorsque la reconstitution de l'image est parfaite, on a :

$$\begin{cases} MSE \longrightarrow 0 \\ PSNR \longrightarrow \infty \end{cases}$$

Donc, pas de distorsion.

Ce qui peut être vérifié sur le graphe de la figure I- 1, qui lie la distorsion au nombre de bits nécessaires au codage de l'image et qui est appelé «la fonction débit /distorsion» :

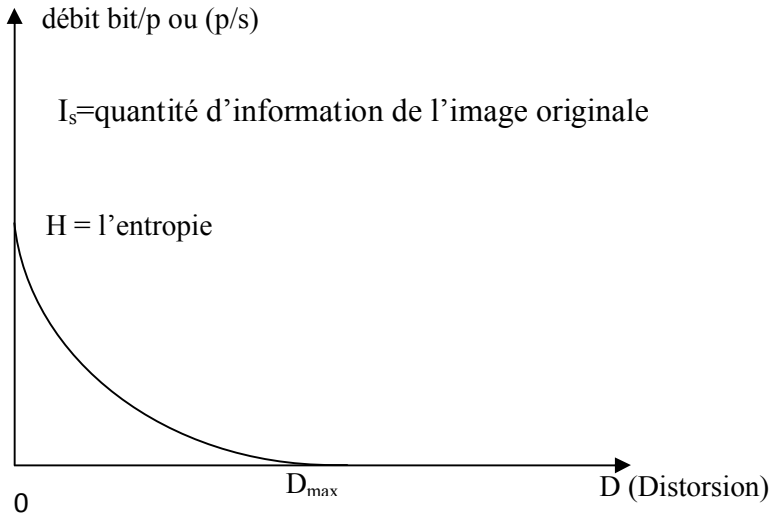


Figure I-1 : La fonction débit/distorsion

De la figure I-1, nous constatons que plus on réduit le nombre de bits par pixel, plus on introduit des distorsions ; donc D augmente. Par conséquent, on peut distinguer deux zones sur le graphe :

- La zone comprise entre I_s et H, c'est là où, D = 0 ; c'est-à-dire pas de distorsion,
- La zone comprise entre H et 0, c'est là où, D≠0 ; ce qui veut dire qu'il y a des distorsions.

I.2. Principe général de la compression d'images : [11,13]

Le schéma fonctionnel de la compression est représenté dans la figure ci –dessous :

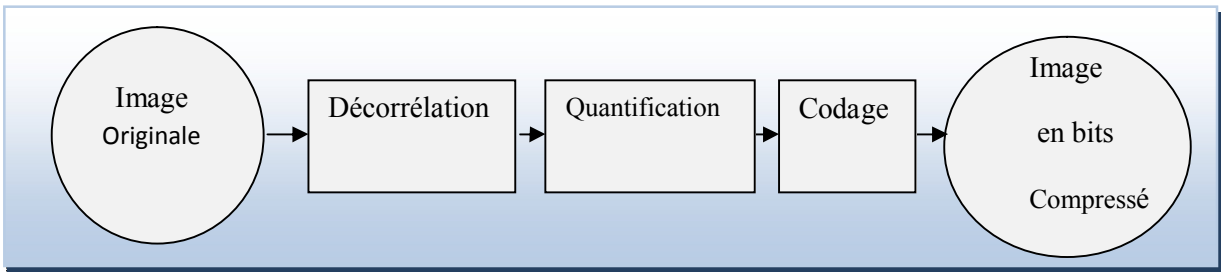


Figure I-2 : Schéma général d'un codeur source

I.2.a. Décorrélation :

La dépendance existante entre chaque pixel et ses voisins (la luminosité varie très peu d'un pixel à son voisin) traduit une corrélation très forte dans l'image. On essaie donc de tirer partie de cette corrélation, pour réduire le volume d'information, en effectuant une opération de décorrélation des pixels. Cette décorrélation consiste à transformer les pixels initiaux en un ensemble de coefficients moins corrélés, c'est une opération réversible.

I.2.b. Quantification :

La quantification des coefficients obtenus a pour but de réduire le nombre de bits nécessaires pour leurs représentations. Celle-ci représente une étape clé de la compression. Elle approxime chaque valeur d'un signal par un multiple entier d'une quantité q, appelée«quantum élémentaire» ou «pas de quantification».

La quantification peut être scalaire ou vectorielle. L'un des résultats fondamentaux des travaux de Shannon concernant la relation : débit/distorsion montrent que l'on obtient de meilleures performances en utilisant la quantification vectorielle.

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

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

Plusieurs types de codages seront détaillés ci-après.

I.3. Classification des méthodes de compressions:[15]

La compression d'images fait appel à une variété d'algorithmes du 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, définis dans la figure I-3

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

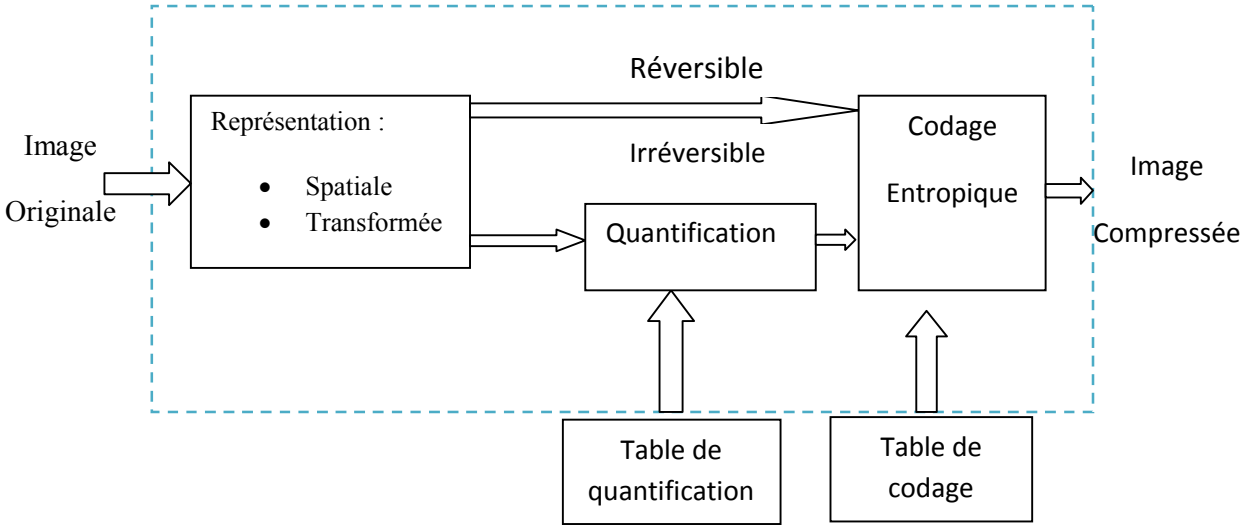


Figure I-3 : Structure générale d'un schéma de compression d'images

I.3.1. Les méthodes sans perte (réversibles) [11]

Ces méthodes sont dites à codage exact qui présente l'avantage d'une reconstruction exacte de l'image, leurs inconvénients sont un taux de compression insuffisant et la dépendance de celui-ci par rapport au contenu de l'information. Les méthodes sans perte

peuvent s'appliquer dans le domaine spatial ou plus difficilement dans le domaine des fréquences. Parmi ces méthodes, on distingue :

I.3.1.a. 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

Svmbole	Fréquences	Code	
A	15	0 0	Deuxième division
B	7	0 1	Première division
C	6	1 0	Troisième division
D	6	1 1 0	Quatrième division
E	5	1 1 1	

Figure I-4 : codage de Shannon-fano

On voit bien que les trois symboles de fréquences les plus élevées ont eu un code sur deux bits, et les deux symboles de fréquences inférieurs ont un code de trois bits.

Bien que le codage de Shannon-fano soit un grand pas en avant, il a été rapidement remplacé par un système de codage encore plus efficace : le codage de Huffman.

I.3.1.b. Le codage de Huffman

La méthode développée par D.Huffman en 1952, traite l'extension des codes optimaux, elle permet la détermination d'un VLC (code à longueur variable) préfixé. La méthode de compression Huffman consiste à construire un arbre qui va nous permettre de donner un code pour chaque symbole en fonction de sa fréquence.

Le codage Huffman peut être appliqué à une source ayant un alphabet quelconque. Les étapes de son algorithme sont :

1. Ordonner par ordre décroissant les probabilités des symboles de la source S. La liste des symboles est prise comme feuilles de l'arbre à construire.
2. Générer un nœud intermédiaire à partir des deux feuilles ayant les probabilités les plus petites, auquel, on affecte la probabilité résultante.
3. Reconstruire une nouvelle liste à partir des probabilités restantes et retourner à l'étape 2 jusqu'à ce que la liste ne contienne plus qu'un seul nœud pour deux probabilités (racine de l'arbre).
4. Coder avec retour arrière depuis le dernier groupe en ajoutant un « 0 » ou un « 1 » pour différencier les symboles préalablement regroupés.

Donc : Le mot binaire d'un niveau de gris, ou encore d'une feuille de l'arbre, s'obtient simplement en écrivant de gauche vers la droite les bits rencontrés en parcourant les branches qui permettent de descendre de la racine vers la feuille considérée. Ainsi, on constitue une table de codes composée de mots codes de différentes longueurs qui sera transmise avec les symboles codés afin de l'utiliser pour le décodage.

- **Exemple de codage :**

Soit la source S de l'exemple, de la figure I-5, ayan un alphabet de 5 symbole dont les probabilités sont respectivement : $p_i = \{0.39, 0.18, 0.15, 0.15, 0.13\}$; i=0….4

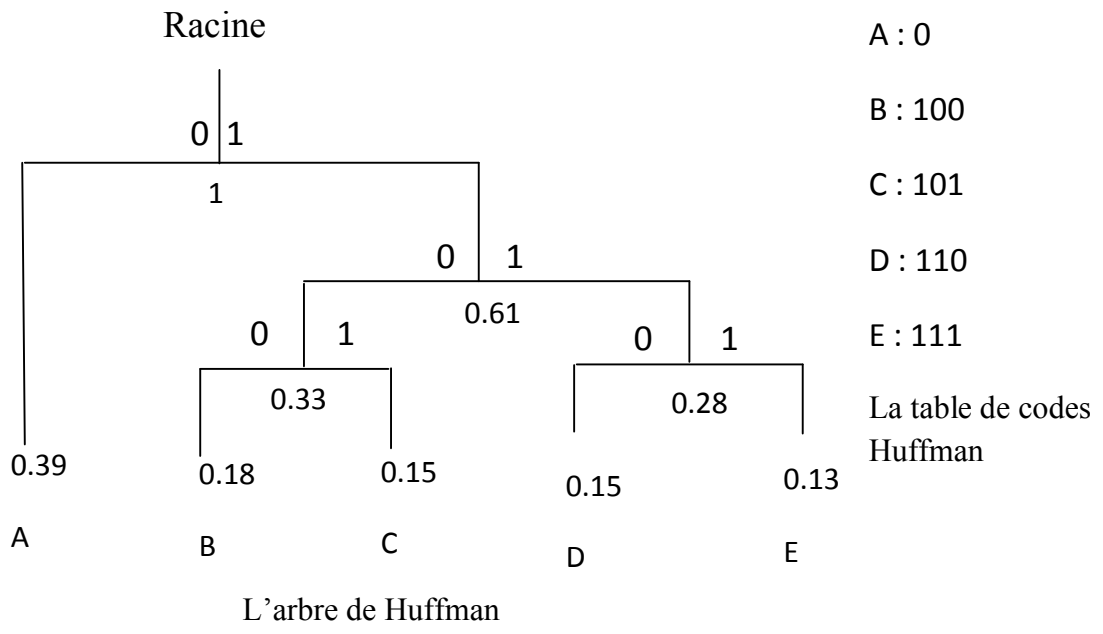


Figure I-5 : Exemple d'un codage de Huffman

Cette méthode consiste à diminuer au maximum le nombre de bits utilisés pour coder une information. Son algorithme se base sur l'utilisation de la fréquence d'apparition d'un fragment pour le coder, et plus un fragment est fréquent, plus on utilisera moins de bits pour le coder.

I.3.1.c. Codage arithmétique [14]

Contrairement aux algorithmes de Huffman et Shannon-fano qui associent à des symboles des codes binaires dont la taille dépend de leur distribution; le code arithmétique produit un code pour une séquence de symbole toute entière.

La procédure du codage arithmétique est la suivante :

1. Calculer la probabilité associée à chaque symbole dans la chaine à coder.

2. 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 décodage.

3. Initialiser la limite inférieure de l'intervalle de travail à la valeur 0 et la limite supérieur à la valeur 1.

4. Tant qu'il y a un symbole à coder dans la chaine :

- Sa largeur= limite supérieure – limite inférieure.

❖ Sa limite inférieure = limite inférieure +largeur * (limite basse du sous intervalle du symbole).

❖ Sa limite supérieur = limite inférieure + largeur *(limite haute du sous intervalle de symbole)

5. La limite inférieure code la chaine de manière unique.

Contrairement à Huffman, il n'est pas obligatoire que chaque code ait un nombre entier de bits. Par exemple, un symbole, de probabilité 0.9 a pour entropie 0.14 mais Huffman affectera probablement un code de 1 bit (ou plus), et la séquence codée aura un nombre de bits plus long qu'en théorie. Le codeur arithmétique est plus performant que le codeur de Huffman, mais il est plus difficile à implémenter.

I-3-1-d- Codage en RLE (Run Lenght Encoding) [15]

Une plage représente une suite de symbole ayant la même valeur ; elle est caractérisée par son adresse de départ, sa longueur et sa valeur.

Cette compression et beaucoup moins efficace que d'autre méthodes cependant son avantage est d'être facile à implémenter.

Elle est utilisée par les fax et dans les méthodes par transformation comme JPEG.

Exemple :

Soit une ligne quelconque d'une matrice présentant une image numérique :

100 100 100 101 102 102

Plage n°1 :100 100 100 (3,100) ;

Plage n°2 :101 (1,101) ;

Plage n°3 : 102 102 (2,102) ;

I.3.1.e. La compression Lempel Ziv Welch (LZW) [10,11]

Cet algorithme est un des plus répondu. On les trouve par exemple dans les formats GIF et TIFF. Le premier élément de la famille des compresseurs LZ a été crée par Abraham Lempel et Jacob Ziv. En 1977 Ce compresseur nommé LZ77 est utilisé dans les programmes d'archivage de données comme PKZIP, ARG ou bien LHA. Il est spécialisé

dans les données textuelles alors que LZ78 est plus efficace pour les données binaires comme les images.

Une amélioration des algorithmes précédents a été introduite par Terry Welch. Le résultat fut l'algorithme LZW. Le codeur LZW et capable de travailler avec n'importe qu'elle type de données. Il est rapide en compression et décompression et ne nécessite pas d'opération à virgule flottante. De par le fait qu'il encode au niveau de bit et non au niveau de l'octet, il ne soucie pas du processeur et de la manière dont il code les informations.

La méthode LZW consiste à remplacer par quelques bits, un mot, une phrase ou même un paragraphe entier. Ces bits sont constitués d'une manière unique à laide d'un dictionnaire créé au fur et à mesure des besoins. Bien entendu, il n'est pas question de transférer ou de sauvegarder le dictionnaire. Il doit être crée dynamiquement.

Le flot d'information à compresser est découpé en chaine d'octets. Chaque chaine est comparée au dictionnaire, si elle n'est pas présente, elle est stockée. Elle est ensuite écrite dans le flot de sortie compressé. Quand une chaine, déjà rencontrée, apparait dans le flot, elle est codée et transmise si elle a une longueur inférieure au plus grand mot du dictionnaire. Pour le décodage, le dictionnaire est reconstruit dans le sens inverse. Il n'est pas nécessaire ainsi de transmettre le dictionnaire. Bien souvent, pour la compression et la décompression, le dictionnaire est initialisé avec les 256 valeurs de la table ASCII. Ainsi tous les codeurs et les décodeurs LZW initialisent leur dictionnaire de la même manière.

Supposant que nous disposions d'un dictionnaire initial composé de 256 caractères correspondant à la table ASCII, le rang des caractères est compris entre 0 et 256 ; nous ajoutons à ces 256 codes deux autres codes possédant les rangs 256 et 257 et servants de codes de contrôle.

Le principe est fondé sur le fait qu'une séquence de caractère peut apparaitre plusieurs fois dans un fichier ou dans un flux de transmission.

La méthode LZW exploite cette fréquence d'apparition en stockant dans un dictionnaire les séquences de caractères identifiables par un rang de classement qui peut être aléatoire. Chaque fois qu'une séquence répertoriée, on la transmet sous forme détournée par l'intermédiaire de son rang de classement. La compression consiste non plus à envoyer une suite de caractères, mais remplacer des séquences par un indice référencé dans un dictionnaire. L'intérêt de cette méthode réside dans le fait qu'un indice est transféré sur 12

bits avec un dictionnaire contenant 4096 emplacements (séquences différentes) sinon le transfert de la moindre petite séquence de 2 caractères se ferait sur 16 bits ; on gagne 4 bits à chaque séquence connue. Par contre, lors d'une suite inconnue, on perd 4 bits car on transmet un octet.

I-3-2 - Codage de compression avec pertes [10, 12,13]

Afin d'atteindre des taux de compression assez importants avec les images complexes, une compression irréversible est nécessaire. Elle fournit des compromis entre la qualité de l'image et le degré de compression. Sachant que, ces méthodes engendrent des pertes d'information, même si, elles sont indécelable à l'œil nu, on peut leurs associer les techniques de rehaussement pour améliorer l'apparence des images de compression.

I-3-2-a- Codage par dictionnaire [15]

Les méthodes de compression étudiées jusqu'ici utilisent un modèle statistique pour coder des symboles uniques. Elles effectuent la compression en codant les symboles en chaines de bits qui utilise moins de bits que les symboles originaux. Les algorithmes de compression à base de dictionnaires utilisent une méthode complètement différente pour compresser les données. Cette famille d'algorithmes code des chaines de symboles de longueur variable comme des prototypes uniques. Les prototypes forment un index dans un dictionnaire.

i- Quantification scalaire [15]

La quantification scalaire (QS) est une méthode irréversible très largement employée en compression pour plusieurs raisons, parmi lesquelles on peut citer :

- L'intérêt général de la quantification, dont une application particulière est la conversion analogique numérique.
- La simplicité qui permet des traitements temps réel dans les applications difficiles : radar, télévision…
- La QS est généralement la base d'autres méthodes.

En général, un quantificateur scalaire est une fonction en escalier, l'intervalle de toutes les valeurs possibles de symboles d'entrée est divisé en « n » intervalles adjacents, appelés intervalles de décision. Chaque symbole appartient à l'un de ces intervalles, et sa valeur codée correspond au numéro de son intervalle. Tous les symboles d'entrée appartenant à

un même intervalle sont codés et reconstruits avec la même valeur. La figure I-6 interprète ce principe :

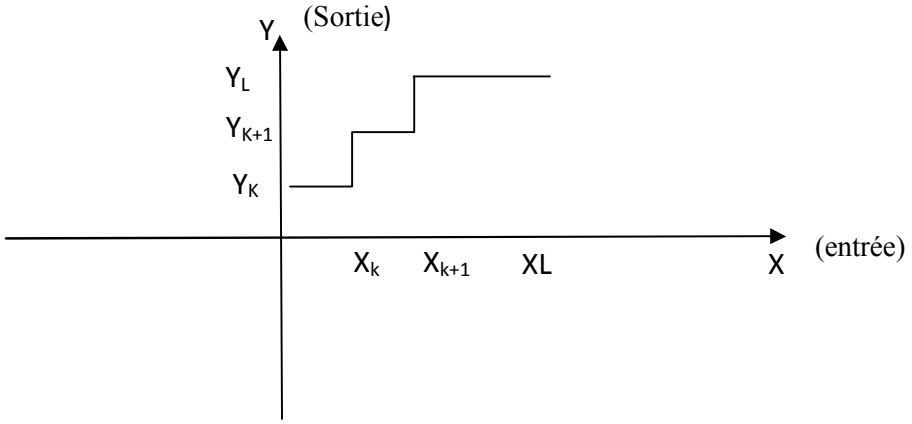


Figure I-6 : principe de la quantification scalaire.

Régions de décision (l'entrée) :(X_k, k=1…L+1).

Seuils de décision (sortie) : (y_j, j=1…L).

Si x £ (x_k, x_{k+1})→QS(x)=Y=y_L

La quantification introduit inéluctablement une distorsion.

ii- Quantification vectorielle (QV) [15]

Il a été montré qu'il est possible d'améliorer la compression de données en codant des vecteurs plutôt que des scalaires. La QV découle de ce principe, elle représente une généralisation de la QS.

La QV peut être vue comme une application Q associant à chaque vecteur d'entrée "V" de dimension K un vecteur "V'" tel que : V' =Q(V) de même dimension appartenant à un ensemble fini Y appelé dictionnaire de taille N, avec :

$$Y = \{V'_j , j=1…….N\}$$

La quantification vectorielle se décompose en deux applications : un codeur et un décodeur.

- **Codeur :**

Le rôle de celui-ci consiste, pour tout vecteur V_i du signal (donc de l'image) d'entrée, à rechercher dans le dictionnaire Y le code vecteur V_I' le plus proche du vecteur V. c'est uniquement l'adresse du code vecteur V_J' ainsi sélectionnée qui sera transmise ou stockée. C'est à ce niveau que s'effectue la compression.

