REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR

ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE DE BATNA

FACULTE DES SCIENCES DE L'INGENIEUR

MEMOIRE

Présenté au

DEPARTEMENT D'ELECTRONIQUE

Pour l'obtention du diplôme de

MAGISTER en micro ondes pour télécommunications

Par

DOUAK Fouzi

Ingénieur d'Etat, Institut d'Electronique-Université de Batna

Intitulé

Reconstruction des images compressées en utilisant les réseaux de neurones artificiels et la DCT

Soutenue le :...../...../2008

Devant le jury constitué de :

Dr. Ahmed Louchene	M.C. U. Batna	Président
Dr. Nabil Benoudjit	M.C. U. Batna	Rapporteur
Dr. Redha Benzid	M.C. U. M'sila	Co-Rapporteur
Dr. Djamel Saigaa	M.C. U. Biskra	Examinateur
Dr. Lamir Saidi	M.C. U. Batna	Examinateur

Je dédie ce travail à :

Ma mère,

Mon père,

Mes frères et ma sœur,

Tous mes amis sans exception.

Remerciements et gratitude

Ma haute gratitude, mes profonds respects et mes sincères remerciements et reconnaissances à mon rapporteur Monsieur ***Nabil BENOUDJIT****, maître de conférence à l'université de Batna qui m'a guidé avec grande patience tout au long de l'élaboration de ce travail et pour ses aides précieuses qui ont judicieusement éclairé mon chemin vers l'aboutissement et la réussite de la concrétisation de mon mémoire.*

Je tiens à exprimer mes plus vifs remerciements à mon Co-rapporteur Monsieur ***Redha BENZID****, maître de conférence à l'université de M'sila pour ses aides, son accompagnent et ses éclairages techniques. Je lui dois les remerciements les plus sincères.*

Je profite de l'occasion de la présentation de ce travail pour exprimer Mes vifs remerciements à Monsieur ***Ahmed LOUCHENE****, maître de conférence à l'université de Batna, pour avoir accepté de présider le jury de soutenance.*

Je tiens également à présenter ma profonde gratitude à Monsieur ***Djamel SAIGAA****, maître de conférence à l'université de Biskra, qui a accepté d'examiner mon travail malgré son éloignement et ces charges.*

J'exprime également mes remerciements à Monsieur ***Lamir SAIDI****, maître de conférence à l'université de Batna, d'avoir accepté de juger ce modeste travail.*

A cette occasion ma gratitude va à titre particulier à tous mes enseignants qui ont donné le meilleur d'eux-mêmes durant le cycle de ma formation. Qu'ils en soient chaleureusement remerciés en amont et en aval pour leur généreux don de savoir sans lequel le privilège du savoir faire serait pour moi inaccessible.

Mr DOUAK Fouzi

Table des matières

Introduction Générale 1

Chapitre I

Généralités sur la compression d'image

I.1.Introduction 5
I.2.Définition d'une image et des types d'images 5
I.3.Changement d'espace de couleur 6
I.4.Compression d'image et critères d'évaluation 8
I.5.Image médicale 9
I.5.1.Capteurs Médicaux 10
I.5.1.1.La radiographie 10
I.5.1.2.Le scanner 11
I.5.2.Critères spécifiques dans le domaine médical 13
I.5.3.Compression des images médicales 14
I.6.Taux de compression 15
I.7.Optimisation de la compression en termes de débit-distorsion 15
I.8.Codage sans perte 17
I.8.1.Codage arithmétique 18
I.8.2.Codage de Huffman 20
I.8.3.Codage Lempel-Ziv 21
I.8.4.Le Codage par plage (Run length Encoding) 21
I.8.5.Codage différentiel 23
I.9.Codage avec perte 23
I.9.1.Quantification 24
I.9.1.1.Quantification Scalaire 24
I.9.1.2.Quantification Vectorielle 25
I.10.Formats d'images 25
I.11.Conclusion 26

Chapitre II

Réseaux de neurones

II.1.Introduction....28
II.2.Réseaux de neurones MLP....28
II.2.1.Architecture et fonctionnement du réseau multicouche....28
II.2.2.Fonctions de transfert....30
II.2.3.Mise en œuvre des réseaux neuronaux....32
II.2.4.Algorithmes d'apprentissage....33
II.2.4.1.Rétropropagation du gradient....33
II.2.4.2.Résumé de l'algorithme de Rétropropagation....36
II.2.4.3.Considérations pratiques....37
II.2.4.4.Accélération de l'algorithme avec le momentum....37
II.3.Réseaux de neurones à fonction de base radiale (RBF)....37
II.3.1.Formalisme....38
II.3.2.Apprentissage des réseaux RBF....40
II.3.3.Choix de la métrique et de la largeur des noyaux....41
II.4.Applications des réseaux de neurones sur les images....43
II.4.1.Prétraitement des données....43
II.4.2.Base d'apprentissage....44
II.4.3.Architecture du réseau....46
II.4.4.Critères d'apprentissages....47
II.4.5.Apprentissage des réseaux de neurones....48
II.4.6.Conception du Réseau de neurones....48
II.4.6.1.Configuration et optimisation des caractéristiques du réseau....48
II.4.7.Réseaux de neurones MLP....48
II.4.7.1.Présentation des simulations effectuées....48
II.4.7.2.Algorithme d'apprentissage....49
II.4.7.3.Analyse des résultats....50
II.4.7.4.Validation des résultats....51
II.4.8.Réseau de neurones RBF....58
II.4.8.1.Algorithme d'apprentissage....59
II.4.8.2.Validation du modèle....61
II.5.Compression d'image médicale....68

II.6.Conclusion 74

Chapitre III

Transformée de Cosinus Discret (DCT)

III.1.Introduction 76
III.2.Codage par Transformée 76
III.3.Norme JPEG 77
III.3.1.Principe de la compression JPEG 77
III.3.1.1.Traitement des images couleur 78
III.3.1.2.DCT et IDCT 79
III.3.1.2.1.Calcul de la DCT et de l'IDCT 79
III.3.1.3.Quantification 80
III.4.Les courbes de scanning 82
III.4.1.Définition et intérêt 82
III.5.Méthode proposée 85
III.5.1.Compression d'image 85
III.5.2.Compression sans perte 89
III.5.3.Les Résultats de la simulation 91
III.5.3.1.Résultats sur les images aux niveaux de gris 91
III.5.3.2.Résultats sur les images couleur 94
III.5.3.3.Application sur les images médicales 99
III.6.Conclusion 103