- **Décodeur :**

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

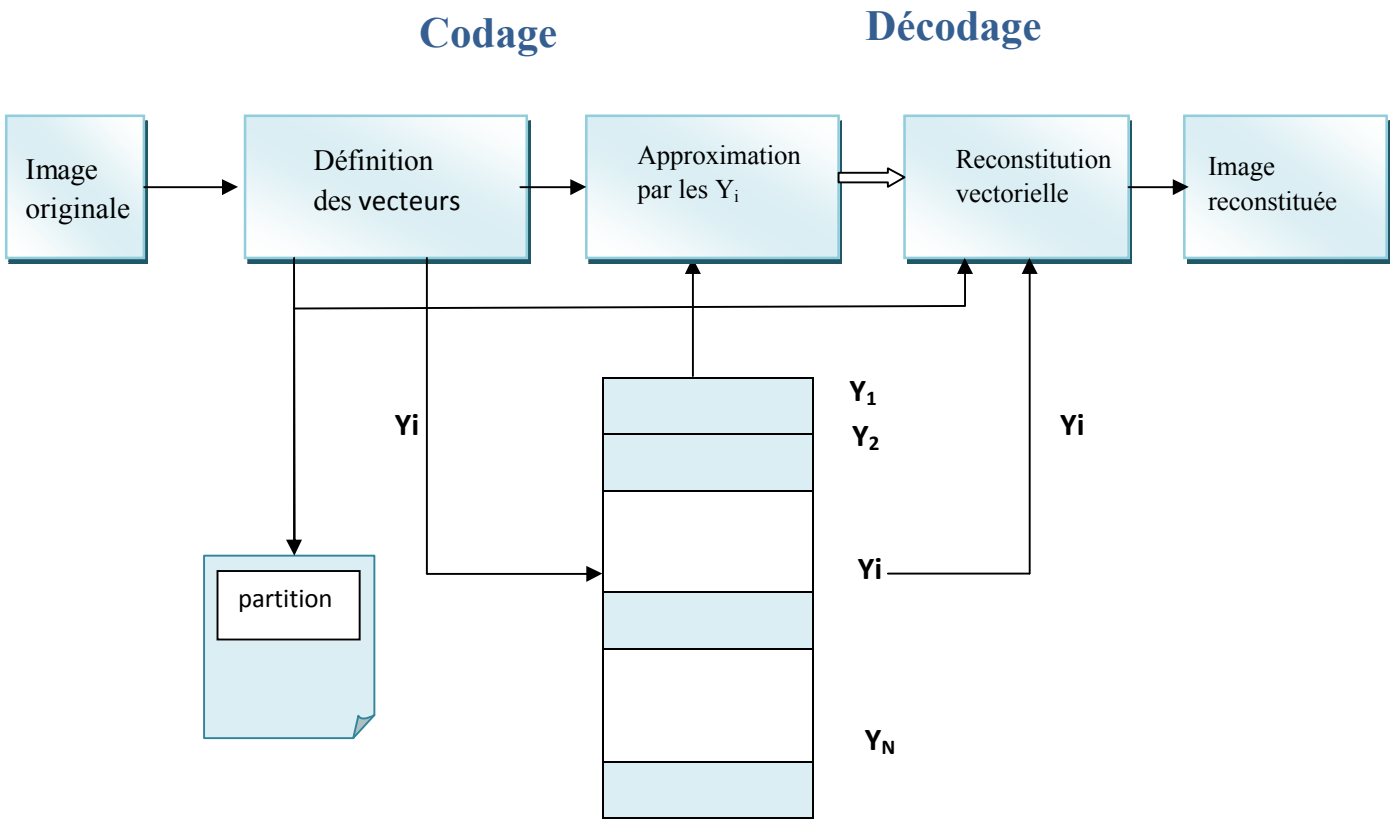


Figure I.7 : Synoptique général du codage d'image par quantification vectorielle

I.3.2.2. Codage par transformation [14]

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 :

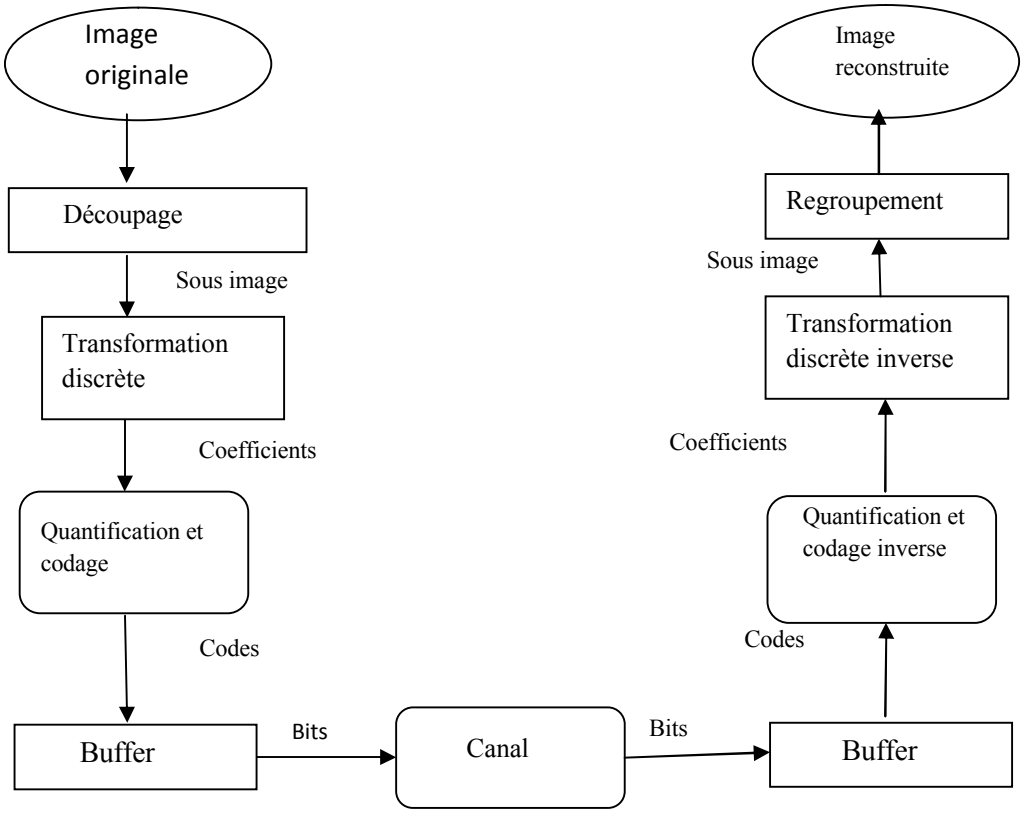


Figure I.8: principe d'un système de transmission d'un codeur par transformation

i- Transformation en cosinus discrète DCT [14]

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 formule générale de la DCT et celle de son inverse sont données par les relations suivantes :

$$F(u,v) = \frac{2}{\sqrt{N*M}} C(u)\cdot C(v) \sum_{n=0}^{N-1}\sum_{m=0}^{M-1} X(n,m) \cos\left[\frac{(2n+1)u\pi}{2N}\right].\cos\left[\frac{(2m+1)v\pi}{2M}\right]$$

Avec :

n,u=0 ,1……..N-1

m,v=0,1…….M-1

Et pour la transformation inverse :

$$X(n,m) = \frac{2}{\sqrt{N*M}} \sum_{u=0}^{N-1}\sum_{v=0}^{M-1} C(u).C(v).F(u,v) \cos\left[\frac{(2n+1)u\pi}{2N}\right]\cos\left[\frac{(2m+1)v\pi}{2M}\right]$$

Avec :

n,m : les coordonnées dans le domaine spatial.

u,v : les coordonnées dans le domaine fréquentiel.

$$C(u), C(v) = \begin{cases} \frac{1}{\sqrt{2}} & \text{Si } u \text{ ou } v = 0 \\ 1 & \text{Si } u \text{ et } v \neq 0 \end{cases}$$

Cette méthode est utilisée dans la norme JPEG qui une abréviation de Joint Photographique Expert Group. Apparue en 1989, 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.

Les techniques défiées 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-9 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 .10**:**

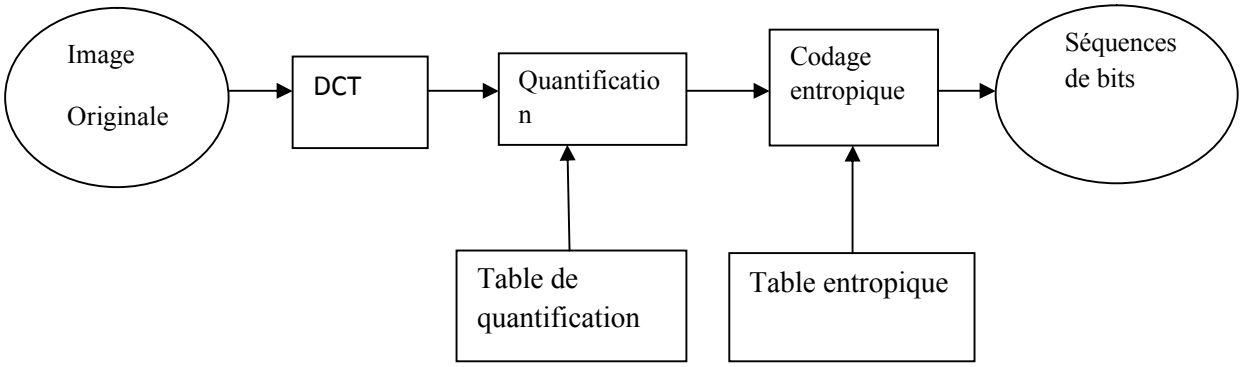


Figure I-9: principe de compression JPEG avec perte

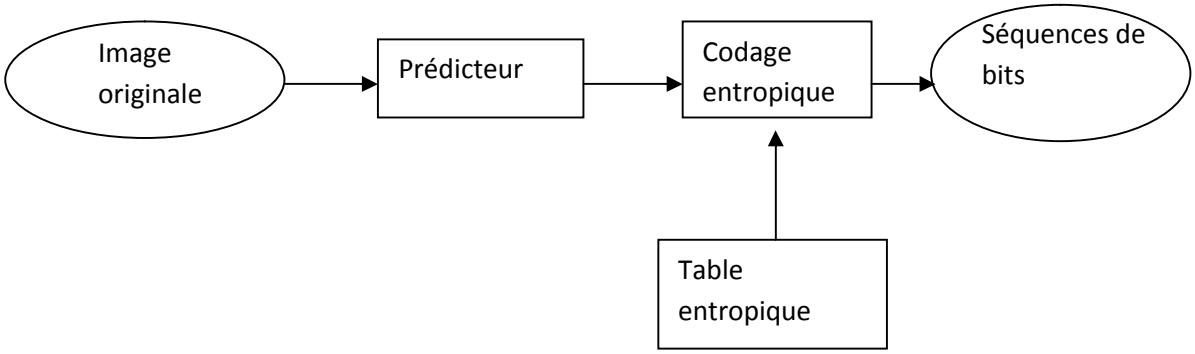


Figure I-10: Principe de compression JPEG sans perte

ii- Transformation en ondelettes discrète TOD [10,14]

La transformation en ondelettes discrète (TOD) bidimensionnelle repose sur la notion d'analyse multirésolution d'une image. Celle-ci est décomposée en un ensemble de sous bandes représentant l'information portée par l'image source à différent niveau de résolution : l'image d'approximation étant une version réduite et lissée de l'image initiale tandis que les images de détails « horizontaux, verticaux, diagonaux » contiennent uniquement des informations relative à la texture locale et aux contours des régions de l'image, à une résolution donnée et selon une direction donnée. L'algorithme de

MALLAT permet d'obtenir ces sous-images par application récursive, d'abord sur l'image source puis sur l'image d'approximation obtenue à chaque niveau, d'un banc de filtre passe-bas/passe-haut appliqué successivement selon les lignes et les colonnes de l'image à transformer.

Les ondelettes appliquent une méthode d'approche globale à l'image, ceci permet d'éviter le phénomène de mosaïque propre à la technologie JPEG (approche par blocs) dès que le taux de compression devient trop élevé. Nous verrons plus en détail les fondements de cette méthode dans le prochain chapitre.

Cette méthode est utilisé dans la norme JPEG 2000, elle utilise un procédé de décorrélation basé sur la transformée en ondelette discrète (TOD) et non plus la transformation en cosinus discrète (TCD) avec codage spécifique des sous image. Elle fournit également une structure d'organisation des données compressées très flexible. Les stratégies adoptées pour améliorer la qualité aboutissent à des algorithmes de compression de forte complexité. Du fait de la variété des techniques mises en place pour répondre aux différents profils, JPEG2000 s'apparente plus à une«boite à outils»qu'à un schéma de codage unique.

La première version définitive du standard a pris forme en décembre 2000. Ce nouveau standard a pour objectif d'offrir de nouvelles fonctionnalités permettant de rependre à une demande croissante, à savoir :

Obtenir des performances de compression supérieures à son prédécesseur JPEG notamment pour des débits très faibles.

Permettre d'organiser le fichier compressé de plusieurs manières, notamment en fonction de la résolution désirée ou de la qualité de reconstruction.

- Avoir un mode de compression sans perte performant.

- Fournir la possibilité de coder des parties d'une image avec une qualité supérieure à d'autres parties

- **Chaines de codage et décodage JPEG2000 [13,14]**

Le codage et décodage d'une image au format JPEG2000 s'effectuent en quatre étapes principales : Les trois étapes classiques en compression d'image (Transformation,

quantification, codage) plus une étape de pré-traitem.ent de l'image qui a pour but de rendre l'opération de codage plus efficace. Ces différentes étapes sont illustrées sur la figure I.11.

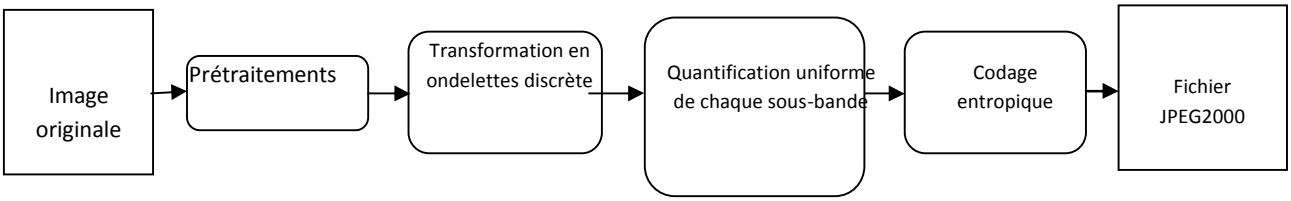


Figure I.11. Diagramme de chaine de codage de l'algorithme JPEG2000

I.3.2.3. Les transformations fractales [13,14]

Dans les techniques précédemment citées, l'information pour chaque pixel est mémorisée, par contre dans cette technique, ce sont des instructions ou des formules qui seront mémorisées pour la création de l'image. On démontre mathématiquement que pour chaque image, il existe une formule de calcul. Sa résolution est indépendante de la compression, on pourra avoir une résolution supérieure à 100. A l'inverse de la quantification vectorielle, la compression fractale est asymétrique, elle prend du temps pour la compression mais la décompression est rapide, elle consiste à lire en une simple lecture les formules mathématiques et puis à reconstituer l'image. La partie la plus difficile est de générer les formules qui représentent correctement l'image. La compression fractale, considère l'image composée de plusieurs petites images. Cette technique cherche à trouver les relations reliant les petites images composantes de l'image initiale puis les représenter par des formules mathématiques. Actuellement, la qualité des images reconstruites suite à une compression fractale est nettement inférieure à celle obtenue avec la norme JPEG. Mais lorsque les taux de compression sont supérieurs à 50, les algorithmes fractals permettent de maintenir un niveau d'image correct, alors que les JPEG sont illisibles. De plus la compression fractale permet une reconstitution à toutes les tailles.

I.3.2.4. Les méthodes hybrides [15]

Les méthodes hybrides de compression combinent entre les méthodes de domaine spatial et celle du domaine transformé, ainsi le codeur hybride regroupe les avantages des deux techniques qui le composent.

Conclusion :

Nous avons vu dans ce chapitre la cause de compression et les différentes solutions avec perte d'informations non visibles a l'œil nu telle que les méthodes par dictionnaires et par transformations, ou sans perte d'informations telle que Sannon-fano, Huffman, Arithmétique, RLE et LZW. Toutes ses méthodes exploitent la redondance de l'information de l'image. Nous avons vu en particulier les normes de compression JPEG et JPEG2000, cette dernière norme a intégré des techniques plus performantes avec une nouvelle alternative basée sur la transformée en ondelettes.

Introduction

Plusieurs transformées sont utilisées pour la compression d'informations, elles mettent en évidence l'intérêt de projeter le signal représentant cette information sur une base de fonction orthogonale. Parmi les transformées les plus utilisées on trouve la transformée en cosinus (DCT) et en sinus (DST), la transformée de Karhunen-Loeve et la transformée de Fourier qui permettent une analyse spectrale des signaux donc bien localisées en fréquence, mais complètement délocalisées en temps et elles n'admettent pas la non stationnarité.

A fin d'améliorer la localisation en temps et en fréquence, tout en préservant du mieux la localisation en fréquence, Gabor à utiliser une approche améliorée dite transformée de Fourier à fenêtre glissante (**TFFG**). Cette fenêtre est de taille fixe, elle ne convient pas à la représentation des signaux ayant des composantes de taille différentes d'elle. Alors peut-on avoir une fenêtre de taille variable ?

Morlet a proposé en 1981, une transformée où la taille de la fenêtre est variable, ceci grâce à un paramètre d'échelle d'où la naissance de la transformée en ondelettes.

II.1. Le principe des ondelettes [3,5]

L'idée de base est l'analyse multi-résolution d'un signal cela équivaut à son analyse à plusieurs distances successives.

Les ondelettes sont des fonctions qui sont simultanément caractérisées par une fréquence et localisées dans le temps (ou dans l'espace).

Plusieurs types de ces fonctions ont été utilisés (Meyer, Spline, Daubechies, Morlet…) on constitue pour chacun des types une famille autour d'une « ondelette mére » (Mother wevelet) donnant naissance à des filles (daughter wavelets) obtenues par contraction.

II.2. La définition de la Transformée en Ondelettes [6,14,9]

La théorie des ondelettes est récente, elle est née au milieu des années 80. Et pourtant, elle est déjà utilisée dans de nombreux domaines qu'ils soient théoriques ou pratiques (analyse harmonique, vision par ordinateur, traitement du signal, compression d'images, analyse de turbulences, etc...). Son succès est dû à son adaptativité aux données et à sa facilité d'implémentation.

Les ondelettes sont des fonctions élémentaires, générés à partir d'une ondelette mère, par dilatation en fréquence d'un facteur d'échelle **a** et translation dans le temps de **b** .Elle permet d'analyser un signal, de le manipuler puis le synthétiser.

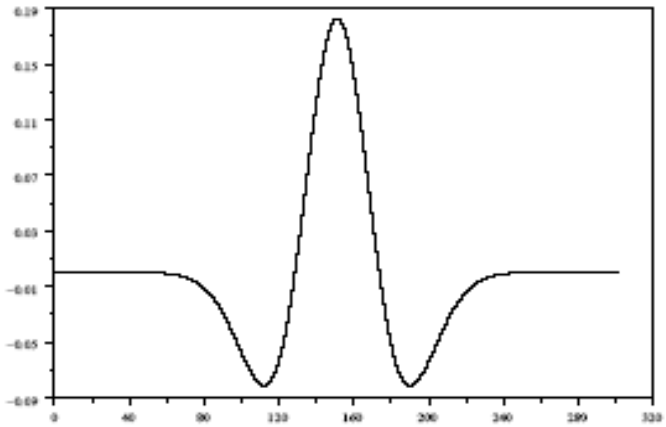


Figure-II-1 Ondelette mère

L'ondelette mère peut engendrer une famille $\psi_{a,b}(t)$ (a>0 et b réel) par dilatation

$$\psi_{a,b}(t) = \frac{1}{\sqrt{a}}\psi\left[\frac{t-b}{a}\right] \qquad (2\text{-}1)$$

$\frac{1}{\sqrt{a}}$ Le facteur de normalisation ; peut être aussi pris à 1 ou $\frac{1}{a}$.

Les paramètres a et b sont les deux arguments de la TO.

❖ Le fait que la transformée utilise des fonctions bien localisées dans le plan temps-fréquence, lui donne beaucoup d'avantages.

La résolution en fréquence de la transformée dépend du facteur de dilatation « a » par le principe d'Heisenberg, on peut donc choisir arbitrairement celle-ci suivant ce que l'on désire analyser.

Pour des signaux physiques présentant des variations très rapides, des discontinuités, l'analyse en ondelettes est adaptée car l'ondelette permet de détecter ses singularités et analyser celles-ci. Cette particularité rend l'analyse en ondelettes complémentaire à l'analyse de Fourier.

- **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 est la s le départ d'une ondelette

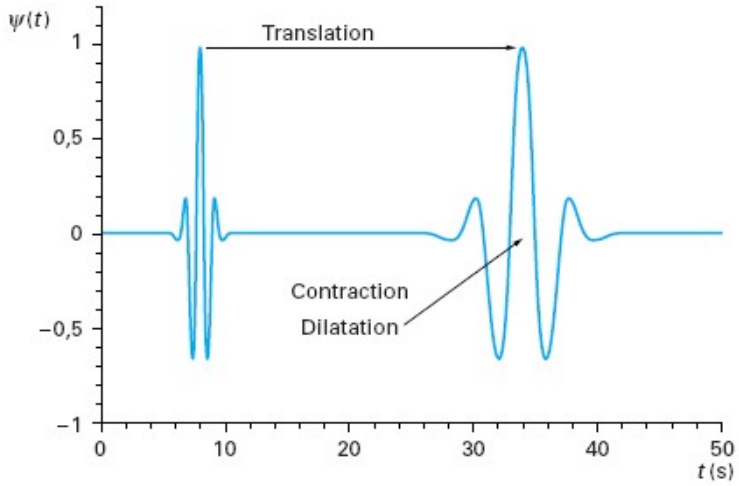


Figure-II-2 dilatation et translation

II.3. Propriétés des ondelettes [9]

- **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.