Chapitre IV

Étude comparative

IV.1.Introduction 105
IV.2.Méthodes proposées 105
IV.2.1.Méthode basée sur la décimation et les réseaux de neurones 105
IV.2.2.Méthode basée sur la DCT et les nouveaux scanning adaptatif et l'encodeur sans perte 107
IV.2.3.Étude comparative globale 110
IV.3.Conclusion 112
Conclusion Générale 114
Bibliographie

Introduction générale

Introduction générale

Dans nombreux domaines, l'image numérisée remplace les images analogiques classiques. Dans le domaine médical, l'utilisation des images radiographies, ultrasonores, IRM, pose un grand problème de stockage et d'archivage. Par exemple ; un hôpital de 200 lits, produit en moyenne chaque année 875 Go de données images. En plus du problème de stockage, si de telles images doivent être transmises via un réseau, la durée de la transmission est souvent trop longue. Pour palier à tous ces problèmes, la compression de ces images devient une opération nécessaire et impérative. Le but principal de la compression des images est de réduire la quantité de bits nécessaires pour les décrire tout en gardant un aspect visuel acceptable des images reconstruites [1].

Les études ont prouvé que le service de radiologie d'un grand hôpital peut produire plus de 20 terabits de données d'image par année [2]. Même s'il y avait la possibilité de stockage infini, il reste le problème crucial à savoir la transmission de ces images à travers un réseau interne ou externe.

De nos jours la téléradiologie est importante surtout pour les hôpitaux se trouvant dans le milieu rural, où il y a un manque énorme de spécialistes dans le domaine de radiologie. En effet, un technicien de base dans le domaine de radiologie dans ce type d'hôpitaux, peut prendre une radio à un patient, puis l'envoyer via un réseau à un grand hôpital se trouvant dans le milieu urbain pour avoir un diagnostic qui permettra au médecin traitant de prescrire la thérapie nécessaire au patient.

La compression permet l'archivage efficace des images médicales et des données associées que l'on pourra consulter, analyser ou étudier, éventuellement à distance. Dans ces systèmes, la réduction de la taille des images, tout en gardant une bonne qualité, est une nécessité. La compression autorise le transfert par réseau, en temps réel, des images : elle permet l'utilisation des systèmes de communication pour le télédiagnostic en télémédecine.

Les deux objectifs principaux de la compression d'image vont, d'une part occupé moins d'espace mémoires et, d'autre part, permettre de transmettre plus rapidement l'information. La télémédecine qui est un domaine d'application de l'imagerie médicale dans cette étude. Elle a pour but la possibilité de transférer des images en un temps limité [3].

Nous distinguons deux types de compression de données, selon qu'il est possible ou non de retrouver exactement l'information de départ : la compression sans perte, et avec perte.
La compression sans perte est complètement réversible, aucune perte d'information n'est introduite par les processus de compression/décompression. Les taux de compression que nous pouvons atteindre sont limités. Pour les images médicales, ils sont de l'ordre de 2 à 3. La

compression sans perte est employée dans le cas où on ne permet habituellement pas la perte de données et c'est exactement ce qui est exigé, comme dans le cas des images médicales.

La compression sans perte, peut être très efficace dans l'imagerie médicale [4]. Les techniques de prédiction (Prediction-based) sont plus efficaces pour le codage sans perte parce qu'elles produisent les meilleurs résultats de codage et sont très faciles à mettre en application [5].

La compression avec perte est capable de réaliser une compression beaucoup plus élevée (20 :1) et même d'un rapport plus élevé.

Il y a beaucoup de techniques de compression d'image disponibles pour la compression des images, telles que celles basées sur la DCT (Discrete Cosine Transform) [6], JPEG (Joint Photographic Experts Group) [7], quantification vectorielle [7] [3], codage sous-bande, JPEG2000 [8], et des techniques de compressions basées sur la transformée d'ondelette [9] tel que EZW (embedded coding using zerotrees of wavelet coefficients) ou SPIHT (set partitioning in hierarchical trees) [10], [11] etc., Le but commun de toutes ces techniques est d'obtenir un taux de compression élevé. Ce sont toutes ces considérations qui ont guidé ce travail. Par conséquent l'objectif est de proposer des méthodes de compression pour des applications spécifiques, plus pointues particulièrement dans le domaine médical. Nous montrerons que les algorithmes proposés, comme techniques sur lesquelles notre travail est basé, possèdent des propriétés très intéressantes pour atteindre cet objectif.

Dans le traitement d'image, les réseaux de neurones sont aussi de plus en plus utilisés. Lors de la réalisation des différentes applications, plusieurs types d'architectures de réseaux et différentes méthodes d'apprentissage ont été utilisés. On les utilise pour la compression, et la restauration de l'image, la détection de contours, l'extraction de régions, la reconnaissance de textures, la reconnaissance de caractères, et enfin la segmentation des images.

Les approches utilisant les réseaux de neurones artificiels pour le traitement intelligent des données semblent être très prometteuses au regard des avancées déjà enregistrées ici et là.

Dans notre travail nous avons proposé deux méthodes de compression. La première basée sur les réseaux de neurones (MLP, RBF) selon une méthode simple qui est la décimation. La deuxième repose sur une méthode de compression à base de transformée de cosinus discret (DCT), les nouveaux scannings adaptatifs et encodeur sans perte [12].

L'organisation générale du mémoire est décrite ci-dessous :

Le chapitre I est consacré à la présentation des différentes méthodes de compression des images avec perte et sans perte, et des notions générales telles que les mesures de performance et de distorsion.

Le chapitre II est dédié à la présentation générale des réseaux de neurones MLP et RBF ainsi qu'une étude comparative des performances obtenues lors de la reconstruction de l'image.

Le chapitre III traite deux parties. Dans la première partie nous présentons la norme de compression JPEG, dans la deuxième partie, nous présentons une nouvelle méthode de compression d'image [12] qu'est validée par une étude comparative avec des méthodes récentes citées dans la littérature telle que CBTC-PF [13] et JPEG.

Le chapitre IV par une étude comparative présente les différentes méthodes de compression utilisées dans notre travail (MLP, RBF, DCT-Scanning-sans perte, CBTC-PF et JPEG).