- **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 crées 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.

- **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. Condition d'admissibilité [12]

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 :

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

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

$\hat{\Psi}(w)$ 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 :

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

II.5. Définition de la transformée en ondelettes continues (TOC) [9,12]

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 détaille 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 :

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

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

$\overline{\psi}$ 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.

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

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

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

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 :

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

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 [9,12]**

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.6. La transformée en ondelettes discrètes [9,12]

En pratique, on a plus souvent affaire a 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 orthonormales 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 :

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

où

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

$\langle f, \Psi_{j,k} \rangle$: 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.7. Analyse multirésolution (AMR) [12]

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 cote 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 multiéchelle 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.7.1. Définitions [12]

Définition II.1 [12]

Une analyse multirésolution de $L^2(R^n)$ (espace des fonctions carrée sommables) est une suite croissante de sous espaces vectoriels fermés $\{V_j\}_{j\in Z}$ de $L^2(R^n)$ (2.15) ayant les propriétés suivantes:

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

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

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

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 :

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

et

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

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

Définition II.2. [12]

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

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

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

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

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.7.2. La fonction d'échelle [12]

Théorème II.1. [12]

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

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

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

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

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 II.2. [12]

Soit $\{V_j\}_{j \in Z}$ une analyse multirésolution 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

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

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 multirésolution 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:

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

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.7.3. Les fonctions ondelettes [12]

Théorème II.3. [12]

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

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

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 :

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

où

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

L'ensemble des définitions et théorèmes sus-cités, 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.

i- Filtres associé aux ondelettes [11]

Dans la pratique, les signaux qu'on manipule sont des signaux échantillonnés. Le traitement spatio-fréquentiel de ces signaux est effectué à laide de filtres numériques qui engendre une analyse multirésolution sous certaines conditions (voir AMR).

Les bancs de filtres associés aux fonctions d'échelle et d'ondelettes sont donnés par le schéma ci-dessous :

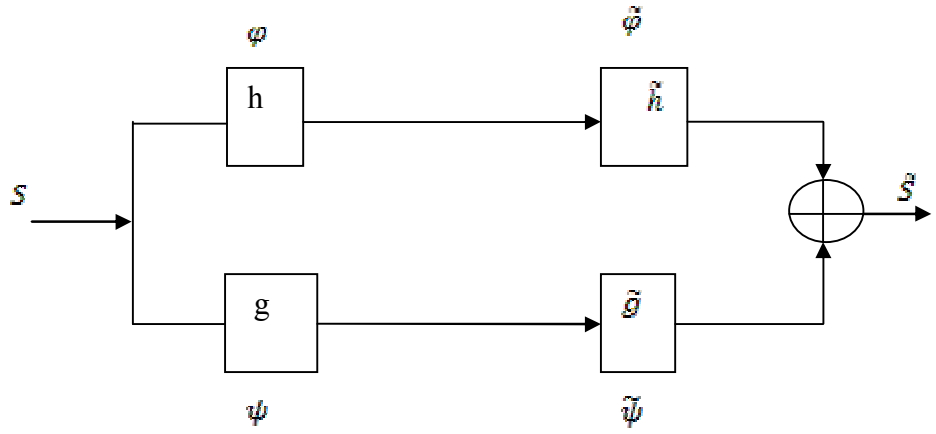


Figure II-3 : bancs de filtres associés aux fonctions d'ondelettes et échelles

h et g représentent respectivement les filtres passe bas et passe haut à l'étape d'analyse, $\tilde{h}$ et $\tilde{g}$ constituent les filtres passe bas et passe haut à l'étape de synthèse.

Dans le cas de l'analyse multirésolution orthogonale on associe une même base d'ondelette ψ aux filtres d'analyse et de reconstruction. Ainsi, une même fonction d'échelle ϕ est associée aux filtres d'analyse et de synthèse. par contre, dans le cas

biorthogonal, les bases d'ondelettes et fonction d'échelles d'analyse et de synthèse qui sont respectivement Ψet $\tilde{\Psi}$, φ et $\tilde{\varphi}$ ne seront pas les mêmes dans les deux étapes d'analyse et de synthèse.

Le signal S issu à l'entrée du banc de filtre est décomposé par filtrage avec décimation, dans ce cas, en deux sous-bandes ; une sous-bande basse fréquence et une sous-bande haute fréquence. La reconstruction de s est obtenue en sommant les deux signaux filtrés par $\tilde{h}$ et $\tilde{g}$ après avoir effectué l'opération d'interpolation.

II.8. Algorithme de S.Mallat [10,14]

Afin de construire un algorithme d'analyse et de synthèse pour les images bidimensionnelles, S.Mallat a fait appel à l'analyse multirésolution. Cet algorithme décrit une transformation en ondelette discrète, il est analogue à la pyramide laplacienne introduite par burt et Adelson. Mais, il est plus efficace et permet une sélection plus fine des orientations des contours.

En introduisant des filtres h_0 et g_0 calculés à partir des relations suivantes :

$$h_0(n) = \frac{1}{\sqrt{2}} \int \varphi\left(\frac{1}{2}u\right)\varphi(u-n)\,du \qquad (2.45)$$

$$g_0(n) = \frac{1}{\sqrt{2}} \int \Psi\left(\frac{1}{2}u\right)\Psi(u-n)\,du \qquad (2.46)$$

S.Mallat a réussi à faire adapter les expressions suivantes :

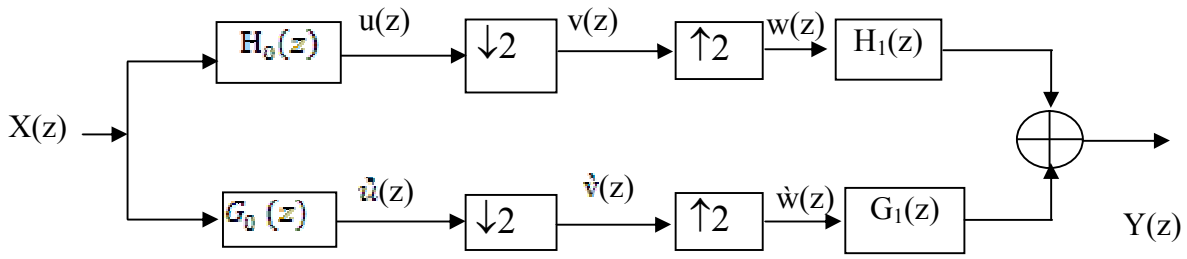
$$S_j^d f = \langle f(u), \varphi(2^{-j}u - n)\rangle \qquad (2.47)$$

$$D_{jn} = \langle f(u), \Psi(2^{-j}u - n)\rangle \qquad (2.48)$$

Pour le traitement par calculateur, une motivation essentielle de l'analyse multirésolution en traitement des images, est mise en œuvre à l'aide de filtre numériques, est donnée par la possibilité de traiter séparément chaque sous-image. La notion de multirésolution nous permet de considérer l'image sur plusieurs échelles différentes.

II.8.1. Schéma d'analyse et de synthèse [11]

Le schéma ci-dessous englobe les opérations de filtrage, de décimation (sous échantillonnage) et d'interpolation (sur échantillonnage), d'où son appellation schéma de codage en sous bande.



↓2 Décimation (conserver un échantillon sur 2)

↑2 Interpolation (intercaler un zéro entre chaque deux échantillons)

Figure II.4: **schéma d'analyse et de synthèse d'un signal par bancs de filtres**

II.8.2. Filtres miroirs quadrature (QMF) [10]

Depuis leur introduction, les filtres miroirs en quadrature ont été utilisés pour le codage de parole et des images. Dans ce type de filtres on pose $G_1(Z) = -H_0(-Z)$. La reconstruction est exacte que dans le cas de filtre RII (réponse impulsionnelle infinie), cela entraine un cout de calcul important. La limitation des filtres RII pour obtenir RIF provoque des dégradations dans la reconstruction.

II.8.3. Filtres conjugués en quadratures (QCF) [10]

Ces filtres ont été proposés par Smith et Barnwell et présentent l'avantage d'être à reconstruction exacte. Smith, Barnwell et Daubechies ont réussi à calculer les h(k) (surnommé filtre de Daubechies) en posant :

$H(w) = \frac{1}{2}(1 + e^{jw})^N Q(e^{jw})$ où $Q(e^{jw}) = \sum_{n=0}^{N-1} q(n)\ e^{jnw}$ q(o) = 0.

Les relations entre les filtres $G_0 H_0$, G_1 et H_1 sont données par :

$$\begin{cases} H_1(Z) = -G_0(-Z) \\ G_1(Z) = -H_0(-Z) \\ G_0(Z) = H_0(-Z^{-1})Z^{-(N-1)} \end{cases}$$

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

II.8.4. Algorithme pyramidal de S.Mallat [10]

L'Algorithme pyramidal adapté par S Mallat , utilise la multirésolution 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 directions horizontale, verticale et diagonale.

i- Cas monodimensionnel [10,12]

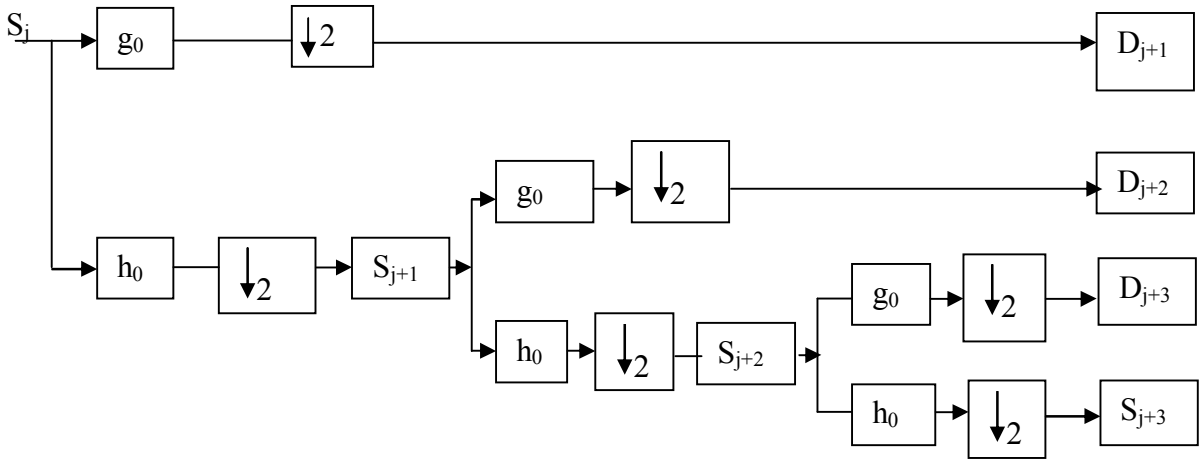


Figure II.5: algorithme de décomposition d'un signal

Le principe de ce schéma consiste à décomposer le signal S_j en deux sous signaux D_{j+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.

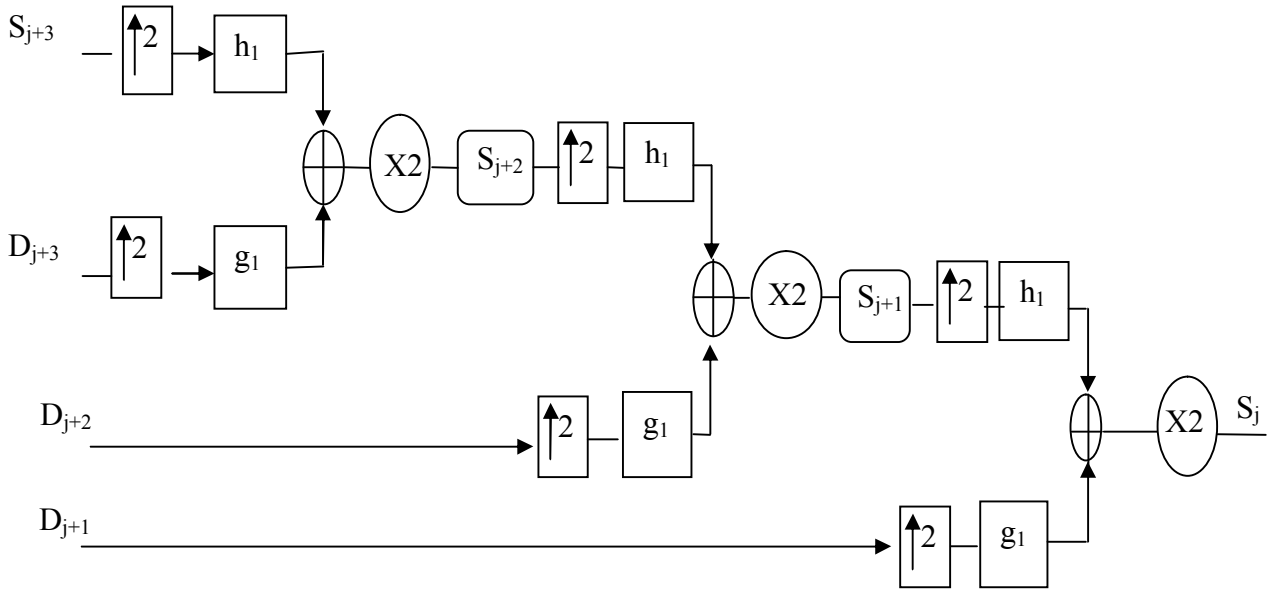
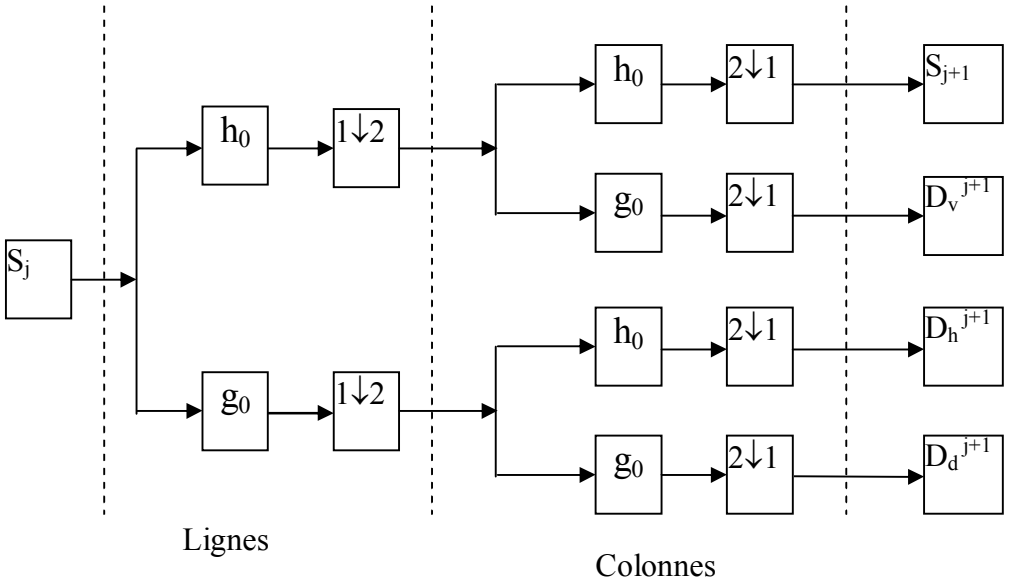


Figure II-6 : Algorithme de reconstruction d'un signal

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 ($g1_)$ 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.

ii- Cas bidimensionnel [9,10,12]

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

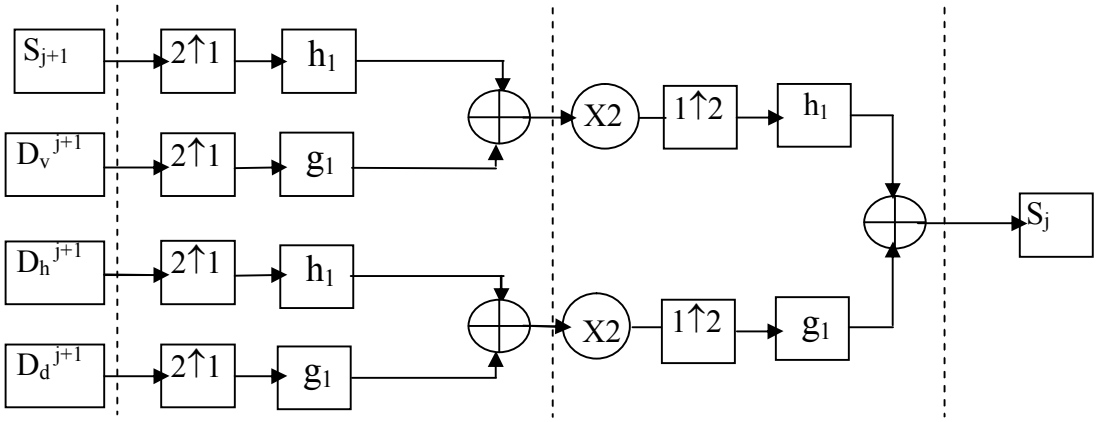


X Convolution (ligne ou colonne) avec le filtre X

1↓2 Décimation d'une ligne sur deux

2↓1 Décimation d'une colonne sur deux

Figure II-7: Etape de décomposition



X Convolution (ligne ou colonne) par le filtre X

X2 Multiplier par 2

2↑1 Mettre une ligne de zéro entre deux lignes

1↑2 Mettre une colonne de zéro entre deux colonnes

Figure II.8: Etape de reconstruction

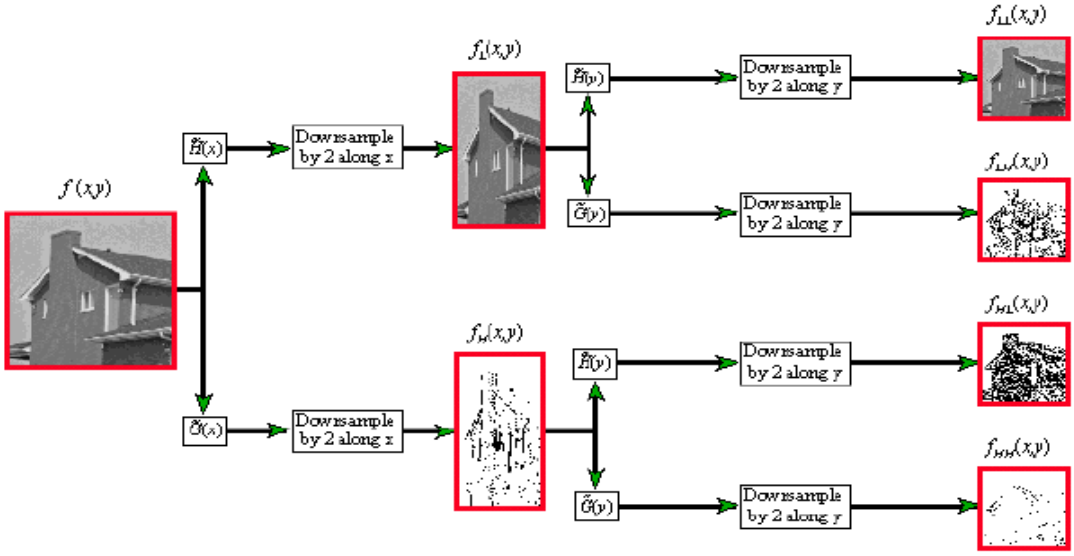


Figure II.9 Exemple de la décomposition en ondelette de l'image maison à un seul niveau

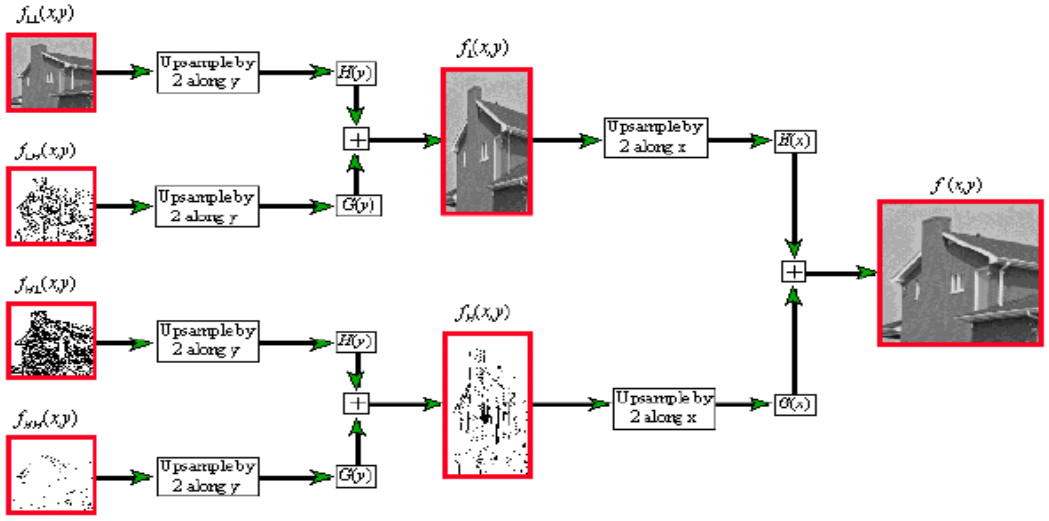


Figure II.10 Exemple de la reconstruction en ondelette de l'image maison à un seul niveau

II.9. Bases d'ondelettes biorthogonales [10]

Pour les applications en traitement d'images, il est important que les filtres associés aux fonctions échelle et ondelettes 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 limités pour diminuer le nombre d'opérations à effectuer lors des calculs et permettre une exécution rapide des traitements.
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, les seuls filtres RIF orthogonaux, symétriques et à reconstruction exacte sont ceux correspondant à la base de Haar (dont l'ordre de régularité est faible), tous les autres filtres RIF orthogonaux à reconstruction exacte sont à phase non linéaire (exemple : filtres de Daubechies).
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.

II.10. Extension aux paquets d'ondelette [10]

Le principe de la décomposition d'un signal sur une base de paquets d'ondelettes consiste à s'affranchir de la structure dyadique du pavage temps-fréquence induite par la transformée en ondelette discrète. Pour ce faire, on généralise la théorie de l'AMR en créant de nouveaux espaces de projection orthogonaux issus de la décomposition des sous-espaces de détails W_J. Ces sous espaces sont organisés selon une architecture d'arbre binaire. La projection du signal sur l'ensemble des sous espaces constitue la décomposition du signal en paquets d'ondelettes.

i- Décomposition en paquets d'ondelettes [10]

Nous avons vu précédemment que la décomposition en ondelettes à une échelle J correspondait à la représentation dans l'espace $V_j \oplus W_J \oplus \ldots \oplus W_{-1}$. Chaque étape de la décomposition consiste à séparer V_{J+1} en V_J et W_J, tout en conservant W_j. D'un point de vue temps-fréquence, il s'agit seulement d'un raffinement des basses fréquences.

Peut-on faire mieux ? Une possibilité directement réalisable consiste à décomposer les espaces W_j en sous-espaces séparant les hautes et basses fréquences, comme ont été séparés les espaces V_j.

C'est en1990 que R.R.coifman, Y.Meyer, S. Quake et M.V. Wickerhauser developpnt cette nouvelle décomposition pour l'appliquer à la compression de signaux. Peu après, R.R.coifman, Y.Meyer, S. Quake et M.V. Wickerhauser

démontre que cette séparation amène à de nouvelle représentation de $L^2(R)$, en particulier à d'autre bases ; ainsi sont nés les paquets d'ondelette.

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. sLes 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.

Le but est bien entendu de traiter les coefficients de détails avant de procéder à la reconstruction.

Les paquets d'ondelettes étant une extension de la TOD induite par la généralisation de l'analyse multirésolution, ils permettent une analyse plus affinée du signal (image) dans la perspective d'améliorer la compression tout en préservant la qualité de l'image reconstruite.

Introduction

En observant bien la nature ainsi que ses objets, on se rend compte de leurs diversités et de leurs complexités. Cette complexité vient du fait que ces objets sont très riches en formes géométriques, qui se reproduisent sans fin à toutes les échelles; ce qui nous donne une sensation d'irrégularité. **[15]**

Prenons, par exemple un cristal, si on l'observe : on remarque à l'intérieur une seule forme géométrique qui se répète plusieurs fois. Le plus impressionnant, c'est qu'on dirait, que celui-ci est l'assemblage d'un ensemble de triangles irréguliers issus de la Brisure de ce cristal. De là vient la notion de la géométrie fractale et de la similitude interne appelée aussi auto-similarité.

III.1. Théorie des fractales

III.1.1. Notion d'une fractale [13,15]

On appelle une fractale un sous –ensemble A d'un espace métrique X, tel que A est extrêmement compliqué géométriquement. Le terme « fractal » est issu du latin « fractus » qui signifie brisé, irrégulier. Un objet fractal a la particularité d'une structure géométrique qui se produit sans fin à toutes les échelles et se caractérisant par les propriétés suivantes :

- F contient des détails à n'importe quelle échelle,
- F est exactement, approximativement auto-similaire,
- La dimension fractale de F est supérieure à sa dimension topologique,
- Il existe une description algorithmique de F.

Selon la définition citée ci-dessus, un objet fractal se caractérise par :

- Ses parties qui ont la même forme ou structure que l'objet global à la différence qu'elles sont à une échelle réduite et peuvent être légèrement déformées.
- Sa forme est soit :
  - Extrêmement irrégulière,
  - Extrêmement interrompue ou fragmentée et ce quelle que soit l'échelle d'examen.
- Il contient des éléments distinctifs dont les échelles sont très variées et couvrent une très large gamme.

III.1.2. L'auto-similarité : [9,13]

Un objet fractal est dit self-similaire ou auto-similaire s'il est composé de N copies de lui même, chacune étant une réduction d'un facteur r selon toutes les coordonnées de l'objet global (auxquelles des transformations affines, comme les translations ou rotations peuvent ou non être impliquées). La notion d'auto-similarité revient dans plusieurs contextes car les objets naturelles sont auto-similaires (ou presque). Citons le cas d'un arbre qui se compose d'un tronc contenant des branches, des tiges et des feuilles. Celui-ci se décompose en un nombre fini de branches qui lui ressemblent et qui ont à leur tour un nombre fini de tiges contenant des feuilles qui sont toutes similaires. On peut dire alors, qu'à partir d'une tige et d'une feuille, l'arbre peut être reconstruit par la reproduction de ces deux éléments. Comme exemple, la fougère de Barnsley représentée par figure III.1.

Figure III.1 : La fougère de Barnsley.

Exemple :

Le triangle de Sierpinski : Ce triangle est constitué de trois triangles qui sont une réduction d'un facteur d'échelle de l'ensemble, et chacun des trois triangles est lui même constitué de trois autres triangles qui sont une réduction d'un facteur ...etc.

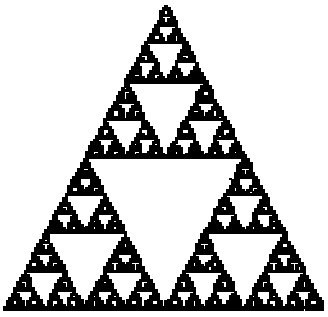


Figure III.2 : Triangle de Sierpinski.

En prenant en considération l'auto-similarité, on peut représenter les images (les formes redondantes) par un nombre restreint de paramètres, plutôt que de stocker les images de chaque feuille de chaque arbre, ou de chaque pic de chaque montagne, il est possible et préférable de faire appel à une fonction relativement simple qui génère n'importe quel niveau de détails. C'est le théorème de **M.F.Barnsley** appelé **IFS** (**I**terated **F**unctions **S**ystem).

III.1.3. Dimension fractale : [13,15]

Les mathématiciens nous ont habitués à regarder le monde de manière « idéalisée » à travers l'étude des courbes et des surfaces continues et dérivables, alors que celles-ci sont l'exception dans les phénomènes naturels. Il est admis depuis fort longtemps que la dimension du point est 0, de la droite 1, du plan 2et de l'espace 3. Pour repérer un point dans le plan, on conçoit facilement qu'il est nécessaire seulement de spécifier 2 coordonnées, tandis que dans l'espace, nous en avons besoin de 3. A la fin du XIX^{e} siècle, péano démontre qu'une coordonnée suffit. Il obtient ce résultat en construisant des courbes irrégulières qui recouvrent le plan ou l'espace. Ces révélations provoquent alors une crise dans les mathématiciens et on qualifie de « monstre », ces constructions qui viennent bouleverser les conceptions établies. A titre d'exemple, citons cette phrase de Hermite en 1893 : « Je me détourne avec effroi et horreur de cette plaie lamentable des fonctions continues qui n'ont pas de dérivées……. ».

Vers 1960, Mandelbrot ressuscite ces idées sous le nom de géométrie fractale. On considère alors des courbes irrégulières continues mais non dérivables et auto-similaires (avec homothétie interne). Mandelbrot définit l'auto-similarité dans ces termes : « une figure est auto-similaire si chaque partie de cette figure est géométriquement semblable à la figure dans son ensemble ».

De façon plus formelle, nous définissons dans un espace euclidien R^3avec $r \in R$ ($r > 0$), une transformation appelée similarité (désignée par T) comme suit :

$$T(X) = rX, X \in R \qquad (3\text{-}1)$$

La similarité transforme un ensemble S en un ensemble T(X). Un ensemble fini S est auto-similaire de rapport r et de degré N si S peut être défini comme étant l'union de N

sous-ensembles disjoints, chacun étant congru à T(S) c'est –à-dire, identique à T(S) à une translation, une échelle et / ou une rotation près.

Exemple : Courbe triadique de Koch

Désignons par A et B, deux points d'un segment. Divisons le segment AB en 3 parties égales. Remplaçons la partie centrale par deux segments de même longueur, $|AB|/3$, forme un pic. Appliquons les mêmes transformations sur les quatre segments résultants et ainsi de suite jusqu'à l'infini (figure III 3).

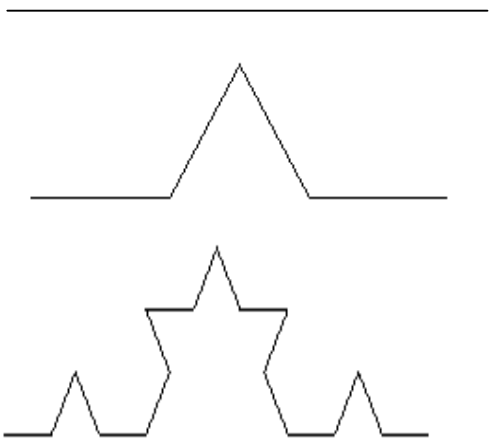


Figure III.3 : Courbe triadique de Koch

La courbe triadique de koch est une courbe irrégulière non dérivable et auto-similaire. Peu importe l'échelle où nous regardons cette courbe, elle a toujours le même comportement. A chaque étape, un segment est transformé en N= 4 parties égales, déduites du segment par une auto-similarité de rapport r = 1/3. En fait, chaque segment est transformé en 4 nouveaux segments dont les longueurs sont transformées dans le rapport 1/3.On peut aussi être intéressé à calculer la longueur (L) d'une telle courbe. En supposant que la longueur du segment AB est 1, on peut voir facilement que :

L= 1+ 1/3 + 4/9 + 16/27 +…..

$$= 1 + 1/3 + \sum_{i=0}^{\infty} (4/3)^i \qquad (3\text{-}2)$$

Nous sommes en présence d'une série géométrique divergente de raison 4/3. Par conséquent, L tend vers l'infini.

Déterminons maintenant la longueur de la courbe à partir d'une certaine échelle, c'est-à-dire, avec un bâton de longueur ε donnée. Nous obtiendrons différentes mesures L (ε) selon la longueur de ce bâton. Par exemple, si AB est de longueur unitaire, on obtient :

$$L(1) = 1$$

$$L(1/3) = 4/3$$

$$L(1/9) = 16/9.$$

$$\lim_{\varepsilon \to 0} L(\varepsilon/3) = \infty \qquad (3\text{-}3)$$

Sachant que L ($\varepsilon/3$)= 4/3 L (ε), on peut poser : L (ε) = ε^A. Cherchons à trouver la valeur de A. on obtient :

$$4/3\ \varepsilon^A = L\ (\varepsilon/3) \qquad (3\text{-}4)$$

Ou encore,

$$3^{-A} = 4/3$$

C'est-à-dire,

$$A = 1 - \log 4/\log 3 \qquad (3\text{-}5)$$

Par conséquent, dans le cas de la courbe triadique de Koch,

$$L\ (\varepsilon) = \varepsilon^{(1-\log 4/\log 3)} \qquad (3\text{-}6)$$

On peut effectuer les mêmes calculs pour différentes constructions ; on aboutit alors au résultat général suivant :

$$L\ (\varepsilon) = \varepsilon^{(1-D)} \qquad (3\text{-}7)$$

$$\text{Où} \quad D = (\log N)/\log(1/r).$$

N= nombre de parties de segment générées lors de la transformation d'un segment,

r = rapport d'auto-similarité.

En conclusion, la constante D est considérée comme la dimension fractale de toute construction fractale. Pour la courbe de Koch la dimension qu'on recherche est :

$D = \frac{log4}{log3} \approx 1.26185$, qui est unique et invariable pour toute partie comme pour l'ensemble de la courbe formant l'image « du flocon de neige » indique l'irrégularité dont la densité est intermédiaire entre une ligne (dimension1) et une surface (dimension2).

III.2. Espace métrique : [9,13,15]

Un espace métrique est un ensemble X sur lequel une fonction « distance » à valeur réelles d : X*X →R est définit avec les propriétés suivantes :

- d(a,b) ≥ 0 , ∀ a,b ∈ X
- d(a,b) = 0 si et seulement si a=b, ∀ a,b ∈ X
- d(a,b) = d(b,a), ∀ a,b ∈ X (symétrie)
- d(a,c) ≤d(a,b) + d(b,c), ∀ a,b,c ∈ X

Le nombre réel d (a,b) est appelé la distance de a à b.

Une telle fonction « d » est dite « métrique ».

III.3. Théorie d'un IFS : [9, 13,15]

III.3.1. Principe :

Le noyau de compression d'image basée sur les fractales, est l'utilisation des similarités au sein de la même image à différentes échelles.

La théorie d'un IFS (Iterated Functions System) est une extension de la géométrie classique. Elle utilise un ensemble de transformations affines pour exprimer des relations

entre les parties d'une même image. En utilisant seulement ces relations, on peut exprimer et définir des images complexes (nuages, feuilles, arbre, ….etc.).

Donc avec la théorie des IFS, on peut décrire aussi clairement des formes complexes que de décrire des formes régulières avec une autre méthode.

III.3.2 Méthodologie : [9, 13,15]

On appel un IFS W de facteur de contraction S, tout ensemble de n transformations contractantes linéaires : w_1, w_2, …..,w_n définies dans un espace métrique (X ,d) avec :

$$S = \text{Max} (S_i , i = 1,..., n) \qquad (3\text{-}8)$$

Où :

X : espace des images ; d : une mesure de distance.

On le note :{X ; w_i, i=1,2,...,n }, les transformations w_i composants un IFS sont des transformations affines.

Soit A un sous ensemble d'un plan, considérons A comme une image, le collage obtenu par l'application de N contractions à A et l'assemblage du résultat peut être exprimé par :

$$W(A) = w_1(A) \cup w_2(A) \cup w_3(A) \cup .... \cup w_n(A) \qquad (3\text{-}9)$$

Les w_i peuvent être décrites comme une combinaison de rotation, de réduction d'échelle et de translation de coordonnées des axes dans un espace à n-dimension(s).

$$w_i = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix}$$

Où : a, b, c, d, e, et f représentent des nombres réels (paramètres de transformation).

On peut générer les valeurs des transformations comme suit :

$$a = r \cos\theta \, , \; b = -s \sin\varphi \, , \; c = r \sin\theta \, , \; d = s \cos\varphi$$

Où :

r : est le facteur de réduction sur x

s : est le facteur de réduction sur y

θ: est l'angle de rotation sur x

φ: est l'angle de rotation sur y

e : est la translation en x

f : est la translation en y

Comment peut-on trouver une transformation affine W qui produit un effet désiré ?

Voyons comment trouver la transformation affine qui transforme une grande feuille en une petite (voir figure III.4), i.e. trouver les paramètres a, b, c, d, e et f pour lesquels la transformation W a la propriété suivante :

W (grande feuille) ≈ *petite feuille.*

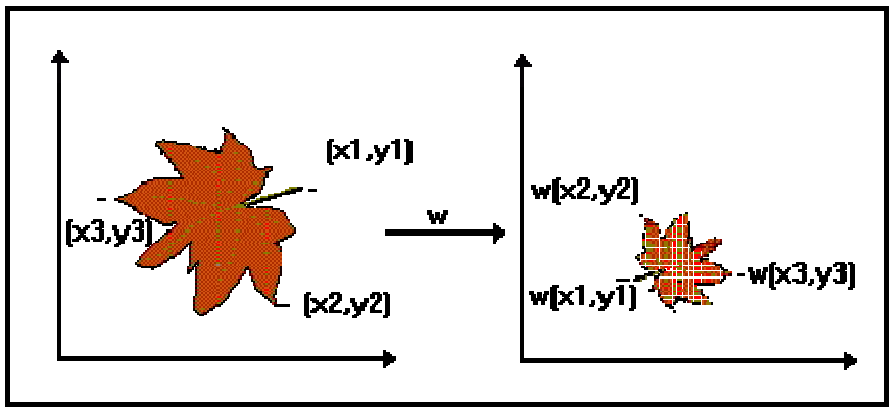


Figure III.4: transformation d'une grande feuille en une petite feuille.

La procédure à suivre est :

- Introduire des axes x et y.
- Marquer trois points sur la grande feuille et déterminer leurs coordonnées (x_1,y_1), (x_2,y_2), (x_3,y_3).

- Marquer les trois points correspondants sur la petite feuille et déterminer leurs coordonnées (a_1, b_1), (a_2, b_2), (a_3, b_3) respectivement.

Déterminer les valeurs des coefficients a, b et e par la résolution du système des trois équations suivant :

$$\begin{cases} x_1 a + y_1 b + e = \alpha_1 \\ x_2 a + y_2 b + e = \alpha_2 \\ x_3 a + y_3 b + e = \alpha_3 \end{cases}$$

Déterminer c, d, et f de la même manière par le système des trois équations suivant :

$$\begin{cases} x_1 c + y_1 d + f = \beta_1 \\ x_2 c + y_2 d + f = \beta_2 \\ x_3 c + y_3 d + f = \beta_3 \end{cases}$$

Chacune des transformations w_i doit avoir une probabilité Pi, avec une importance relative aux autres transformations.

La somme des probabilités est égale à 1.

III.3.3. Association des probabilités : [13]

On associe à chaque transformation affine w_i d'une image A une probabilité Pi calculée par l'expression suivante :

$$P_I(w_i) \approx \sum_{k-1}^{n} \frac{|a_i d_i - c_i b_i|}{|a_k d_k - c_k b_k|} \qquad (3\text{-}10)$$

Où :

$$w_i \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a_i & b_i \\ c_i & d_i \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} e_i \\ f_i \end{bmatrix}$$

Pour i = 1,2,...,n.

Remarque : Si un des nombres Pi est nul, il faudra le remplacer par une petite valeur positive (0.001 par exemple) et ajuster les autres probabilités de façon à ce que leur somme soit égale à 1.

III.3.4. Attracteur d'un IFS : [13,15]

Soit un IFS $\{IR^2 ; w_i, i=1,2,...,n\}$. L'opérateur W : $H(IR^2) \in H(IR^2)$ défini par :

$$W(B) = \coprod_{i=1}^{n} wi(B)), \quad \forall B \in H(IR^2) \qquad (3\text{-}11)$$

est contractant et a pour facteur de contraction celui de l'IFS. L'opérateur W possède un unique point fixe At donné par :

$$At = W(At) = \lim_{n \to 0} w^{on}(x), \ \forall\ x \in H(IR^2). \qquad (3\text{-}12)$$

L'objet At est appelé *attracteur de l'IFS* et il est égal à l'union de n copies de lui même transformées par $w_1, w_2, ..., w_n$.

Exemple :

Soit un IFS $\{IR^2 ; w_1, w_2, w_3\}$ composé des transformations affines suivantes :

$$w_1\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 1/2 & 0 \\ 0 & 1/2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0 \\ y_0/2 \end{bmatrix}$$

$$w_2\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 1/2 & 0 \\ 0 & 1/2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} -x_0/2 \\ -y_0/2 \end{bmatrix}$$

$$w_3\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 1/2 & 0 \\ 0 & 1/2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} x_0/2 \\ -y_0/2 \end{bmatrix}$$

Son facteur de contraction est égal à 1/4.

Toutes les transformations réduisent l'objet initial par 0.5 (dans chaque direction). Le point fixe de cet IFS est appelé " triangle de Sierpinski " (Figure III.5)

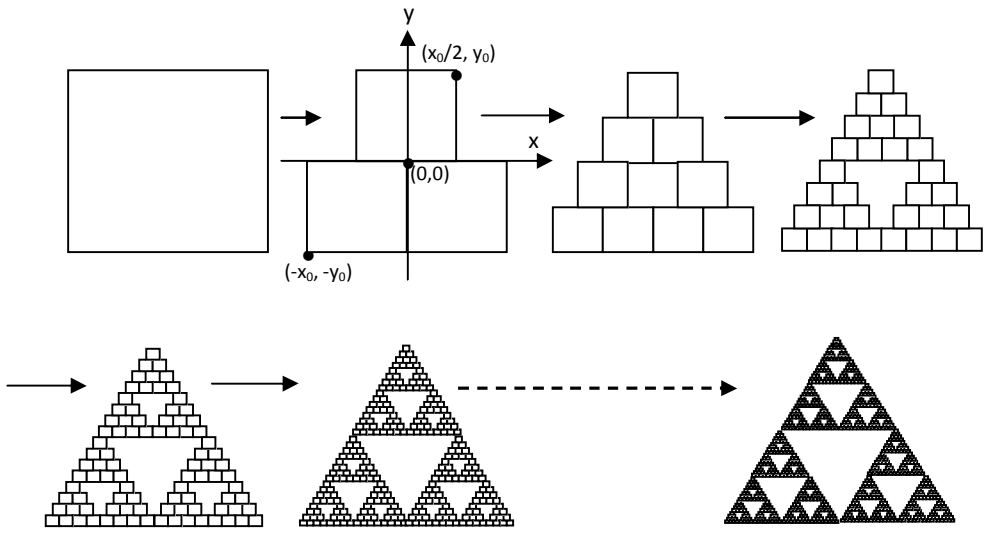


Figure III.5 : construction de triangle de Sierpinski.

Le carré initial centré à l'origine, de cotés de longueur $2x_0$, $2y_0$ est transformé en trois carrés homothétiques par les transformations w_1, w_2, w_3. Ce processus est en suite itéré jusqu'à un triangle de forme fixe.