La dernière partie de ce mémoire est consacrée à la conclusion générale et à quelques perspectives en la matière.

Chapitre I

Généralités sur la compression des images

I.1. Introduction

Compresser une image consiste à trouver une représentation numérique de taille inférieure à celle de l'image initiale. L'opération de décompression permet de retrouver la représentation initiale, soit sous une forme identique, soit légèrement dégradée.

La compression est d'ailleurs souvent liée à un format de fichier (.jpg, .gif, .bmp) ou à un protocole de transfert de données.

Notons que la compression des images est divisée en deux axes principaux : compression avec perte et celle sans perte. Le premier type de compression, utilise uniquement le principe de la réduction de l'information et n'engendre pas de perte, le deuxième type définit une représentation approximative de l'information.

Nous allons dans ce chapitre exposer quelques concepts de l'image puis nous abordons les besoins en compression d'image médicale, et par la suite nous présentons les éléments fondamentaux pour mesurer la qualité de l'image compressée sous une forme quantitative.

I.2. Définition d'une image et des types d'images

Une image est stockée en mémoire sous forme de collection de points élémentaires appelés pixels. Nous pouvons considérer une image numérique comme une page de nombres organisés en tableau ou en matrice (image bitmap). Chaque nombre représente les caractéristiques du pixel. La position de chaque pixel peut être exprimée par deux coordonnées sur l'axe horizontal *X* et l'axe vertical *Y* comme le montre la figure ci-dessous.

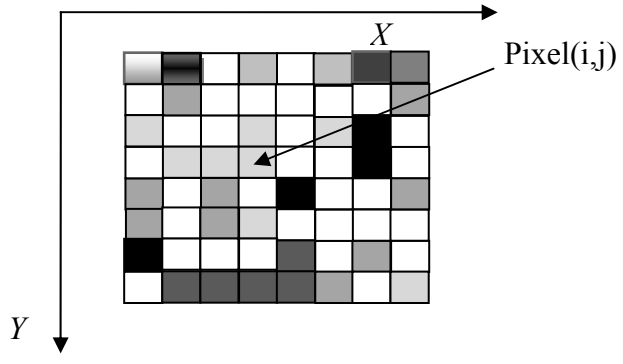


Figure I.1 Elément d'une image : le pixel

Le codage d'un pixel dépend du type d'image et nous en recensons trois types :

1. Les images à deux niveaux : On parle des images binaires, c'est-à-dire un seul bit décrit chaque point, 0 représente un point noir, 1 représente un point blanc.

2. Les images à plusieurs niveaux de gris : un nombre plus élevé de niveaux de gris peut être distingué .La plupart du temps, les images monochromes sont codées sur huit bits.

3. Les images couleurs : la couleur peut être codée, soit par composition de couleurs primaires, soit par composition d'informations de luminance et de chrominance. En effet, une couleur peut être représentée par un ensemble de trois coordonnées, c'est-à-dire elle peut être reproduite par la superposition de trois couleurs primaires comme le montre la figure I.2. Le système *RGB* (*RVB*) utilise les couleurs primaires : rouge, vert et bleu. La valeur du pixel doit représenter les composantes trichromatiques de la couleur. En général, nous disposons de huit bits pour coder une composante, soit 24 bits pour coder la valeur d'un pixel.

Le système *RGB* peut donc définir plus de 16 millions de couleurs. Des résultats expérimentaux ont prouvé que l'œil est beaucoup plus sensible aux variations fines de l'intensité lumineuse (luminance) qu'à celles de la couleur (chrominance). Donc, Il en résulte que nous pouvons nous contenter de transmettre l'information de couleur avec moins de détails que l'information de luminance [3], [14].

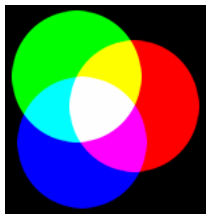


Figure I.2 Superposition des trois couleurs : rouge, vert et bleu (RGB)

I.3. Changement d'espace de couleur

Toute longueur d'onde visible peut être visuellement simulée en convoluant le signal avec les fonctions de sensibilité des trois différents capteurs rétiniens du système visuel humain dit *LMS* (Large=565nm dit rouge, Medium=535nm dit vert, Short=430nm dit bleu). Dans le cas d'une compression avec perte, la reconstruction de chaque bande (*RGB*) risque de ne pas appréhender les structures de l'image de la même façon, engendrant différentes erreurs de reconstruction et par la même, de fausses couleurs visuellement choquantes. On préférera donc un espace de luminance et chrominance rouge et bleu *YCbCr* (ou *YUV*) où les primaires sont décorrélées, ce qui offre l'avantage de séparer les informations d'intensité lumineuse et

de couleur. Un tel espace permet de gérer les premières avec plus de soin [14], [15].

Passage de l'espace *RGB* à l'espace *YCbCr* [8] est définie par l'équation suivante :

$$\begin{bmatrix} Y \\ C_b \\ C_r \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.144 \\ -0.16875 & -0.33126 & -0.5 \\ 0.5 & -0.41869 & 0.08131 \end{bmatrix} \times \begin{bmatrix} R \\ G \\ B \end{bmatrix} \quad \text{(I.1)}$$

Les équations de passage des composantes *YCbCr* aux composantes *RGB* s'obtiennent facilement par l'équation ci-dessous :

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1.402 \\ 1 & -0.34413 & -0.71414 \\ 1 & 1.772 & 0 \end{bmatrix} \times \begin{bmatrix} Y \\ C_b \\ C_r \end{bmatrix} \quad \text{(I.2)}$$

La transformation des composantes *RGB* en composantes *YCbCr* est une transformation bijective, il n'y a pas de pertes de données pendant cette étape. Les schémas de la figure I.3 montrent les différents formats de sous-échantillonnage existant :

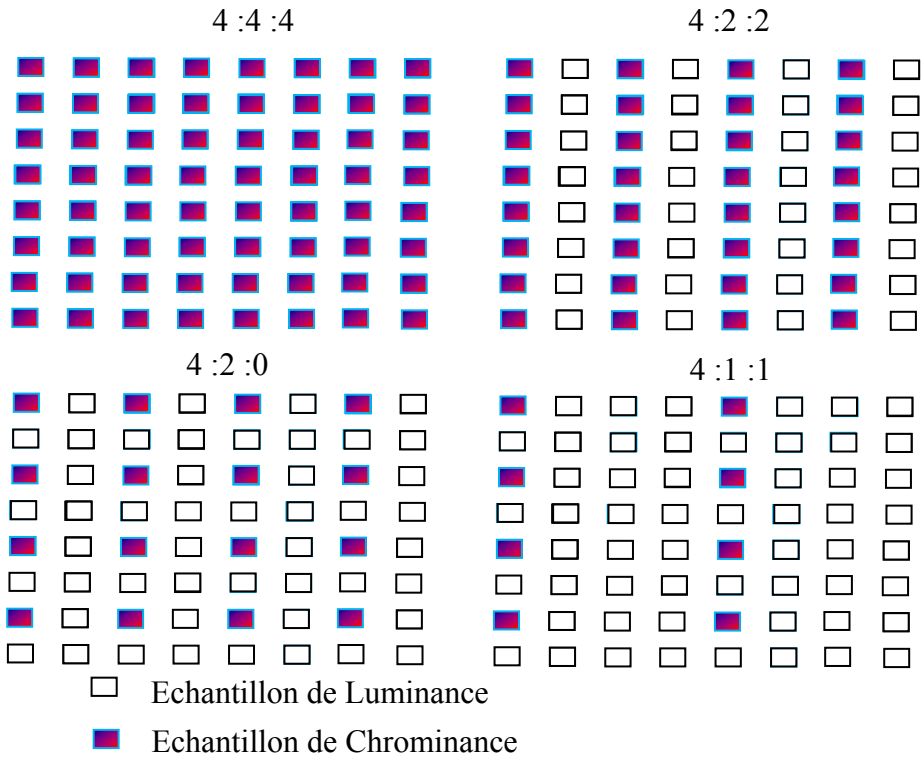


Figure I.3 Les différents formats de sous-échantillonnage des chrominances

Il existe plusieurs formats d'échantillonnage pour *YCbCr* tels que le format 4 :4 :4, 4 :2 :2, 4 :1 :1 et 4 :2 :0 [16] [17].

Il existe d'autres modèles comme les espaces : *HSI* (Hue Saturation Intensity) et *HSV* (Hue Saturation Value), qui prennent en relation les notions intuitives de teinte, saturation, et luminance.

- *CMY(K)* (Cyan Magenta Yellow (blacK)) utilisé pour les impressions couleur.
- *YIQ*, *YUV*, *YCbCr* (Luminance-Chrominance) utilisé en imagerie et en vidéo.
- Il existe également des espaces colorimétriques utilisés pour le traitement d'images comme les systèmes $I_1I_2I_3$, *XYZ*, *Luv* et *Lab* qui ont été définis par la *CIE* (Commission Internationale de l'Éclairage).

I.4. Compression d'image et critères d'évaluation

La compression consiste à réduire le volume de données nécessaire à la description de l'image. L'exploitation de la redondance spatiale et des faiblesses du système sychovisuel a permis de développer des méthodes où la quantité de données nécessaires pour représenter l'image est réduite [18].

La qualité visuelle des images numériques augmente continuellement avec le développement de nouvelles techniques d'affichage (multirésolution, transmission progressive) et de nouvelles technologies d'acquisition (haute définition, nouveau hardware).

Cependant, la taille de ces images augmente proportionnellement avec leur qualité, donc leur stockage constitue un enjeu principal dans le monde numérique.

La compression s'impose comme une étape incontournable pour optimiser l'utilisation de ces grands volumes d'informations dans les réseaux informatiques. L'objectif principal de la compression d'image est de réduire la quantité d'information nécessaire à une représentation visuelle fidèle à l'image originale [19].

Deux techniques sont utilisées pour évaluer la distorsion : subjective et objective.

1. Les méthodes subjectives, nécessitent des tests psychovisuels de l'œil humain. Les tests sont réalisés à plusieurs échelles avec des groupes de personnes.

2. Les méthodes objectives utilisent le rapport signal crête sur bruit pour une image dont le maximum est $(2^R - 1)$ dénoté *PSNR* entre la source initiale et celle distordue, où *R* le nombre de bits de l'image originale.

Soient $x_{i,j}$ la valeur initiale de l'image et $\hat{x}_{i,j}$ sa valeur codée ou distordue, *M* et *N* étant les dimensions de l'image.

L'erreur quadratique moyenne *MSE* est définie par l'équation :

$$MSE = \frac{1}{M \times N} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} (x_{i,j} - \hat{x}_{i,j})^2 \tag{I.3}$$

$$PSNR = 10 \times \log_{10} \left(\frac{(2^R - 1)^2}{MSE} \right) dB \tag{I.4}$$

D'autres critères objectifs sont utilisées tels que [20] :

1. L'erreur absolue moyenne *MAE* : si cette valeur est petite, la reconstruction est meilleure, elle est définie par l'équation ci-dessous :

$$MAE = \frac{1}{M \times N} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} \left| x_{i,j} - \hat{x}_{i,j} \right| \tag{I.5}$$

2. L'erreur moyenne quadratique normalisée *NMSE* : ce critère normalise le *MSE* sur l'échelle de 0 à 100%.

$$NMSE = \frac{100 \times MSE}{\sigma^2}, \tag{I.6}$$

avec σ^2 l'énergie moyenne de l'image originale.

I.5. Image médicale

La médecine a pour but la prévention ou la guérison des maladies ou des blessures qui peuvent affecter les êtres vivants; et pour cela, il est nécessaire de bien connaître l'état du patient à soigner : avant tout traitement, le médecin délivre un diagnostic sur l'état du patient; dans cette optique, la médecine utilise de nombreux signaux (électrocardiogramme, électroencéphalogramme, ...) ou images (radiographies, échographies, ...) qui donnent des indications sur l'état critique de la santé du patient [21].

I.5.1. Capteurs Médicaux

I.5.1.1. La radiographie

En 1895, Wilhelm Rontgen, un professeur de l'université de Wurzburg (Allemagne) publie un article intitulé ‘‘sur une nouvelle sorte de rayonnement’’: il a découvert les rayons X, rayons pour lesquels certains matériaux tels le papier ou les tissus humains (la peau par exemple) sont tout à fait transparents, alors que ce rayonnement est plus ou moins atténué par d'autres matériaux, les os par exemple. Ce rayonnement permet donc de voir, entre autres, à l'intérieur du corps humain, ce que ne permettent évidemment pas les rayons lumineux. Pour cette découverte, Wilhelm Rontgen reçoit en 1901 le premier prix Nobel de Physique.