III.3.5. Théorème de collage : [13,14,15]

Soit l'espace métrique complet (X , d). Soit un point A ∈ H(X) et un IFS

{X ;w_i, i=1,2,...,n} avec le réel s, 0 < s < 1 pour facteur de contraction.

On vérifie la relation :

$$h_d(A, At) \leq \frac{1}{1-s}.h_d(A, \coprod_{i=1}^{n} W(A)) \qquad \forall A \in H(X). \qquad (3\text{-}12)$$

Ce théorème fournit une borne supérieure à la distance de Hausdorff h_d entre un point A inclus dans H(X) et l'attracteur At d'un IFS.

Il indique que s'il est possible de transformer un objet A de manière à vérifier :

(3-13)

$$A \approx W(A)$$

tout en s'assurant que W est contractante, alors le point fixe At de l'opérateur W est proche de A.

III.3.6. Théorème de collage généralisé : [13,14,15]

Considérons l'opérateur W finalement contractant, avec pour exposant de contraction l'entier n ; il existe alors un unique point fixe $x_f \in R^2$ tel que :

$$x_f = W(x_f) = \lim_{n \to +\infty} w^{on}(x) \, , \forall x \in H(X). \qquad (3\text{-}14)$$

Dans ce cas :

$$h_d(A, At) \leq \frac{1}{1-s} \cdot \frac{1-\alpha^n}{1-\alpha} h_d(A, W(A))$$

Où α est le facteur de Lipchitz de W et s est le facteur de contraction de W

III.3.7. Problème inverse : [13,15]

Jusqu'ici, on a vu que les IFS permettent la génération d'images totalement auto-similaires en utilisant un nombre restreint de transformations. Seulement, dans le cadre de l'application de la géométrie fractale pour la compression d'images et étant donnée une image initiale (image fractale), il est important de trouver le bon IFS de telle sorte que l'attracteur de l'IFS est l'image à coder. La recherche de cet IFS permettant d'approximer une image donnée est un problème inverse difficile à résoudre. Barnsley montre qu'à l'aide du théorème de collage on peut résoudre ce problème. Toutefois, ce théorème ne fournit aucune méthode pour trouver l'IFS. Cette image approximée reste quasiment invariante et on obtient des taux de compression trop élevés, puisque la mémorisation des coefficients de la transformation fractale W nécessite " moins d'informations " que la mémorisation de l'image originale. Dans ce cas, on dit que la compression d'image par fractale est une méthode de compression avec perte du fait que l'attracteur ne constitue qu'une approximation de l'image originale.

Cependant, l'approximation d'une image naturelle, qui n'est pas auto-similaire, à partir d'elle même n'est pas facile à réaliser comme la génération d'une image fractale à partir d'un IFS.

Pour venir à bout de ce problème, et pour exploiter la similarité entre les zones d'une image réelle, il faut délimiter et séparer ces zones ; c'est ce qu'on appel les PIFS (Partitionned Iterated Functions System) qui constituent une extension des IFS et qui peuvent donc coder n'importe quelle image.

III.4. La théorie des PIFS : [13,15]

III.4.1. Principe:

Les images réelles ne contiennent pas le même type d'auto-similarité que celui des fractales tel que le triangle de Sierpinski ou la fougère de Barnsley, où l'image est constituée de l'union des répliques réduites de l'image entière. En effet, l'image réelle est formée de copies des parties d'elle-même. On le voit clairement sur l'image de LENA (Figure.III.6), où un échantillon de régions sont similaires à différentes échelles : une portion de son épaule chevauche (ou repose sur) une région qui lui est presque identique, et la portion de reflet du chapeau dans le miroir est similaire (après transformations) à une partie de son chapeau. On devra alors, tolérer quelques erreurs, lors du codage de l'image comme un ensemble de transformations. L'image codée ne sera pas donc une copie identique de l'image originale mais plutôt une approximation d'elle-même.

Figure III.6 : Les parties auto-similaires dans l'image Lena.

III.4.2. Point fixe du PIFS : [13,14,15]

Dans ce cas, le théorème du point fixe reste valable. Ainsi, pour toute l'image initiale f_0, la séquence :

$$W(f_0), W(W(f_0)),\ldots, W^{oi}(f_0)$$

Converge vers une image f qui vérifie W(f)=f

L'image f, notée |W|, est appelée le point fixe ou l'attracteur de la transformation W.

III.4.3. Compression par PIFS : [13,15]

La compression d'une image par fractale repose sur des transformations que nous appelons *transformations fractales* et qui consiste à transformer l'image à l'aide d'un opérateur finalement contractant, de manière à ce que son aspect visuel reste quasiment inchangé. Pour cela, la transformation de l'image est composée de n sous-transformations élémentaires, chacune opérant sur un bloc de l'image, de la manière suivante (voir Figure III.7) :

L'image A (du départ) est partitionnée en n blocs r_i appelés blocs destination (ou blocs range). Elle s'écrit : $A=U^{n}_{i=1}\ ri$

Cette partition du support de l'image en blocs destination est appelée partition R.

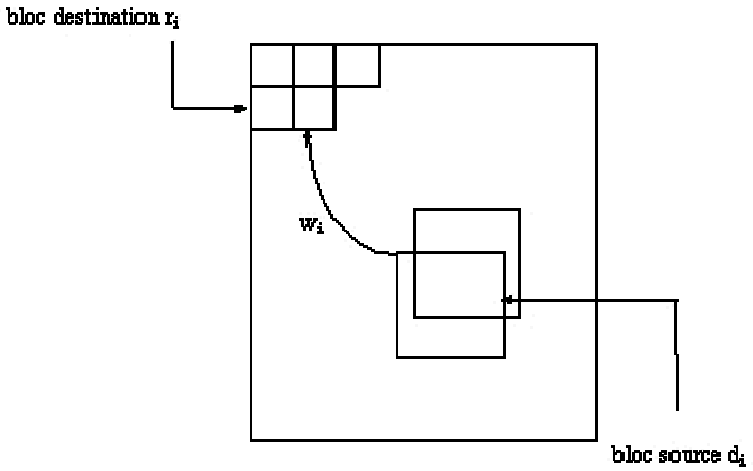


Figure III.7 : Blocs destination r_i et blocs source d_i

Chaque bloc destination est ensuite mis en correspondance avec un autre bloc transformé $w_i(d_i)$ lui " ressemblant " au sens d'une mesure d'erreur sur les niveaux de gris. Le bloc d_i, appelé bloc source est recherché à travers d'une librairie composée de Q blocs destination appartenant à l'image. Les Q blocs ne forment pas nécessairement une partition de l'image mais sont représentatifs de toute l'image. La géométrie de la région d_i doit être proche de celle de bloc r_i, de façon à limiter le temps de calcul lors des comparaisons inter-blocs, puisque ceux-ci sont liés à la complexité de la transformation spatiale utilisée.

Une seconde contrainte impose que les blocs source soient en moyenne de surface supérieure à celle des blocs destination.

La transformation W de l'image A est formulée à l'aide de l'équation suivante :

$$W(A)=\coprod_{i=1}^{n} w_i(d_i)=\coprod_{i=1}^{N} \overline{r_i} \approx \coprod_{i=1}^{n} r_i = A \qquad (3\text{-}15)$$

Où $\overline{r_i}$ est l'approximation du bloc destination r_i obtenue en transformant le bloc source d_i par w_i (l'opération permettant d'obtenir le bloc $\overline{r_i}$ à partir du bloc d_i est appelée opération de collage).

Chaque transformation w_i s'écrit sous la forme :

$$w_i \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} a_i & b_i & 0 \\ c_i & d_i & 0 \\ 0 & 0 & s_i \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} e_i \\ f_i \\ l_i \end{bmatrix}$$

Où i = 1,...,n.

On peut l'écrire aussi :

$$w_i(A) = w_i(x,y,f(x,y)),\ \mathrm{f(x,y)} = \text{niveau de gris.}$$

Dans ce cas, Le niveau de gris constitue la troisième dimension, s_i contrôle le contraste et l_i la luminance.

Les conditions de base pour la conception d'un système de codage d'image digital basé sur les fractales sont :

- Le choix de partitionnement de l'image ;
- Le choix de la mesure de distorsion ;
- Le choix des transformations fractales.
-

III.4.3.1 Partitionnement de l'image : [9, 13,15]

Pour minimiser le nombre de blocs à coder et de là, le nombre de transformations locales, différents modèles de partitionnement d'images pour la compression par fractales ont été étudiés, nous citerons :

Partitionnement carré : uniforme ou à deux niveaux, très utilisé de fait qu'il est facile à réaliser, il a été adopté dans la méthode de Jacquin.

Partitionnement quadtree : Vu l'utilisation de blocs de taille fixe dans le partitionnement carré, ce qui constitue un lourd inconvénient (existence de régions contenants beaucoup de détails), une généralisation de ce dernier est l'utilisation d'un partitionnement quadtree.

Réalisé par un découpage récursif en blocs carrés (quatre quadrants de même dimension), ce processus est représenté par un arbre de degré quatre (à chaque noeud interne, correspond quatre fils), la racine correspond à l'image entière.

Partitionnement rectangulaire : Il est semi-rigide dans le sens où les arêtes des blocs demeurent obligatoirement soit horizontales, soit verticales : la géométrie des blocs est à orientation définie.

Partitionnement HV (Horizontal Vertical) : L'image est partitionnée horizontalement et verticalement avec récursivité pour former deux nouveaux rectangles. Il est flexible car la position de la partition est variable.

Partitionnement à base de polygones : L'image est partitionnée adaptativement en polygones (processus récursif). Cette technique permet une meilleure représentation des objets formant l'image. Il est flexible car la position de la partition est variable et permet de représenter les régions avec beaucoup de détails.

Partitionnement triangulaire : Il est souple car il est calculé sur un ensemble de points pouvant être positionnés à peu prés n'importe où sur le support de l'image.

III.4.3.2 La mesure de distorsion : [13,15]

Comme la distance de Haussdorf entre deux sous-ensembles de IR^n. Dans le cas des images, on définit la distance comme une mesure de degré de proximité ou de ressemblance entre deux images. Plusieurs métriques existent dans l'espace des images, on cite :

- **La métrique SUP :**

définie par :

$$d(f,g) = SUP \mid f(x,y) - g(x,y) \mid \qquad (3\text{-}16)$$

$(x,y) \in I^2$

$f(x,y) \in [0,n] = I$; $g(x,y) \in [0,n] = I$. (En considérant les images f et g comme des images carrées de taille nxn).

Cette métrique recherche la position (x,y) où les deux images f et g diffèrent le plus, et considère cette différence comme étant la différence entre elles. Elle est plus utilisée en théorie.

- **L_2 DISTANCE (distance quadratique moyenne) :**

$$dL_2\,(f,g) = L_2 \text{ distance } (f,g) = \frac{1}{n}\sum_{x=1}^{n}\sum_{y=1}^{n}[f(x,y) - g(x,y)]^2 \qquad (3\text{-}17)$$

- **L_1 DISTANCE :**

$$dL_1\,(f,g) = L_1 \text{ distance } (f,g) = \frac{1}{n^2}\sum_{x=1}^{n}\sum_{y=1}^{n}[f(x,y) - g(x,y)]^2 \qquad (3\text{-}18)$$

- **La distance RMS (Root Mean Squared):**

$$RMS(f,g) = \frac{1}{n}\sqrt{\sum_{x=1}^{n}\sum_{y=1}^{n}[f(x,y)-g(x,y)]^2} \qquad (3\text{-}19)$$

Ces trois dernières distances sont très utilisées car elles sont faciles à calculer et donnent une bonne indication sur le degré de ressemblance entre deux images.

III.5. Méthode de JACQUIN : [9,13,15]

L'approche de A. Jacquin est fondée sur une partition R régulière à géométrie carrée. L'image est partitionnée en blocs destination (ou blocs parents) carré de taille fixe égale à BxB pixels (B = 8). L'algorithme recherche, pour chacun des blocs destination r_i, le bloc source d_i de taille DxD (D = 2B) qui minimise l'erreur d ($r_i , \overline{r_i}$) où $\overline{r_i}$ est l'approximation de r_i calculée à partir du bloc source d_i. La mesure d'erreur *d* est donnée par :

$$d\,(r_{ij} - \overline{r_{ij}}\,)\,) = \sum_{j=1}^{B^2} (r_{ij} - \overline{r_{ij}})^2 \qquad (3\text{-}20)$$

où r_{ij} et $\overline{r_{ij}}$ sont respectivement les valeurs des pixels d'indice j à l'intérieur du bloc original r_i et du bloc collé $\overline{r_i}$.

III.5.1. Collage d'un bloc source sur un bloc destination : [9,13,15]

L'opération de collage d'un bloc source d_i sur un bloc destination r_i, réalisée par la transformation w_i, se décompose en deux parties :

- **La transformation spatiale (géométrique) :**

Elle ramène le bloc source d_i de taille DxD à l'échelle et au-dessus du bloc destination r_i de taille BxB. Le bloc ainsi résultant, noté $b_2^{(i)}$, est obtenu par décimation des pixels du bloc source. : un pixel de coordonnées (x_i , y_j) dans $b_2^{(i)}$ est donné par l'équation suivante :

$$b_2^{(i)}(x_i , y_j) = \frac{1}{4}\left[t_i(x_k, y_l) + t_i(x_k, y_{l+1}) + t_i(x_{k+1}, y_l) + t_i(x_{k+1}, y_{l+1})\right] \qquad (3\text{-}21)$$

dans laquelle (x_k , y_l) sont les coordonnées d'un pixel de niveau de gris noté t_i à l'intérieur du bloc d_i.

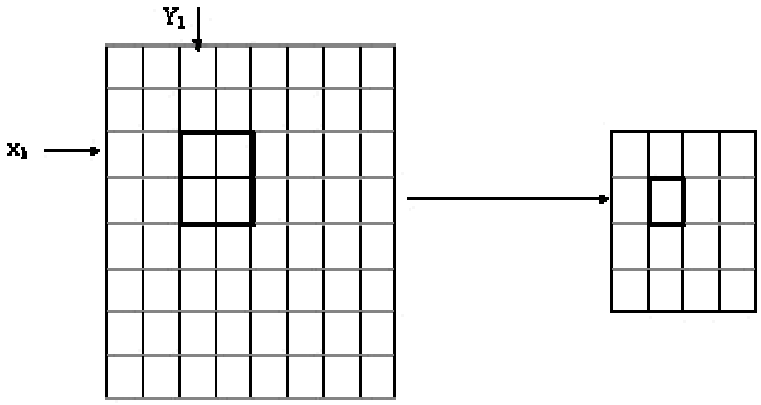


Figure III.8 : Trasformation spatiale d'un bloc "source"

- **La transformation massique :**

Cette transformation agit sur la luminance des pixels du bloc $b_2^{(i)}$ pour approximer le bloc destination r_i.

Les transformations massiques utilisées appartiennent à deux classes :

1ere classe : Elles agissent seulement sur les valeurs des pixels :

1) Absorption par g_0 : La valeur du pixel (i , j) est affectée (forcée) par la valeur de g_0.

$$M_0(b_2^{(i)})_{i,j} = g_0,\ g_0 \in \{0,...,255\}. \qquad (3\text{-}22)$$

2) Translation de luminance *:* On ajoute à la valeur de pixel (i , j) une constante Δg.

$$M_1(b_2^{(i)})_{i,j} = b_2^{(i)}{}_{i,j} + \Delta g,\ \Delta g \in \{-255,...,255\} \qquad (3\text{-}23)$$

3) Echelonnage par α ou modification de contraste (contrast scaling) :

$$M_2(b_2^{(i)})_{i,j} = \alpha\, b_2^{(i)}{}_{i,j},\ \alpha \in [0,1]. \qquad (3\text{-}24)$$

4) Inverse de couleurs : On inverse la couleur du pixel (i,j) en la substituant à la valeur maximale des niveaux de gris g_{max}

$$M_3 (b_2^{(i)})_{i,j} = g_{max} - b_2^{(i)}{}_{i,j} \qquad (3\text{-}25)$$

Où $M_k (b_2^{(i)})_{i,j}$ est la transformation massique numéro k.

2eme classe : Elles ne modifient pas les valeurs des pixels, mais les mélangent ou les déplacent dans les blocs, elles sont appelées les isométries, elles sont au nombre de huit :

1°) Identité :

$$ISO_0(b_2^{(i)})_{i,j} = b_2^{(i)}{}_{i,j.} \qquad (3\text{-}26)$$

2°) Réflexion orthogonale par rapport à l'axe de symétrie horizontal (i = (B-1)/2) :

$$ISO_1 (b_2^{(i)})_{i,j} = b_2^{(i)}{}_{B\text{-}1\text{-}i,j.} \qquad (3\text{-}27)$$

3°) Réflexion orthogonale par rapport à l'axe de symétrie vertical (j = (B-1)/2) :

$$ISO_2(b_2^{(i)})_{i,j} = b_2^{(i)}{}_{i,B\text{-}1\text{-}j.} \qquad (3\text{-}28)$$

4°*) Réflexion orthogonale par rapport à la première diagonale* (i = j) :

$$ISO_3(b_2^{(i)})_{i,j} = b_2^{(i)}{}_{j,i.} \qquad (3\text{-}29)$$

5°) *Réflexion orthogonale par rapport à la seconde diagonale* (i+j =B-1) :

$$ISO_4 (b_2^{(i)})_{i,j} = b_2^{(i)}{}_{B\text{-}1\text{-}j,\ B\text{-}1\text{-}i.} \qquad (3\text{-}30)$$

6°) Rotation à +90° *:*

$$ISO_5 (b_2^{(i)})_{i,j} = b_2^{(i)}{}_{j,\ B\text{-}1\text{-}i.} \qquad (3\text{-}31)$$

7°) Rotation à +180°:

$$ISO_6(b_2^{(i)})_{i,j} = b_2^{(i)}{}_{B\text{-}1\text{-}i,\ B\text{-}1\text{-}j.} \qquad (3\text{-}32)$$

8°) Rotation à$+270^0$:

$$ISO_7(b_2^{(i)})_{i,j} = b_2^{(i)}{}_{B-1-j,\, i.} \qquad (3\text{-}33)$$

III.5.2. Classification des blocs : [9,13]

La complexité de la transformation massique dépend de la nature du bloc r_i considéré. Jacquin propose pour cela de classifier les blocs carrés à l'aide de la méthode développée par Ramamurthi et Gersho : trois classes regroupent :

- Les blocs homogènes (blocs shade) *:* blocs presque uniforme, qui présentent des variations de luminance négligeables ;
- Les blocs texturés (blocs midrange) *:* se caractérisent par des gradients moyens, mais ne définissent aucun contour ;
- les blocs avec contours (blocs edge) *:* simples et divisés, ils présentent une brusque variation de niveau de gris le long de la limite d'un objet.

Selon la classe à laquelle appartient le bloc destination r_i, une transformation massique plus ou moins complexe lui est associée. Celle-ci dépend du bloc décimé $b_2^{(i)}$ et/ou d'un bloc constant $b_1^{(i)}$ formé de pixels tous égaux à un. Au bloc $b_2^{(i)}$ sera associé un coefficient d'échelle noté $\beta_2^{(i)}$ et au bloc $b_1^{(i)}$ un coefficient de décalage noté $\beta_1^{(i)}$. Le choix de la transformation est fonction de la procédure suivante :

Si le bloc r_i est homogène : absorption des niveaux de gris de r_i. Aucune recherche de blocs source d_i n'est effectuée. La transformation de r_i, codée sur s bits, est donnée par :

$$\overline{r_i} = \beta_1^{(i)} b_1^{(i)} \qquad (3\text{-}34)$$

Où l'entier $\beta_1^{(i)}$ est compris entre 0 et 255.

Si le bloc r_i est texturé : recherche de bloc source d_i, puis modification de contraste et décalage. La transformation de d_i , codée sur m bits, est donnée par :

(3-35)

$$\overline{r_i} = \beta_2^{(i)} b_2^{(i)} + \beta_1^{(i)} b_1^{(i)}$$

Où $\beta_2^{(i)}$ appartient à l'ensemble {0.7, 0.8, 0.9, 1.0}et l'entier $\beta_1^{(i)}$ est compris entre -255 et 255.

Si le bloc r_i contient des contours : recherche du bloc source d_i , puis modification de contraste, décalage et isométrie discrète ISO_i . La transformation de d_i , codé sur e bits est donnée par :

$$\overline{r_i} = ISO_i(\beta_2^{(i)} b_2^{(i)} + \beta_1^{(i)} b_1^{(i)}) \qquad (3\text{-}36)$$

Où $\beta_2^{(i)}$ appartient à l'ensemble {0.5, 0.6, 0.7, 0.8, 0.9, 1.0} et l'entier $\beta_1^{(i)}$ est compris entre (-255 et 255).

Lorsque le bloc destination est texturé ou recouvre des contours, le coefficient d'échelle β_2 est calculé de manière à rendre égaux les écarts types des deux blocs $b_2^{(i)}$ et r_i. Il est ensuite approximé par le coefficient appartenant à un ensemble de valeurs prédéfinies réelles positives, et inférieures à 1. Le coefficient de décalage β_1 est calculé de manière à ce que les moyennes des pixels des deux blocs $b_2^{(i)}$ et r_i soient égales. Il n'est pas quantifié.

III.5.3. Génération des blocs domaine (source) : [9,13]

La génération des blocs source d_i est effectuée en déplaçant sur le support de l'image d'une position un bloc carré (fenêtre) par un pas de h = v = 4 pixels horizontalement vers la droite ou verticalement vers le haut, de manière à ce qu'elle reste à l'intérieur du support. La fenêtre étant initialement située dans le coin gauche bas du support. Lorsque deux blocs sont comparés, les huit isométries discrètes sont considérées. Pour une image de taille 256*256 pixels, une telle recherche est ainsi effectuée au travers d'une librairie composée de Q blocs source où :

$$Q = k\left[\frac{m-D}{h} + 1\right]^2 \qquad (3\text{-}37)$$

avec : k =8 isométries, m = coté de l'image en pixels (256), D = coté du bloc source (16),

h =pas du déplacement(4)

Nous aurons Q = 29768 blocs sources.