La technique de radiographie consiste donc à émettre des rayons X à travers un objet sur l'intérieur duquel on recherche des informations; typiquement, l'objet en question sera un patient, ou plutôt la section de ce patient à analyser : le genou, le dos, ... Les rayons X, lorsqu'ils traversent certains matériaux comme les os, sont atténués ; l'intensité $I_{x,y}$ du rayon qui, partant de la source S avec une intensité I_0, frappe le détecteur en (x, y) suit la loi de Lambert-Beer (voir figure I.4). Sur ce détecteur se trouve un film argentique, qui contient des ions argentés Ag+ ; un nombre d'électrons dépendant de l'intensité du rayon qui frappe le détecteur sont alors libérés, captés par les ions Ag + qui se transforment en argent et perdent leur propriété de transparence ; de sorte, une forte intensité créera une impression noire ; les zones plus opaques aux rayons X apparaissent nettement, en blanc [21].

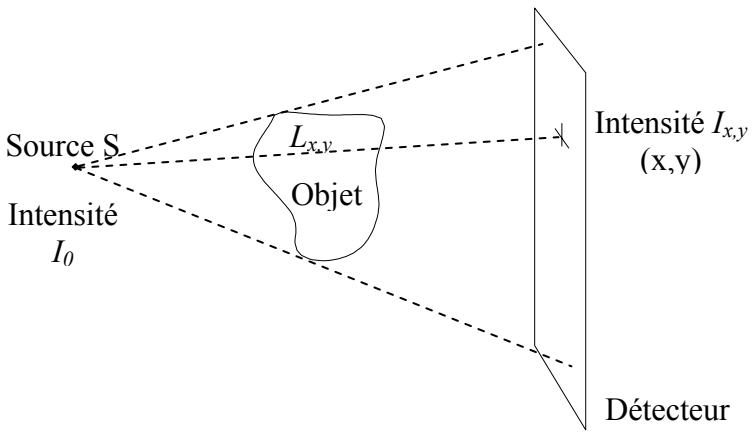


Figure I.4 Atténuation du rayon arrivant en (x, y) selon la loi de Lambert-Beer

Le dispositif expérimental de radiographie est photographié à la figure I.5.

Figure I.5 C-arm : dispositif léger et mobile pour faire de la radiographie en salle d'opération [21]

I.5.1.2. Le scanner

Le scanner fonctionne sur le même principe que la radiographie, mais on veut désormais reconstituer un objet bidimensionnel ou tridimensionnel; l'émission de rayons X est donc faite suivant différentes orientations.

La figure I.6 est une photo de scanner : un tube à rayons X (la source) et des détecteurs sont disposés en couronne ; le patient se trouve sur le lit, qui peut se translater : la couronne tourne autour du patient de sorte à en analyser une coupe transversale, puis le lit se translate : une nouvelle coupe commence alors à être acquise. Le scanner reconstruit donc une succession de coupes internes, donc non accessibles directement par la radiographie du patient ; en acollant toutes ces coupes, on reconstruit un volume.

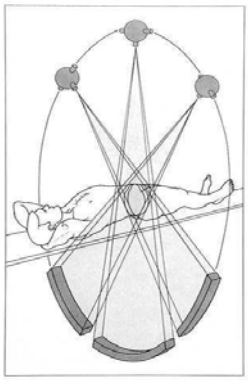


Figure I.6 Reconstruction d'une coupe transversale d'un patient [21]

Sur des modèles plus récents de scanner, la rotation de la couronne et la translation du lit se font en même temps : les données sont donc obtenues suivant une hélice (on parle d'acquisition hélicoïdale). Notons que, même si l'acquisition et la reconstruction se font maintenant très rapidement, la quantité de rayons *X* utilisée pour un scanner est bien supérieure à celle utilisée pour une radiographie, puisque ici il s'agit de reconstruire un objet tridimensionnel.

Considérons un rayon émis par le tube à rayon *X*, situé en *O* ; il va frapper un détecteur situé en X(x, y) avec une intensité atténuée selon la loi de Lambert-Beer : $I_{x,y} = I_0 e^{-\int_{OX} f(\vec{u})d\vec{u}}$; les données dont on dispose pour reconstruire une coupe sont donc les $\int_{OX} f(\vec{u})d\vec{u}$, c'est-à-dire, exprimées à l'aide des coordonnées de la figure I.6,

$$Rf(\vec{\theta},s) = \int_{\vec{x}.\vec{\theta}=s} f(\vec{x})d\vec{x} \qquad \text{(I.7)}$$

$Rf(\vec{\theta},s)$ est la transformée de Radon de *f* ; on sait inverser cette transformée, et donc retrouver, à partir de la donnée des $Rf(\vec{\theta},s)$, la fonction *f(x, y)* dessinant la coupe recherchée.

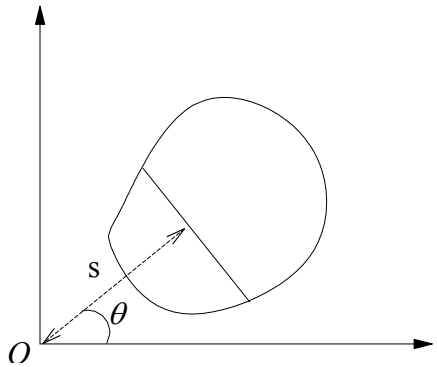


Figure I.7 Transformée de Radon $Rf(\vec{\theta},s)$: c'est l'intégrale de *f* le long d'une droite perpendiculaire à la direction $\vec{\theta}$ et à une distance s de l'origine ($\vec{\theta}$ désigne en fait un point de la sphère unité, c'est-à-dire du cercle trigonométrique dans R2).

Figure I.8 Photo d'un scanner [21]

I.5.2. Critères spécifiques dans le domaine médical

Les contraintes du domaine médical nécessitent d'être prudent dans l'utilisation des techniques précédemment décrites. Du fait, une valeur élevée de *PSNR* peut sembler suffisante pour accepter une technique de codage. Cependant, la dégradation moyennement faible peut être concentrée juste sur une petite zone importante de l'image, et par là empêcher d'établir un diagnostic précis. La question primordiale est l'exactitude du diagnostic ; pour mesurer cela, les évaluations subjectives utilisées par exemple pour des images de vidéo ne sont pas non plus appropriées. C'est pour cette raison que des approches spécifiques sont nécessaires dans le domaine médical. La méthode généralement utilisée en médecine est basée sur l'analyse ROC, provenant du terme anglais Receiver Operating Characteristic [22]. La procédure est similaire à l'évaluation subjective ; elle consiste à faire intervenir des experts humains, puis à analyser les données obtenues. Plusieurs experts décident individuellement si une anomalie est présente ou absente sur les images choisies. Ils donnent également à chaque diagnostic un degré de confiance, entre 1 et 5 par exemple, pour décrire à quel niveau leur décision est sûre. L'analyse des données obtenues sert à déterminer la courbe ROC qui décrit la relation entre les probabilités d'une pathologie correctement ou faussement détectée. C'est en général la surface sous la courbe qui est calculée pour mesurer la fidélité du diagnostic [3].

L'analyse ROC est la technique dominante pour évaluer l'applicabilité des approches de traitement d'images dans le diagnostic. C'est une méthode très coûteuse : une étude ROC typique nécessite environ 300 images pour avoir un degré acceptable de confiance statistique, au moins cinq experts pour évaluer ces images et un statisticien à plein temps pour surveiller la procédure et analyser les données. L'autre désavantage principal est que les origines de cette

méthode résident dans la théorie de la détection de signaux, la mise en équivalence du diagnostic avec une détection de signal et une modélisation trop simplifiée. L'utilisation des notes de confiance, la restriction à une réponse binaire (cas pathologique/normal) et des suppositions statistiques non vérifiées sur les images alourdissent également cette approche.

Des études plus récentes visent alors à la rendre plus appropriée et aisée ou à développer une alternative adéquate [3].

Nous avons présenté dans cette section les principales méthodes pour évaluer la qualité des images compressées : les mesures (numériques), qui cherchent à refléter au mieux l'appréciation subjective, et les procédures d'évaluation de la qualité subjective, dont l'équivalent dans le domaine médical consiste à déterminer la fidélité du diagnostic. Mais, comme nous l'avons mentionné précédemment, la plupart des travaux applique une partie très réduite de ces techniques : ils se restreignent à donner des valeurs objectives du type *MAE*, *MSE*, *NMSE* ou *PSNR*, avec les arguments subjectifs et personnels des auteurs ; étant donné les avantages évidents de cette approche, dans notre travail nous nous contentons également d'évaluer la qualité par des critères numériques et des remarques subjectives.

I.5.3. Compression des images médicales

L'imagerie médicale numérique concerne environ 30% des examens radiologiques. Elle recouvre une grande diversité dans les principes physiques mis en jeu (rayons-X, ultrasons, résonance magnétique nucléaire, émission de positron, médecine nucléaire, CT ou computed tomography et dans les différentes applications médicales (radiographie d'urgence, procédures interventionnelles...). Une méthode de compression bien adaptée à l'une des modalités d'imagerie médicale ne l'est pas forcément pour une autre. Où, tout au moins, une méthode peut demander des adaptations différentes selon les modalités [3].

Une motivation majeure des recherches en compression d'images médicales vient de la proportion croissante d'examens acquis numériquement. Il faut les stocker, les communiquer et les visualiser malgré les masses de donnée requises. Les examens directement acquis numériquement sont par exemple le scanner, l'IRM, l'angiographie. La présence dans un même établissement de films conventionnels et d'images numériques pose des problèmes pratiques. La tendance est de numériser aussi les examens non digitaux à l'origine, en numérisant les films argentiques ou en utilisant des plaques de phosphore (systèmes CR Computed Radiography) à la place des cassettes pour films traditionnels. Toutes les images numériques peuvent être archivées, communiquées et visualisées à l'aide de réseaux numériques tels que les PACS (Picture Archiving and Communication System). La viabilité économique des PACS a été

difficile à prouver lors de leur introduction à la fin des années 80. Mais leur émergence et la tendance irréversible vers les solutions numériques ont fortement motivé l'étude de la compression d'image [3].

I.6. Taux de compression

Ce critère est très important puisqu'il est directement dépendant de la technique de compression utilisée [18].

Il est représenté soit comme une formule, équation (I.8), soit comme un facteur, équation (I.9). Dans les équations (I.8) et (I.9), I_0 est la taille de l'image originale en octet et I_c la taille de l'image comprimée. Le taux de compression peut être aussi quantifié par le nombre moyen de bits par pixel (*bpp*), équation (I.10). L'élément $Bits_{Ic}$ est le nombre total de bits de l'image comprimée et $Pixels_{I_0}$ est le nombre total de pixels de l'image originale [19].

$$\sigma = \frac{I_0}{I_c} \tag{I.8}$$

$$\sigma = \left(\frac{I_0}{I_c} \right) : (1) \tag{I.9}$$

$$\sigma = \frac{Bits_{I_c}}{Pixels_{I_0}} \tag{I.10}$$

I.7. Optimisation de la compression en termes de débit-distorsion

Tout processus de compression est une affaire de compromis entre le taux de compression atteint et la qualité de reconstruction désirée. C'est pourquoi les performances d'un tel processus se jugent sur la qualité du signal reconstruit pour un débit fixe, c'est-à-dire en termes de débit-distorsion. Cette notion de débit-distorsion peut être schématisée par l'intermédiaire d'une courbe (Figure I.9), qui représente l'évolution théorique optimale du débit *R*, en fonction de la distorsion *D*, si l'on tient compte des principes fondamentaux de la théorie de l'information [23].