Dans le cas d'une image de taille 512*512 pixels, le nombre Q de blocs source s'élève à 125000.

III.5.4 Taux de compression : [13]

Le taux en bits par pixels est donné par l'expression suivante :

$$\frac{N_m m + N_s S + N_e e}{(N_m + N_s + N_e)B^2} bits/pixels. \qquad (3\text{-}38)$$

Où : N_m, N_s, N_e désignent le nombre total des blocs destination texturés, homogènes et avec contours dans l'image. m,e,s le nombre de bits servant à coder chacun d'eux et B est le cote du bloc destination.

III.5.5. Optimisation : [9,14]

JACQUIN propose dans un deuxième temps, de rediviser les blocs parents collés $\overline{r_i}$ en quatre sous-blocs destination de taille 4*4 pixels appelés blocs enfants. Les blocs obtenus sont comparés à leurs correspondants dans l'image originale, au sens de la mesure d'erreur donnée précédemment, avec b = 4. Si l'erreur est inférieure à un seuil donné, le code de la transformation parent est sauvegardé, sinon, ils sont codés séparément en recherchant dans l'image le meilleur bloc source de taille 8*8. Le processus de collage est dans ce cas appelé collage enfant.

Si pour un bloc parent, trois ou quatre collages enfant sont nécessaires, seuls sont codés les quatre collages enfants. Si un ou deux collages enfant sont nécessaires, le bloc parent est codé par le collage parent complété des collages enfants. Douze configurations sont possibles :

Une transformation parent, pas de transformation enfant (1 configuration).

Une transformation parent, une transformation enfant (4 configurations).

Une transformation parent, 2 transformations enfant (6 configurations).

Pas de transformation parent, 4 transformations enfant (1 configuration).

III.5.6. Opération de codage : [9,13,15]

Après le collage d'un bloc source (parent ou enfant) d_i sur un bloc destination (parent ou enfant) r_i comprend :

1) l'indice du bloc source retenu parmi les Q blocs de la librairie à condition que ceux-ci soient rangés dans une liste de blocs et que leur organisation sur le support de l'image soit connu. Sinon, il est nécessaire de mémoriser les coordonnées (x_k , y_l) d'un pixel de référence dans le bloc d_i (par exemple, le coin supérieur gauche dans le cas d'un bloc carré).

2) l'isométrie utilisée lors du collage (une parmi huit).

3) les coefficients β_1 et β_2 de la transformation massique.

Cette information est associée à chacun des n blocs destination de la partition R. Elle est codée sur un nombre variable de bits puisque la mémorisation de l'ensemble des trois points n'est pas toujours nécessaire. Elle dépend de la transformation massique utilisée.

III.5.7. Opération de décodage : [9, 13,15]

Afin de reconstruire l'image après codage, la procédure de décodage consiste simplement à itérer la transformation W sur n'importe quelle image initiale A_0, jusqu'à ce que la convergence vers une image stable soit observée.

La séquence d'image $A_n = W_n(A_0)$ est appelée séquence de reconstruction fractale.

La transformation d'une image d'après un code fractale est faite séquentiellement. Chaque bloc destination de l'image $W^{n+1}(A_0)$ est obtenu en appliquant la transformation W_i au bloc source correspondant de l'image $W^n(A_0)$, avec $n>0$.

Le processus de décodage s'arrête lorsque la distorsion entre deux images successives devient inférieure à un seuil fixé.

Conclusion

Dans ce chapitre, nous avons introduit les bases mathématiques nécessaires à la compréhension de la théorie des IFS et dans un deuxième lieu, l'extension de cette méthode en IFS partitionnés (PIFS) et leur rôle dans le codage des images.

En troisième partie, nous avons présenté l'approche de Jacquin (compression d'image par PIFS) basée sur un partitionnement carré, ce qui fait l'objet de l'approche de codage que nous allons appliquer dans notre schéma de compression présenté dans le prochain chapitre.

Introduction

L'inconvénient majeur des méthodes fractales en compression des images réside dans la génération des IFS pendant le processus de codage, très coûteuses en temps de calcul. Cela est essentiellement dû au grand nombre de comparaisons entres les blocs destination r_n et les blocs sources d_n. Une solution consiste à proposer un algorithme non itératif.

L'objectif de cette étude est d'élaborer une méthode hybride pour la compression d'images fixes, associant une approche fractale avec la transformée en ondelettes.

IV.1. Le principe de la méthode

a) Appliquer à l'image originale une transformation en ondelettes discrètes (TOD), telle qu'on obtienne une sous bande lissée (l'approximation A_j) et des sous bandes de hautes fréquences (détails D_j) dont nous avons fixé le niveau de décomposition à 02 (voir figure IV-2).

b) Partitionnement des sous bandes en blocs carrés suivi d'une classification selon la variance en blocs homogènes et hétérogènes.

c) Coder les blocs homogènes suivant les trois différentes directions par leurs valeurs moyenne.

d) Effectuer un codage fractal sur les blocs hétérogènes dans les bandes de détails ($D_j^{(i)}$) issues de l'application de la TOD, suivant les trois différentes directions, respectivement horizontale, verticale et diagonale. Ensuite, procéder au décodage fractal pour reconstituer les sous bandes.

e) Appliquer un codec Hoffman à la sous bande lissée (A_j) de basses fréquences.

f) Restituer l'image, en appliquant une TODI (Transformée en ondelette Discrète Inverse) à toutes les sous bandes.

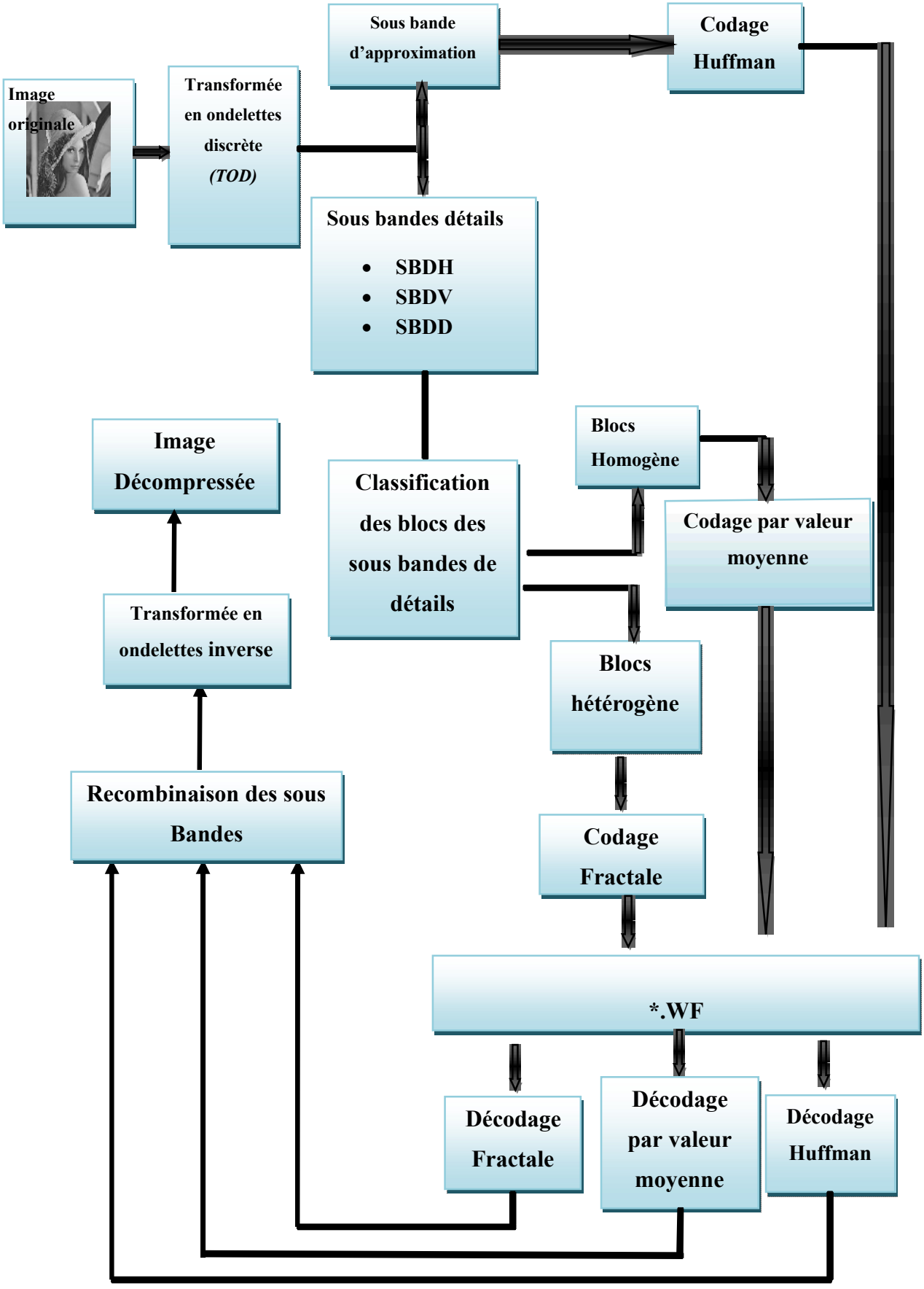


Figure (IV.1): Schéma global de notre méthode de compression

IV.2. La transformée en ondelettes Discrète (TOD)

La décomposition d'une image en sous bandes est réalisée par l'application de l'algorithme de décomposition dyadique de S.Mallat en 2D séparable, adapté, aux filtres utilisés orthogonaux.

Les données à traiter sont sous forme de tableau bidimensionnel (Matrice) contenant les valeurs des pixels de l'image à transformer. La décomposition en sous bandes lissée et détails se fait en trois étapes, comme est illustré à la figure (IV-2)

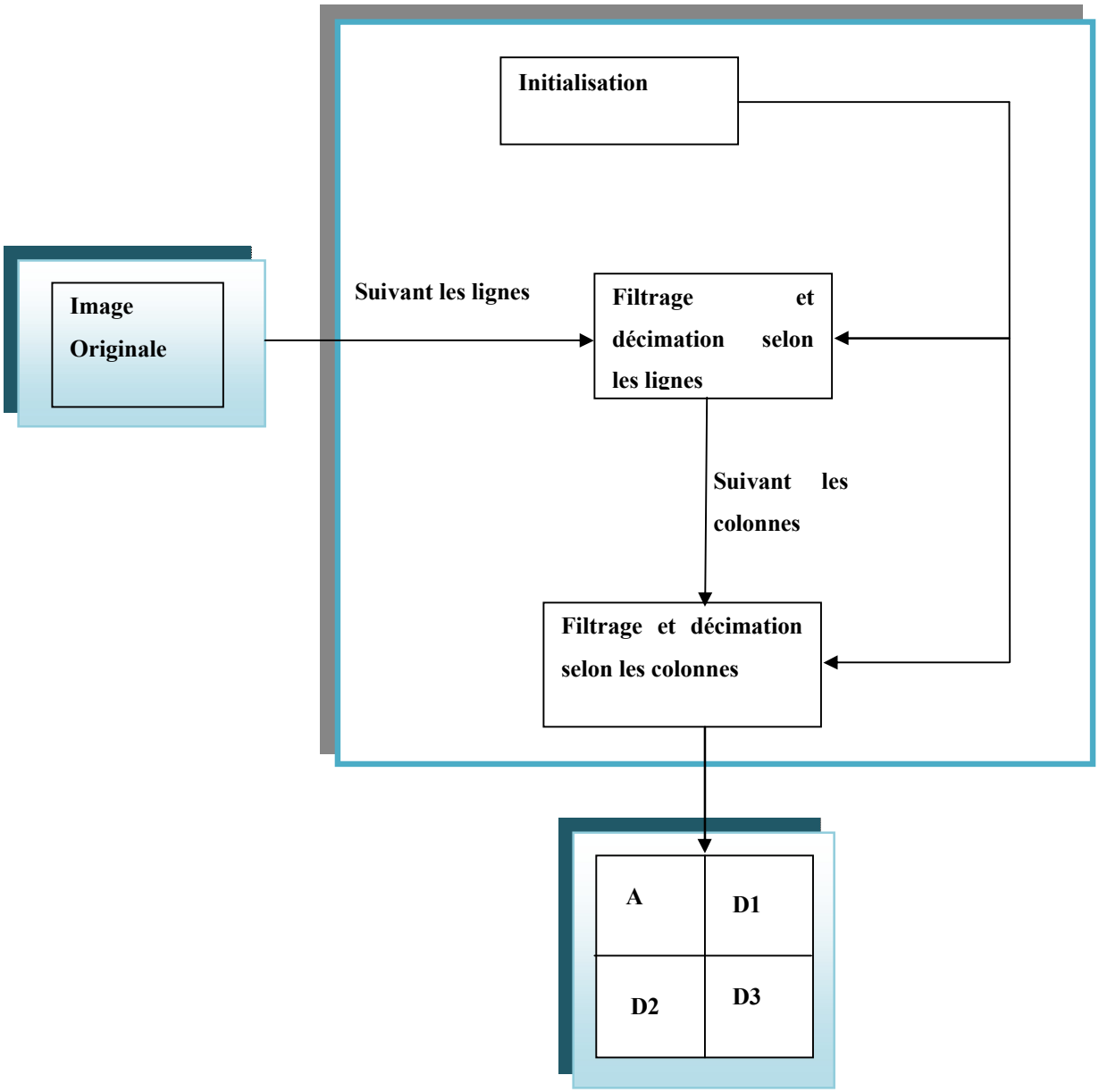


Figure (IV.2) : procédure de décomposition par la TOD directe

IV.2.1. Initialisation

Cette étape permet le choix d'une part, du filtre à utiliser à partir d'un ensemble donnée de filtres orthogonaux, le niveau de décomposition appelé résolution, qui est égale à deux (02) dans le cas de notre méthode.

IV.2.2. Filtrage et décimation selon les lignes

Le filtrage est réalisé par le calcul de la convolution entre les pixels voisins d'une ligne et les coefficients du filtre choisi. Pour réaliser simultanément la décimation, seuls les coefficients des colonnes pairs sont traités. Pour chaque coefficient à traiter, sont appliqués un filtre (H_0): passe-bas et un filtre (G_0) : passe-haut.

IV.2.3. Filtrage et décimation selon les colonnes

Ces opérations sont appliquées de façon analogue sur les coefficients issus de la deuxième étape en transposant lignes et colonnes.

Après avoir effectué ces opérations, on obtient un nouveau tableau de coefficients, divisé en quatre sous images (sous bandes) de même résolution tel que représenter par la figure (IV -3)

	HH		GH
A_j Image		D_j^1 Détails	
	HG		GG
Dj^2 Détails		D^3_j Détails	

Figure (IV.3) : le tableau des coefficients après la TOD direct

La décomposition est réitérée sur la sous bande lissée (approximation) jusqu'à atteindre le nombre de niveau choisi, « dans notre méthode la résolution est fixé à «2».Les sous bandes générées sont à différentes plages de fréquences. Il est à noter que, les coefficients de la sous bande lissée sont de valeurs très importantes, représentant ainsi

les basses fréquences par contre, les coefficients des sous bandes de détails de toutes les directions et à différente résolutions représentent les hautes fréquences.

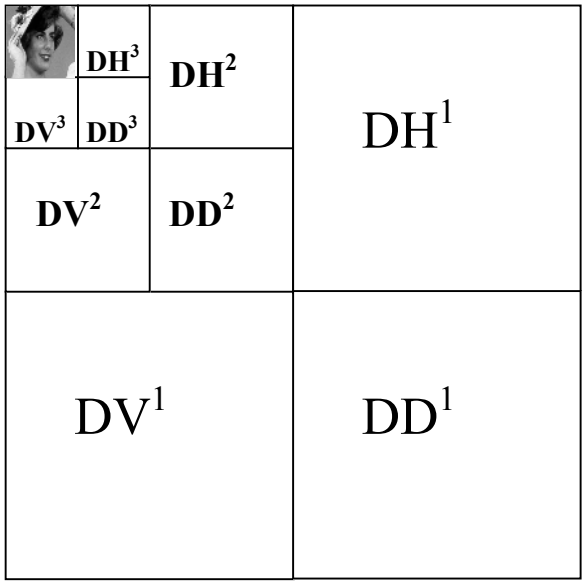


Figure (IV-4) : La décomposition d'une image en résolution 3

IV.3. La procédure du codage fractale

Une fois, nous avons décomposé l'image par la TOD, nous appliquons les fractales pour les sous bandes de détails et pour se faire on va suivre ces étapes :

IV-.3.1. Partitionnement de blocs de détail

Il existe plusieurs types de partitionnement où chaque type a ses avantages et ses inconvénients, nous le choisirons selon sa souplesse et s'il exploite au mieux les auto-similarités entre les parties de l'image et réduit le nombre de transformations à appliquer, parmi ceux-ci on cite : carré, quadtree et triangle. Dans notre méthode, nous utilisons le carré, là où les sous bandes de détails de deuxième résolution serons décomposées en blocs de même taille *(TxT)*, non chevauchés qui la couvrent complètement, qui est appelés blocs destination et en blocs de taille *(2Tx2T)* les sous bandes issues de la première résolution non chevauchés aussi appelés blocs source. Ce partitionnement est simple, mais il est plus adéquat dans la compression par fractales.

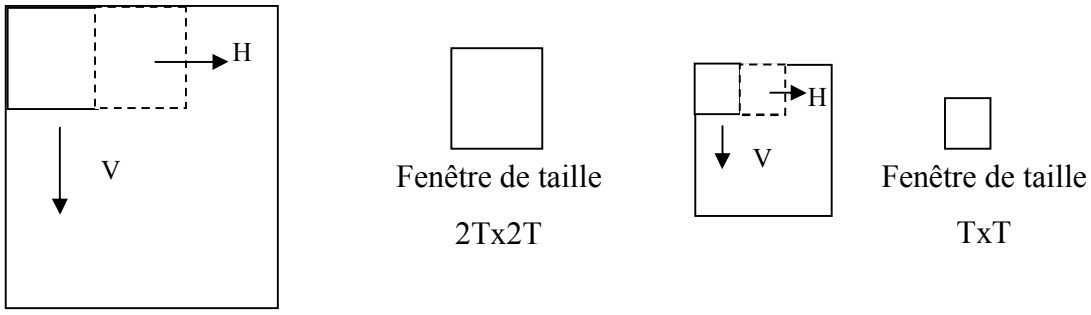


Figure (IV-5) : Procédé de partitionnement des blocs

IV.3.2. Classification

Les blocs seront classifiés selon la variance de leurs niveaux de gris, en blocs homogènes et hétérogènes. Si la variance est inférieure ou égale à une valeur fixée, appelée seuil, le bloc est homogène sinon il est hétérogène. Le seuil séparant les deux types de blocs est obtenu en calculant la moyenne des variances.

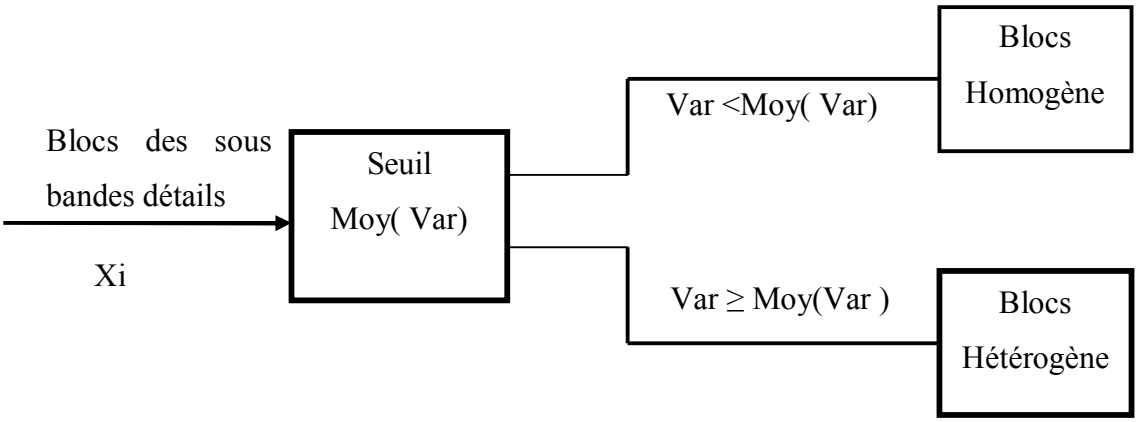


Figure (IV-6) : Schéma de classification

IV.3.3. Codage et décodage des blocs Homogènes :

L'approximation des blocs homogènes se fait en mémorisant seulement leurs valeurs moyennes. Le but de cette opération est d'avoir un taux de compression assez élevé tout en réduisant le temps de calcul. L'étape de décodage se fera par interpolation des blocs par leurs valeurs moyennes.

Après l'étape de classification, seuls les blocs hétérogènes seront codés par les fractales.

IV.3.4. Codage des blocs hétérogènes par des fractales

Pour générer un code IFS, nous proposons un algorithme qui code les sous bandes de hautes fréquences (détails) par approximation fractale basée sur la méthode de Jacquin, qui est plus flexible que les autres algorithmes conventionnels utilisant les transformations contractantes de l'image globale. C'est une méthode non itérative qui requis une procédure d'apprentissage afin de mieux approximer les hautes fréquences (détails) de l'image.