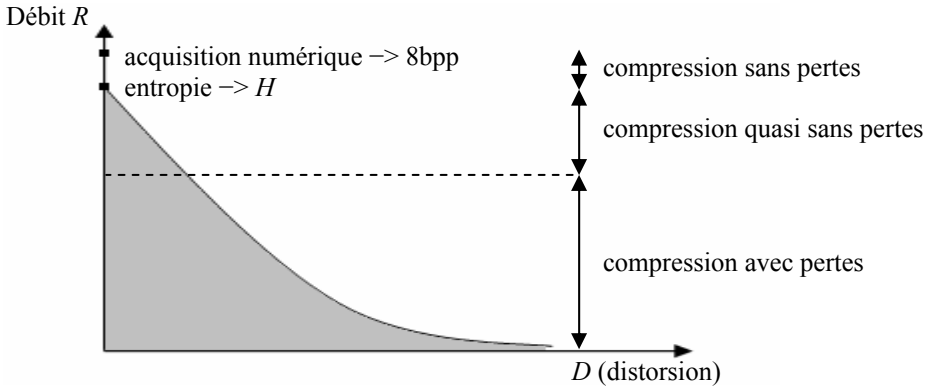


Figure I.9 Evolution théorique optimale du débit en fonction de la distorsion pour une image

Ainsi nous pouvons remarquer que l'acquisition numérique d'une image sur 8 bits correspond à une quantification très fine permettant la représentation de l'image sur 256 niveaux de gris (de 0 à 255). Cette image en niveaux de gris devient alors la source à compresser. Si l'entropie représente le minimum de débit que l'on peut théoriquement atteindre lors d'une compression sans pertes, il nous faut tout de même remarquer que la sensibilité de l'œil étant limitée, elle permet la réalisation d'une compression quasi sans pertes dans une plage de débit plus important. Passé ce débit nous tombons dans un cadre de compression avec pertes. Cette courbe représentant l'optimal alors aucune observation réelle ne peut se trouver dans la partie grisée mais il faut cependant réaliser une compression approchant au mieux cette courbe de débit-distorsion théorique. Il nous faut donc minimiser la distorsion sous une contrainte de débit [23].

D'une autre manière, la compression de données permet de diminuer la taille de stockage et rend aisé leur transport à travers des réseaux modesses de communication. Certaines normes de compression disposent de techniques permettant la résistance aux erreurs en paquets lorsque les données sont transmises sur un réseau dont la qualité de service n'est pas garantie.

Pour une source donnée, la mesure permettant d'évaluer la capacité de codage est l'entropie. La définition de l'entropie *H* est donnée par l'expression :

$$H = -\sum_{n=0}^{M-1} P_r(x_n) \times log_2(P_r(x_n)) \ , \tag{I.11}$$

où $P_r(x_n)$ est la probabilité de la valeur d'intensité x_n dans la source et *M* est la dynamique du signal (pour une image, typiquement *M=256*).

Shannon a démontré qu'il est possible de diminuer le coût de codage d'une source donnée en regroupant les symboles à coder. Plusieurs techniques de codage sont inspirées par ce théorème, par exemple le codage de Huffman. On peut donc coder la source sans perte d'information (réversible) jusqu'à son entropie. En revanche, au delà de l'entropie une distorsion apparaît dans la source, codage irréversible [24].

Il existe deux types bien distincts de compression : La compression sans perte et avec pertes.

I.8. Codage sans perte

La compression sans perte (codage réversible) permet de retrouver la valeur exacte du signal comprimé lorsqu'il n'y aucune perte de données sur l'information d'origine. En fait, la même information est réécrite d'une manière plus concise. Le processus de codage sans perte crée des ‘‘mots-codes’’ à partir d'un dictionnaire statique ou d'un dictionnaire construit dynamiquement. Ces processus s'appuient sur des informations statistiques de l'image. Les codes statistiques les plus répandus sont le codage de Huffman et le codage arithmétique. Le codage statistique permet de s'approcher au mieux de l'entropie. Ils ont pour principe d'associer aux valeurs les plus probables les mots binaires les plus courts [19].

Dans ce cas les compromis liés à ce mode de compression sont selon trois axes illustrée par la figure I.10 [25] :

- Efficacité du codage, ceci peut être mesuré en bit par pixel (échantillon), elle est limitée par l'entropie de la source. Plus l'entropie de la source est grande plus il est difficile a compresser (exemple un bruit aléatoire).
- Temps de codage, celui-ci est lié à la complexité du processus de codage ou de décodage. Il peut être réduit si on augmente la capacité de calcul du composant de traitement. Pour certaines applications ce temps est contraint ce qui impose le choix de la technique de codage.
- Complexité du codeur, elle peut être mesurée à l'aide de la quantité de ressources utilisées en termes de mémoire et du nombre d'opérations arithmétiques. Le nombre d'opérations est donné par MOPS (million of operations per second). Le MIPS

(Millions of Instruction Per Second) est parfois utilisé. Si on ajoute l'aspect mobile de l'application, la consommation d'énergie peut être considérée comme caractéristique de la complexité de codage.

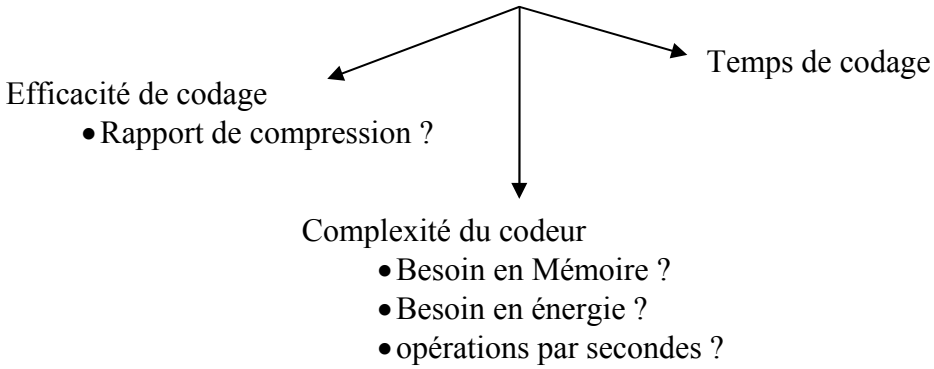


Figure I.10 Les compromis en mode de codage sans perte

I.8.1. Codage arithmétique

Le codage arithmétique se singularise par sa capacité à coder chaque symbole sur un nombre non entier de bits [14]. En réalité, il n'assigne pas un mot de code à chaque symbole mais il associe un point de l'intervalle [0,1] à un ensemble de symboles [18]. Le principe repose sur le découpage de l'intervalle [0,1]. Chaque symbole se voit attribuer une partition de l'intervalle dont la taille est égale à sa probabilité d'occurrence. L'ordre de rangement est mémorisé pour être utilisé lors du décodage.

Le codage arithmétique est généralement plus performant que le codeur de Huffman. Il tend vers la limite inférieure théorique mais cependant il est gourmand en ressources et nécessite de connaître à priori l'intégralité du signal avant de pouvoir procéder au codage [14].