Le codage fractale dans le domaine des ondelettes s'applique sur les coefficients d'ondelettes (D_j^h, D_j^v, D_j^d) des sous bandes détails après avoir partitionner ces dernières en blocs destination r_i de taille TxT et source d_i de taille 2Tx2T (voir figure IV-3). Le codage est réalisé par les transformations contractantes appliquées aux blocs source hétérogènes afin d'être mieux approximer par des blocs destination. Bien que le codage fractal dans le domaine de l'image ait le désavantage de satisfaire le critère de contractivité (ressemblance) avec un grand nombre de blocs recherchés. Mais ce n'est plus le cas dans le domaine des ondelettes, du moment que la contractivité est recherchée uniquement dans les sous bandes de même direction.

Génération du code IFS

Le codage par des fractales consiste à déterminer une transformée affine et contractante w_i qui en l'appliquant aux blocs 'source' D_m, donne la meilleure approximation possible des blocs 'destination' R_n (figure IV-8).

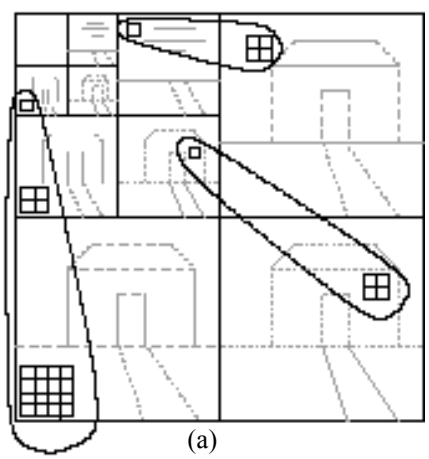
- **Décimation**

La décimation consiste à réduire la taille des blocs source hétérogènes qui sont de taille $2T \times 2T$ en blocs de taille $T \times T$. Cela est réalisé en remplaçant chaque sous bloc de taille *8x8* du bloc considéré par sa moyenne selon la relation suivante :

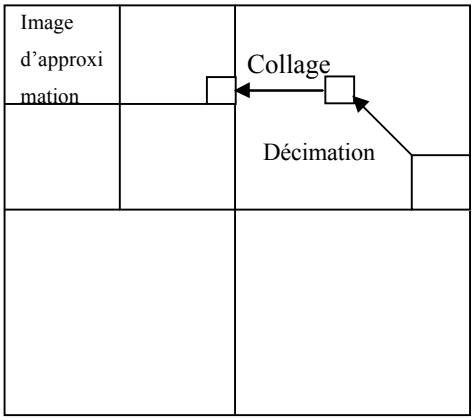
$$\tilde{D}_m(i,j) = \frac{1}{4}[D_m(i,j) + D_m(i,j+1) + D_m(i+1,j) + D_m(i+1,j+1)] \qquad (4-1)$$

$\tilde{D}_m(i,j)$: est la valeur du pixel de la position (i, j) du bloc D_m

Notons que les sous blocs ne sont pas chevauchés.



(a)



(b)

Figure (IV-7) : (a) Hiérarchie des mises en correspondance des blocs source et blocs destination dans une décomposition en ondelettes

(b) Collage d'un bloc source sur un bloc destination

La procédure du collage d'un bloc source issu de la résolution 2^{-j} sur un bloc destination issu de la résolution $2^{-(j+1)}$ se fait à travers un ensemble de transformation contractives (w_i) massiques donnée par la relation (4-2).

$$W_i(d_i) = \alpha_i * G_i \tilde{d}_i \qquad (4.2)$$

où : G_i : les huit fonctions d'isométrie utilisées,

α_i : le facteur d'échelle (modification du contraste) du bloc source $\tilde{d}_i$

Pour trouver le bloc d_i le plus proche du bloc r_i, nous cherchons pour chacun des blocs destination r_i le bloc source d_i qui, après transformation affine contractante, permet de minimiser l'erreur quadratique moyenne h_i obtenue entre r_i et le bloc transformé. Cette erreur s'écrit :

$$h_i = \sum_{k=0}^{L-1} \sum_{l=0}^{L-1} |r_i(k,l) - \alpha_i * G_i[d_i(k,l)]|^2 \qquad (4.3)$$

Où : r_i et d_i représentent respectivement le bloc destination et le bloc source.

L : représente la taille du bloc destination et source décimé

G_i : représente l'une des huit fonctions d'isométries utilisées.

α_i : représente le facteur d'échelle (modification de contraste) du bloc source de position (k,l) avec $\alpha \in [0 , 1]$.

Les transformations massiques utilisées (isométries et modification du contraste) sont définies comme suit :

- **l'isométrie** : il existe huit manières différentes de coller le contenu du bloc source à celui de bloc destination le plus similaire : les quatre rotations par les ongles 0^0, 90^0, 180^0, 270^0 et les quatre symétries pour chaque rotation par rapport à l'axe vertical selon la figure (IV-9)

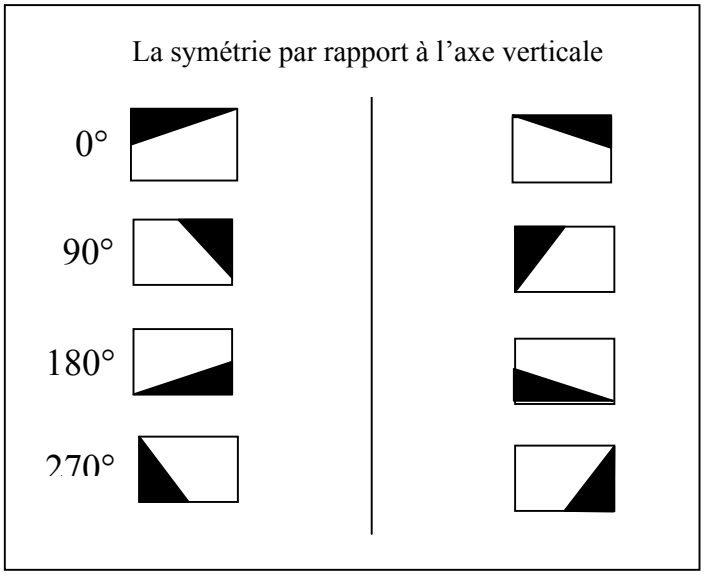


Figure (IV-9) : les huit cas d'orientation d'un bloc

- **Modification du contraste** :

Elle agit seulement sur les valeurs des niveaux de gris des pixels du bloc d_i pour être similaire au bloc destination r_i. Cette transformation w consiste à multiplier les coefficients du bloc source par un facteur α appelé facteur d'échelle, donné par l'équation :

$$W_i(d_i(k,l)) = \alpha * d_i(k,l) : \text{avec } \alpha \in [0, 1] \qquad (4\text{-}4)$$

Soient $x = (x_1, x_2, \ldots x_n)$ l'ensemble ordonné des coefficients d'un bloc destination r_i et $y=(y_1, y_2, \ldots .y_n)$ l'ensemble ordonné des coefficients d'un bloc source transformé par une transformation d'isométrie $G_i(d_i)$, la valeur optimale du facteur d'échelle α_i est obtenue par la relation (4-5)

$$\alpha_i = \frac{\sum_{i=0}^{n-1} x_i * y_i}{\sum_{i=0}^{n-1} y_i^2} \qquad (4\text{-}5)$$

La valeur de α_i peut prendre huit (8) valeurs différentes selon l'orientation considérée

IV.4. Décodage fractale

Le diagramme de décodage représenté dans la figure (IV-10) permet de générer les blocs source par les paramètres fractals reçus par le décodeur, afin de reconstituer les sous bandes détails. Notons que l'algorithme proposé non itératif.

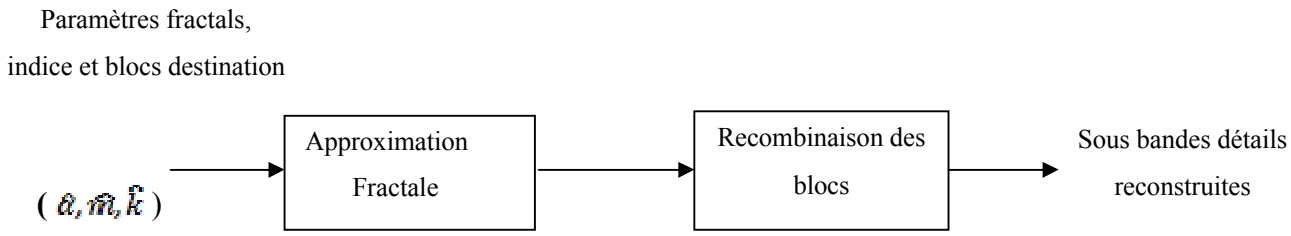


Figure IV-10 : diagramme de décodage

IV .5. Codage et décodage de la sous bande lissée selon le codage d'Huffman

L'algorithme de HUFFMAN est la méthode de codage la plus répondue. Cet algorithme consiste à crée des codes de longueurs variables sur un nombre entier, d'autant plus de bits que la probabilité est faible, les codes de HUFFMAN sont définis de façon ascendante, en commençant par les feuilles de l'arbre (par les derniers chiffres du code) et en remontant vers la racine.

❖ **Etape de codage :**

Le processus de codage est subdivisé en (08) étapes :

- Ouverture du fichier à compresser en lecture comme étant un fichier de caractères.
- Lecture du fichier caractère par caractère pour calculer la fréquence.

- A partir des deux symboles présentant la fréquence la plus faible, un nœud est généré, il lui est affecté un poids égal à la somme des fréquences des deux symboles.
- Le nœud crée remplace, désormais les deux symboles dans la suite du processus. A partir de ces derniers sont affectés respectivement les chiffres binaires 0 pour le plus fréquent et 1 pour le plus rare.

- La même procédure est répétée en considérant les deux symboles ou nœuds de poids le plus faible. Elle est renouvelée tant qu'il reste plus d'un nœud vacant.
- De cette manière, on aura construit l'arbre de HUFFMAN et la table de correspondance est remplie.
- On effectue par la suite une relecture du fichier caractère par caractère et remplacement de chaque caractère par son nouveau code trouvé qui est dans la table de correspondance.

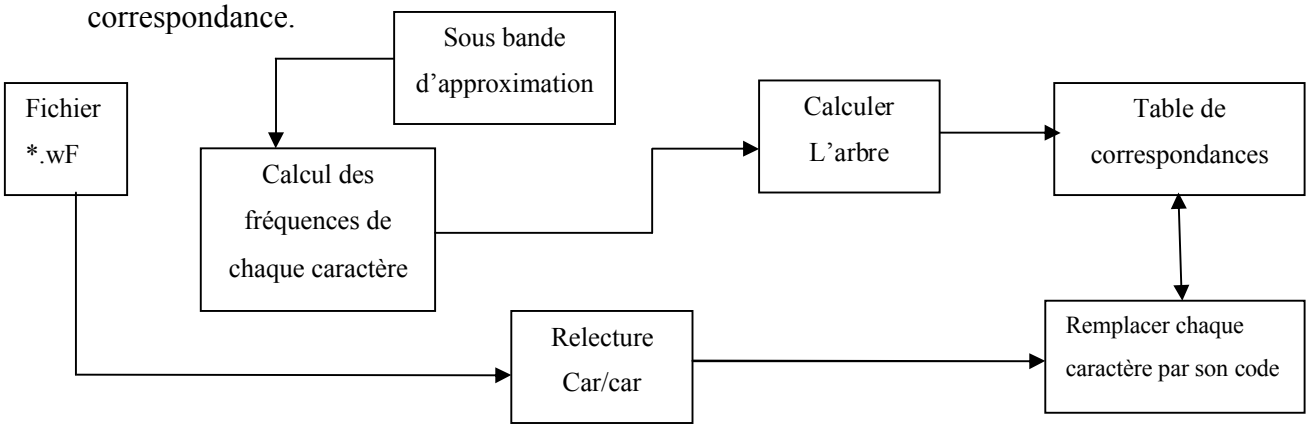


Figure IV -11: schéma du codage HUFFMAN

L'arbre d'Huffman est constitué à partir des trois tables (feuille, arbre et alphabet).

Exemple : soit F contenant les caractères a, b et c

Le processus de codage se déroule selon les étapes suivantes :

1) Calcul des fréquences d'apparition des symboles a, b et c dans le fichier F et les trier dans la table alphabet. (le fichier F contient les caractères)

2) Création de l'arbre de HUFFMAN : en prenant les deux premières fréquences d'Alphabet correspondant à a et b, on construit le sous arbre A_0. Le nœud A_0 ainsi généré est inséré dans la table triée alphabet pour qu'il soit postulant pour la constriction du sous arbre prochain. Le processus est itéré jusqu'à ce que fré (A_i étant le $i^{ème}$ sous arbre). A la fin du processus, on aura les tables alphabet, arbre et feuille.

3) Calcul des codes : les deux tables arbre et feuille servent à retrouver les nouveaux codes de chaque caractère.

A la fin du codage HUFFMAN, le fichier résultat sera stocké dans la mémoire sous le format compressé WF.

❖ **Étape de décodage HUFFMAN :**

Ce processus de décodage va nous permettre de restituer le fichier F à partir des informations stockées dans le fichier d'origine à partir des informations stockée dans le format compressé, et cela suivant les étapes suivantes :

1. Ouverture du fichier compressé*.WF en lecture
2. Extraction de taille_arbre, taille_feuille, arbre feuille et nom fichier
3. Ouverture d'un fichier en écriture en utilisant nom fichier
4. Initialisation d'un pointeur à la racine de la table arbre
5. Lecture du fichier bit à bit, se déplacer vers le fils gauche de l'arbre si bit =0 et vers le fils droit si bit=1 jusqu'à retrouver la nature du nœud = « f », récupérer le numéro de la feuille indiqué par le fils droit et le fils gauche, ou bien trouver nature du nœud = « m » et récupérer le numéro de la feuille indiqué par le fils gauche
6. Le numéro récupérer à l'étape précédente nous indique un emplacement dans la table feuille, cela permet de lire le caractère correspondant au code parcouru
7. Ecriture du caractère lu dans le fichier de sorte les étapes 4, 5, 6 et 7sont itérées tant que la fin du fichier compressé n'est pas atteinte, ainsi on obtient le fichier F sans perte d'information.

IV.6. Transformation en ondelettes Discrète Inverse (TODI)

C'est la phase de synthèse où l'image est reconstruite à partir du tableau des valeurs des coefficients approximés formés des sous bandes lissée et de détails, par l'application de la transformée en ondelettes discrète inverse. Le traitement suit le cheminement inverse de celui suivi pour la décomposition. La reconstruction est exécutée en trois étapes, comme il est illustré à la figure (IV-12)

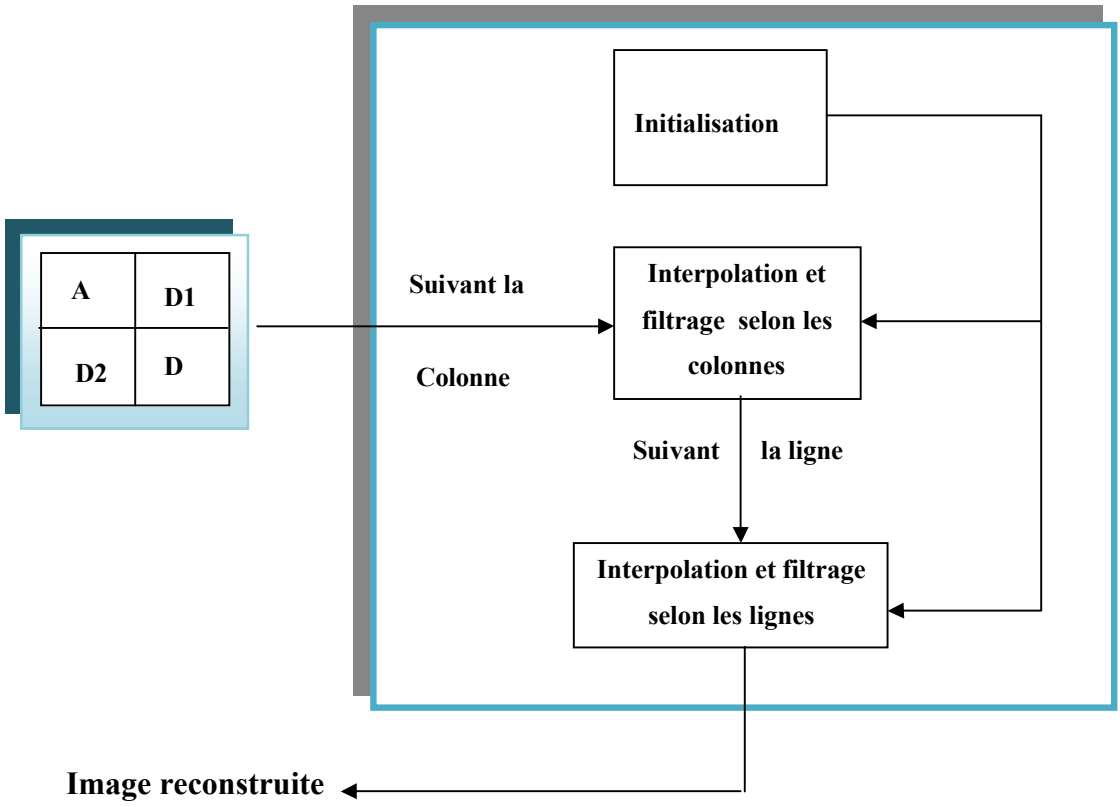


Figure IV-12: schéma de la TODI

- **Initialisation**

Elle consiste à charger les données des sous bandes reconstituées, les filtres inverses H_1 et G_1 correspondants à ceux utilisés à la décomposition et la valeur de la résolution appliquée.

- **Interpolation et filtrage selon les colonnes :**

Les coefficients d'approximation et de détails sont interpolés puis convolués avec les filtres unidimensionnels de reconstruction passe bas et passe haut respectivement. Les résultats des deux convolutions sont ensuite multipliés par la valeur deux puis additionnés.

- **Interpolation et filtrage selon les lignes :**

Les coefficients issus de la deuxième étape sont traités de manière analogue en procédant sur les lignes.

Le traitement est répété jusqu'à atteindre la résolution finale désirée. A la fin de ces procédures, nous reconstituons un tableau de valeurs approximatives aux valeurs des

pixels de l'image originale, qui n'est qu'une représentation de l'image décompressée dans le domaine spatiale.

Conclusion :

Dans ce chapitre, nous avons proposé un schéma de compression des images fixes en régions classifiées basé sur le codage fractale en multirésolution et la transformée en ondelettes discrètes.

Le codage fractale est appliqué pour approximer les sous bandes détails. Notre technique développée est non itératif d'où elle est rapide. Les résultats de notre méthode seront illustrés dans le chapitre suivant.

Introduction

L'analyse par les tests est le mode le plus performant pour évaluer une application informatique. Elle permet d'estimer son aptitude à réaliser les tâches désirées et à trouver les failles rencontrées au cours de son implémentation.

Dans ce chapitre, nous essayerons de présenter les performances de l'approche adoptée en jumelant les ondelettes et les fractales, appliquées à la compression d'images fixes en 256 niveaux de gris.

Nous avons fixé le niveau de décomposition en ondelettes à deux (la deuxième résolution).

V.1. Paramètres de performance

Afin d'évaluer les résultats obtenus, nous considérons les paramètres suivants :

- L'erreur quadratique moyenne (MSE) :

$$MSE = \frac{1}{M * N} \sum_{i=0}^{M} \sum_{j=0}^{N} \left(p(i,j) - p'(i,j)\right)^2$$

- Le rapport signal sur le bruit crête :

$$PSNR = 10 \log_{10} \frac{(255)^2}{MSE}$$

- Taux de compression (Tc)

$$Rc = \frac{\text{Nbre de bits de l'image originale}}{\text{Nbre de bits de l'image compressée}}$$

Le taux de compression en pourcentage est calculé par la relation suivante :

$$Tc = \left(1 - \frac{1}{Rc}\right) \times 100$$

- Le temps de compression : c'est le temps que va mettre le processus de compression.

Le processus de la compression a été implémenté sur un micro-ordinateur Pentium (R) IV de fréquence de travail égale à 2 Ghz, possédant une mémoire centrale (RAM) de 128 MO.
La programmation des différentes procédures a été mise en œuvre en utilisant MATLAB 7 comme langage de programmation.

V.2. Description du logiciel (MATLAB)

Matlab est un logiciel pour effectuer des calculs numériques. Il a été conçu pour faciliter le traitement des matrices.

Intérêt :

- Programmation infiniment plus rapide pour le calcul et pour l'affichage.
- Une librairie très riche.
- Possibilité d'inclure un programme en C/C^{++}.
- Langage interprété : pas de compilation donc pas d'attente pour compiler.
- Code facile à comprendre et très lisible.
- Installation sans problème. Il est possible également d'ajouter facilement des outils supplémentaires appelés Toolbox. Certaines Toolbox sont gratuites et d'autres payantes.
- La documentation est exemplaire, d'une part on peut avoir accès facilement à une aide de la console en tapant HELP plus le nom de la fonction qui nous intéresser, il est également possible d'avoir accès à une aide en lançant la commande doc. Enfin, le langage Matlab et accompagné de plusieurs documents en PDF.

L'avantage de Matlab par rapport à certains langages réside dans le fait que beaucoup de fonction sont préprogrammé (le nombre de fonction préprogrammé va dépendre des Toolbox installés).

Inconvénients :

- Vitesse de calcul moins rapide qu'en C/C^{++}.
- Payant
- Application auto-exécutable peu pratique.

Afin d'évaluer les performances de notre méthode décrite dans le chapitre précédent, nous avons utilisé deux types d'images:

- L'image 'Lena' qui est un portrait de référence composée de diverses textures, dont des textures très fines comme les yeux et les cheveux.
- L'image 'Bobine' est une image qui contient des régions (blocs) homogène plus que des régions (blocs) hétérogène.

Figure (V-1) : image Lena originale

Figure (V-2) : image Bobine originale

V.3. Choix de filtre approprié :

Afin de choisir le meilleur filtre qui nous servira à la décorélation de l'image test, nous allons tester l'influence des différents filtres sur la qualité de l'image.

Les différents résultats sur les images «Lena» et «Bobine» sont inscrits dans les tableaux suivants :

	Image lena	**Image bobine**
Filtres	**PSNR**	**PSNR**
HAAR	29,50	28,80
DAUBECHIES 4	29,87	28,77
DAUBECHIES 8	29,86	28,30
DAUBECHIES10	29,75	28,54
DAUBECHIES12	29,30	28,60
DAUBECHIES14	29,07	28,48
DAUBECHIES16	29,73	28,75
DAUBECHIES20	29,95	27,95

Tableau A : Résultat de l'application des différents filtres sur l'image«Lena» et « Bobine »

V.4. Résumé des meilleurs filtres :

A la lecture du tableau précédent, nous pouvons extrapoler les meilleurs filtres en fonction des valeurs du PSNR, les résultats du meilleur filtre pour chaque image sont représentés dans le tableau suivant :

Images	**Filtre**	**PSNR**
Lena	Daub_20	29,95
bobine	Daub_1	28,80

Tableau B : le meilleur filtre pour chaque image.

V.5. Les résultats obtenus

Afin de vérifier l'adaptation de notre technique de compression sur ces images qui constituent un volume de données, nous présentons les résultats obtenus dans diverses situations en faisant varier au maximum la taille des blocs source/destination c'est-à-dire le taux de compression. Le but de notre méthode est de choisir un bon rapport de PSNR sur le taux de compression pour le stockage de ces images.

Le tableau C montre les différents résultats obtenus de notre application sur les deux images Lena et Bobine.

Taille des blocs source	**Taille Des blocs destination**	**Lena**			**Bobine**		
		PSNR (db)	**TC (%)**	**Tps (s)**	**PSNR (db)**	**TC (%)**	**Tps (s)**
4×4	**2x2**	30,00	70,79	07	29,85	70,50	08
8×8	**4x4**	29,95	74,25	02	29,80	74,11	03
16×16	**8x8**	29,80	75,11	01	29,75	74,99	02
32×32	**16x16**	29,79	75,33	01	29,74	75,33	02

Tableau C : Résultats avec variation de la taille des blocs pour les images Lena et Bobine

V.6. Représentations graphiques :

- **Graphe 1 : variation du PSNR en fonction de la taille des blocs source.**

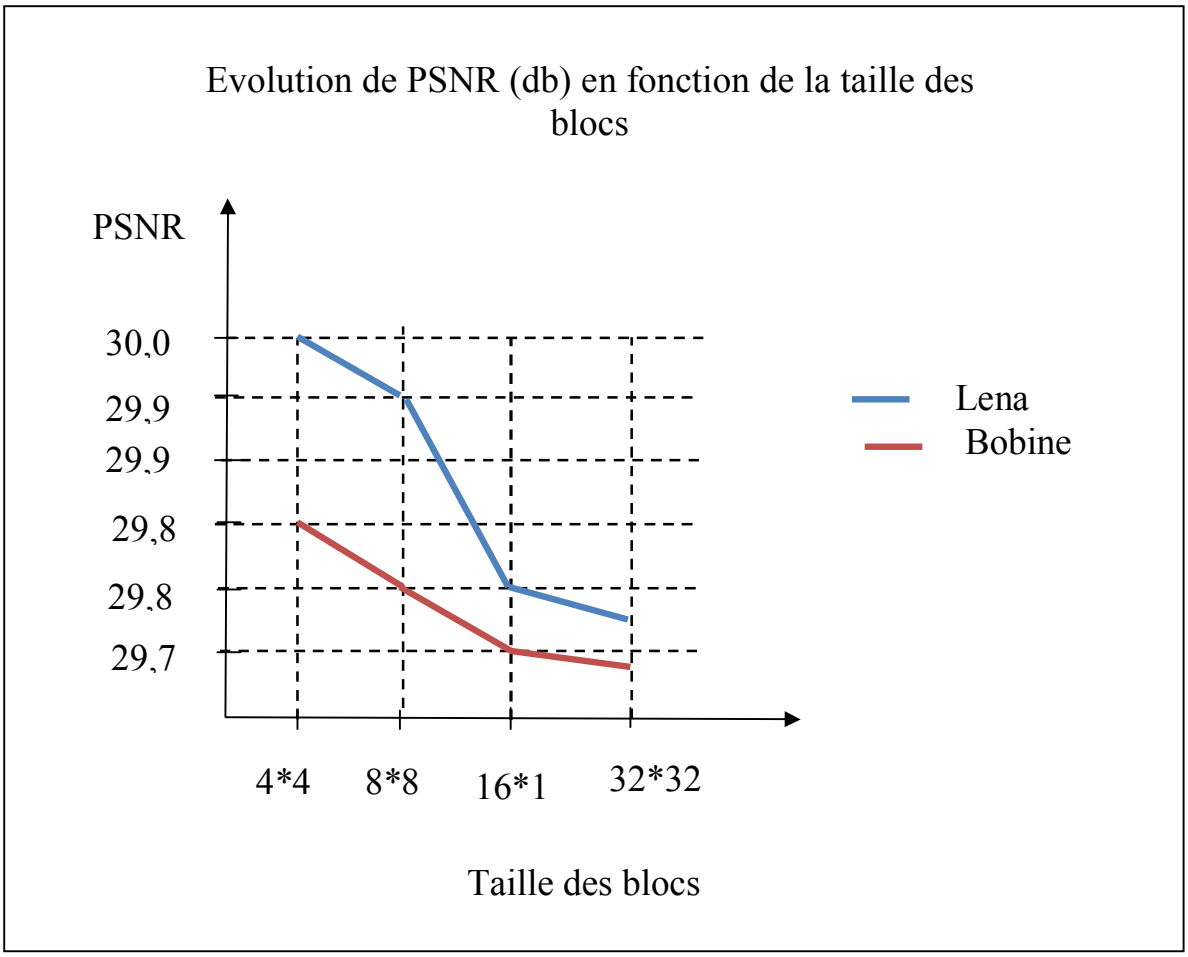


Figure (V-3) : variation du PSNR en fonction de la taille des blocs source

- **Graphe 2 : variation du taux de compression (Tc) en fonction de la taille des blocs source.**

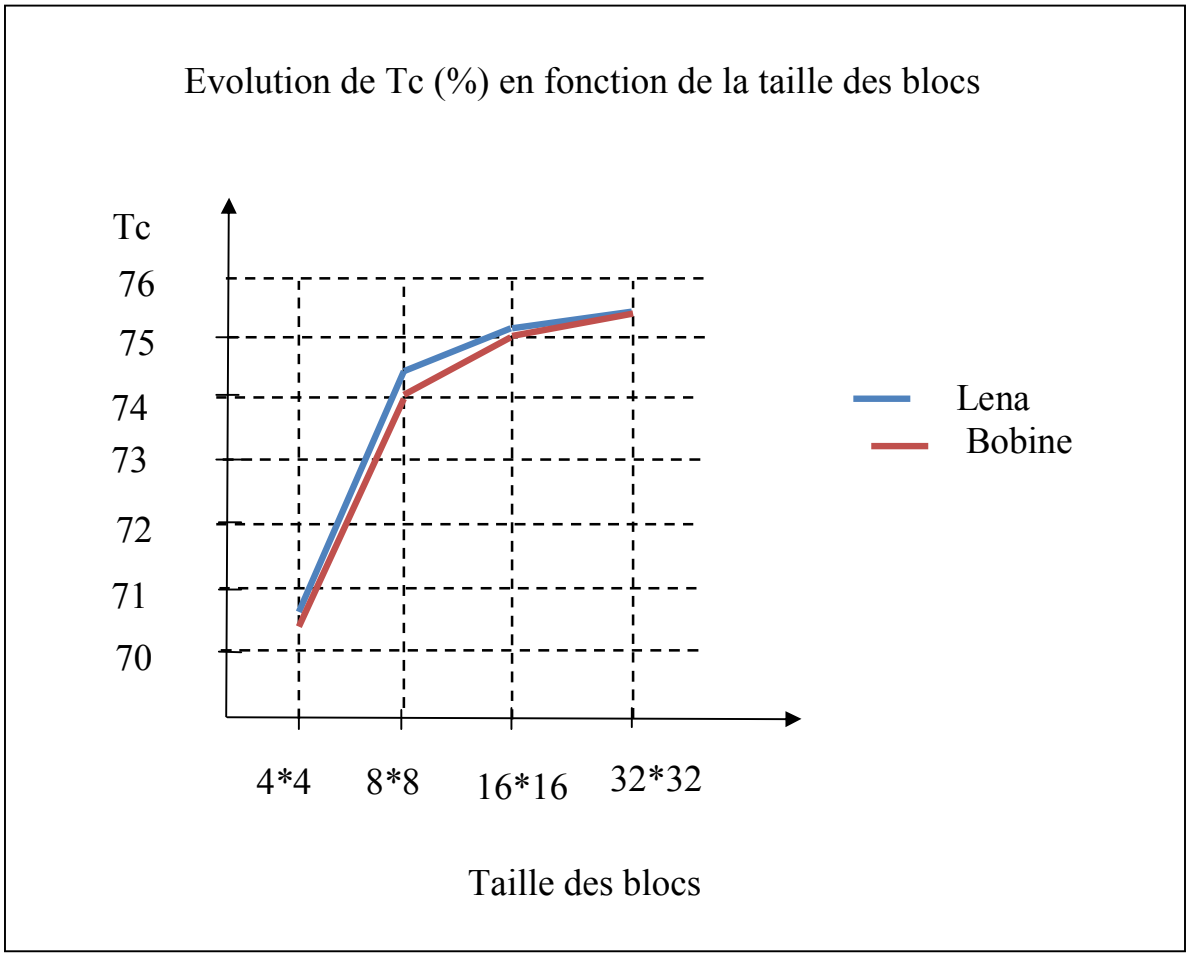


Figure (V-4) : variation du taux de compression (Tc) en fonction de la taille des blocs source

- **Graphe 3 : variation du PSNR en fonction du taux de compression (Tc).**

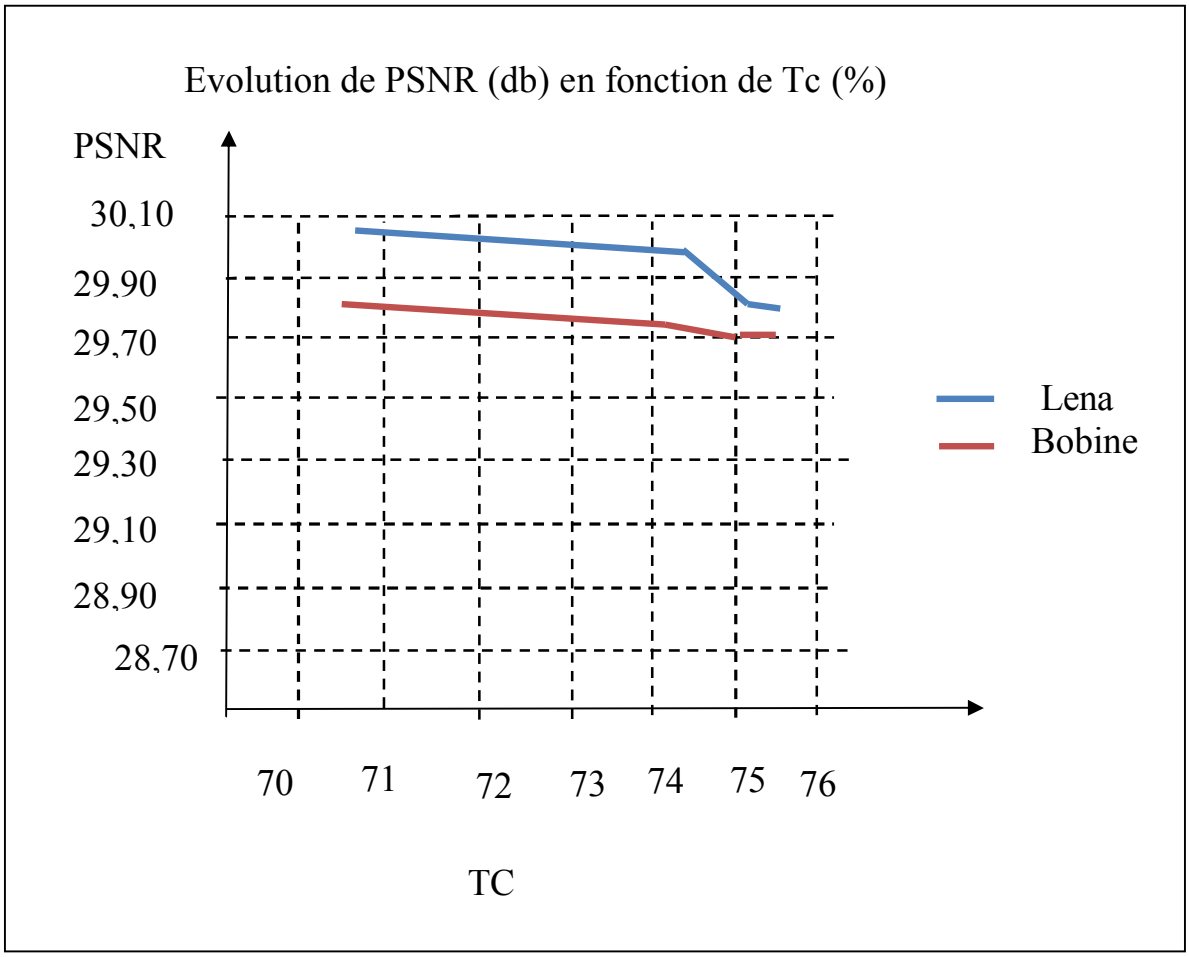


Figure (V-5) : variation du PSNR en fonction du taux de compression (Tc)

V.7. Exemples d'images restituées :

Les figures ci-dessous nous montrent des exemples d'images décompressés :

Figure (V-6) : Image originale Lena

Figure (V-7) : Image restituée PSNR=29.95, T_C=74.25, Taille de bloc source =8*8

Figure (V-8: Image originale Bobine

Figure (V-9): Image restituée PSNR=29.80, T_C=74.11, Taille de bloc source =8*8

V.8. Interprétation des résultats :

D'après les différent graphes tracés à partir des résultats obtenus de l'application de notre méthode, nous avons fait les constations suivantes :

- le PSNR dépend de filtre choisi ainsi que la taille des blocs. Il décroît avec l'augmentation de la taille des blocs. En contre partie, le temps d'exécution augmente légèrement de l'ordre de 1 ms avec la taille des blocs.

- Le taux de compression est proportionnel à la taille des blocs à traiter, plus le taux de compression augmente, plus le PSNR diminue et la qualité de l'image restituée est dégradée.
- Le temps d'exécution de notre application est nettement réduit comparant aux autres méthodes basé sur les fractales.

Nous pouvons aussi noter les remarques suivantes concernant les images restituées pour une taille de 8*8 des blocs source :

- Les images représentées dans les figures.V.7 et V.9, laissent apparaître des artefacts (les frontières des blocs de taille 8*8 sont visibles) et un lissage, qui s'accentuent en augmentant la taille des blocs.
- La dégradation est essentiellement plus importante dans les régions contenant des contours et celles texturées que dans celles homogènes.
- Notons une perte de détails fins et un effet de lissage tel qu'au niveau des cheveux et le chapeau en augmentant la taille des blocs; en revanche, les régions homogènes sont fidèlement reconstituées. Les images apparaissent plus ternes en augmentant la taille des blocs et moins dynamiques que l'image originale.

Néanmoins les artefacts sont réduits dans le cas de notre méthode de compression basée sur le codage fractal inter-directionnel des sous bandes issues de la décomposition en ondelettes, de telle manière que l'effet de bloc est considérablement atténué, laissant ainsi apparaître une image appréciable même à taux de compression élevé.

La méthode de compression que nous avons développée a permis d'obtenir des résultats intéressants comparant aux autres méthodes considérées performantes, comme le cas de la norme JPEG [3] et celle utilisée dans [15].

Conclusion :

L'utilisation de notre méthode a permis d'obtenir des résultats satisfaisant du point de vue taux de compression (Tc), qualité visuelle de l'image reconstituée (PSNR).

Nous pouvons conclure que le taux de compression influe d'une manière importante sur la qualité visuelle de l'image. Ce dernier dépend essentiellement, d'une part du filtre choisit, et d'autre part de la taille des blocs, il s'agit donc d'un compromis à effectuer entre une bonne qualité visuelle et un taux de compression assez élevé avec un temps de calcul réduit.

Conclusion générale

Nous avons présenté dans ce mémoire une méthode hybride de compression d'images fixe en régions classifiées, associant les ondelettes et les fractales.

Nous avons jugé utile de présenter des généralités qui se referont aux différentes méthodes de compression pour situer notre travail. Nous avons ensuite exposé les outils nécessaires à notre schéma de compression par une présentation de la théorie des ondelettes et l'algorithme de S.MALLAT ainsi que la théorie des fractales.

Le codage fractale est appliqué sur les blocs hétérogènes et le codage par valeur moyenne pour les blocs homogènes issues de la partition des sous bandes de détails de la décomposition en ondelettes de l'image originale. Et un codage réversible d'Huffman pour la sous bande lissée.

La méthode développée a permis d'obtenir des images reconstruites de qualité perceptuelle acceptable, des taux de compression élevés et un temps de codage très réduit comparativement aux méthodes considérées classiques 'JPEG et Codage fractale dans le domaine spatiale de l'image'.

Néanmoins, nous pensons pouvoir dans l'avenir apporter une solution à l'apparition des artefacts lors de l'augmentation de la taille des blocs et aussi vis-à-vis du niveau de la résolution. Un codage plus adapté aux différentes régions des sous bandes segmentées, pourrait effectuer un meilleur codage des détails très fins des zones contenants des contours importants, et des textures, et devrait éviter l'apparition de toute perturbation.

Diverses améliorations peuvent être aussi apportées à notre schéma de compression, notamment dans les étapes de décomposition et de codage. Dans l'étape de décomposition, l'utilisation des filtres biorthogonaux permettrait une amélioration dans la qualité perceptuelle ainsi une réduction du temps de calcul. Et dans l'étape de codage, un codage fractale basé sur un partitionnement en HV, en triangulaire ou en polygone, qui sont plus adaptés aux régions contenant des contours, permettra de déduire les artefacts.

Bibliographie

[1] Ameur, S., A.Adane, M.Lahdir: "Compression d'images MÉTÉOSAT en sous bandes par transformation discrète en cosinus et quantification vectorielle", Télédétection, vol.2, n°4, *pp.255-266* ; 2002.

[2] Davoine, F : "Compression d'images par fractales basée sur la triangulation de Delaunay", Thèse de doctorat, Institut Polytechnique de Grenoble, France ; 1995.

[3] Guillois, J.P: "Techniques de compression des images", Edition Hermes, Paris France ; 1996.

[4] Lahdir, M, S.Ameur, A.Adane: "Algorithme non itératif, basé sur les ondelettes biorthogonales et les fractales, pour la compression d'images satellitaires", Télédétection, Vol 6 N° 4 ; 2007.

[5] Moreau, N : "Techniques de compression des signaux", Edition Masson, Paris France ; 1995.

[6] Sadoun, N., S. Ameur : "Application d'un codage fractale en multi-échelles dans le domaine de la transformée en ondelettes biorthogonales pour la compression d'images MSG" 3rd International Symposium on Images/Video Communications over fixed mobile networks ; septembre 13-15, 2006, Hammamet, Tunisie ; 2006.

[7] Burel, Gilles : « introduction en traitement d'images : simulation » ; 1995.

[8] G.Blanchet « signaux et images sous Matlab : Méthode, application et exercices corrigé », 2000.

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

[10] :BELAMIRI.S, SAIDI.T, MEDDANE.R « Compression d'image par paquets d'ondelettes » mémoire d'ingénieur d'état en électronique, UMMTO 2003.

[11] :DRICI.Z, SELMANI.T, YAHIATENE.M « Application des ondelettes biorthogonales pour le codage des images numériques », mémoire d'ingénieur d'état en électronique, UMMTO 2002.

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

[13] :MOUZARINE.H « Codage des images numérique par TCD et fractal », mémoire d'ingénieur d'état en électronique, UMMTO 2002.

[14] : KACI.N, MOUKHTARI.A « Application des ondelettes et des fractales pour la compression des images numérique », mémoire d'ingénieur d'état en électronique, UMMTO 2001.

[15] : AKROUR.L « Compression d'images par fractale dans le domaine de la DCT », mémoire d'ingénieur d'état en électronique, UMMTO 2003.

[16] : S.BERKACHE « Codage d'image en sous bande », mémoire d'ingénieur d'état en électronique, UMMTO 2004.

[17] : NADOUR.L « Application des fractales à la compression de séquences d'image basée sur les quadtrees », mémoire d'ingénieur d'état en électronique, UMMTO 2002.

[18] : KHELFAOUI.B « Application des méthodes réversibles pour le codage des images numériques », mémoire d'ingénieur d'état en électronique, UMMTO 2004.

[19] :HAMRENE.N , DJILALI.I, HAMOUDI.I « codage d'image en sous bande par fractales appliqué aux images médicales », mémoire d'ingénieur d'état en électronique, UMMTO 2005

[20] : http://www.médiathèque.image.fr/collection-électroniques, « Etude bibliographique des méthodes de compression d'images.htm », 1996.