Les différentes étapes de l'algorithme de codage sont :

- L'initialisation : nous affectons à chaque symbole une plage d'intervalle basée sur sa probabilité d'apparition fournie par le modèle. Les bornes externes de cet intervalle sont zéro et un.
- Le traitement du message : nous initialisons un intervalle de travail en prenant comme bornes zéro et un. Le premier symbole est représenté par la plage qui lui est affectée à l'étape 1. Chaque symbole suivant restreint davantage l'intervalle et il est représenté par sa plage relative dans la plage précédente. Ainsi le flot de données est traduit par un nombre contenu dans la dernière plage calculée.

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

Par exemple, supposons qu'on veut coder une partie ''*acaab*'' d'une longue séquence avec une probabilité d'apparition indiquée dans le Tableau I.1 [14].

Symbole	Probabilité	Intervalle
A	0.7	[0, 0.7]
B	0.1	[0.7, 0.8]
C	0.2	[0.8, 1.0]

Tableau I.1 Probabilités des symboles

L'intervalle initial [0, 1] va être divisé en trois sous-intervalles suivant les probabilités des symboles de la séquence. Ce qui donne les sous-intervalles suivants: [0, 0.7], [0.7, 0.8] et [0.8, 1.0].

Dans cette exemple, le 1er symbole est ''a'', l'étiquette appartient donc à l'intervalle [0, 0.7]. Après que le 1er symbole soit codé, les limites inférieures et supérieures de l'intervalle sont respectivement 0 et 0.7 pour le symbole suivant. L'intervalle [0, 0.7] va être divisé en trois sous-intervalles : [0, 0.49], [0.49, 0.56] et [0.56, 0.7] correspondant respectivement aux symboles ''*a*'', ''*b*'' et ''*c*''.

Le 2ème symbole est ''*c*'', de probabilité 0.2. Ainsi, le nouveau sous-intervalle sera [0.56, 0.7]. Ce dernier va être divisé en trois sous-intervalles : [0.56, 0.658], [0.658, 0.672] et [0.672, 0.7].

Le 3ème symbole est ''*a*''. Le nouveau sous-intervalle sera en conséquence [0.56, 0.658]. Le sous-intervalle [0.56, 0.658] va être partagé à son tour en trois sous-intervalles :

[0.56, 0.6286], [0.6286, 0.6286] et [0.6286, 0.658].

Le 4ème symbole est ''*a*'', donc le nouvel intervalle est [0.56, 0.6286].

Et ainsi de suite jusqu'à que l'on code la totalité de la séquence. On aura à la fin, le sous-intervalle suivant [0.60802, 0.61488]. Un nombre contenu dans l'intervalle final comme 0.60803 code sans ambiguïté le message ''*acaab*''. La figure I.11 est une représentation de ce processus.

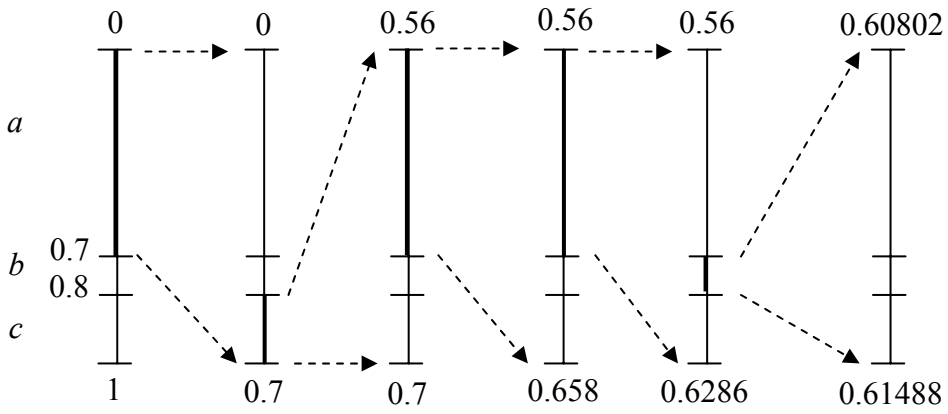


Figure I.11 Génération de l'étiquette pour la séquence ''*acaab*''

I.8.2. Codage de Huffman

Le codage de Huffman consiste à coder les symboles par une représentation de bits à longueur variable. Les symboles ayant la probabilité d'apparition forte sont codés avec des chaînes de bits plus courtes, tandis que les symboles dont la probabilité d'apparition est faible sont codés par des chaînes plus longues. Le code d'un symbole ne doit pas être le préfixe d'un autre code. Cette propriété est admise afin que la reconnaissance soit possible. Pour représenter le codage de Huffman, on utilise l'arbre binaire [24].

Soit un message a codé ''*ABBBBAAC*''. La fréquence d'apparition ainsi que le code Huffman correspondant est donné dans le tableau I.2 et représentés par la figure I.12.

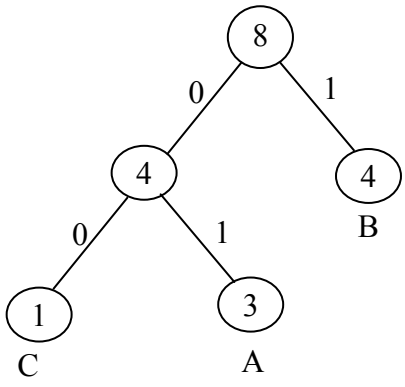


Figure I.12 Arbre binaire de Huffman

Symbole	A	B	C
Fréquence d'apparition	3	4	1
Code Huffman	01	1	00

Tableau I.2 Code de Huffman

I.8.3. Codage Lempel-Ziv

C'est une technique de codage qui utilise un dictionnaire. On cherche dans le fichier les chaînes qui se répètent, puis on mémorise dans le dictionnaire. Ensuite, le codage consiste à remplacer les chaînes mémorisées par leur adresse (ou indice) construite dans le dictionnaire. Recherche de chaînes répétées différentes selon la version de l'algorithme. Il en existe trois versions [24].

- LZ77, version originale, la recherche s'effectue par une fenêtre glissante;
- LZ78, la recherche s'effectue sur tout le fichier. La taille du dictionnaire est limitée en fonction du mode de codage (16, 32, ou 64 bits) ;
- LZW, introduite en 1984, qui est brevetée par la société Unisys, est une amélioration de la LZ78. Le dictionnaire, initialement construit, contient l'ensemble des codes ASCII. Il est élaboré au fur à mesure, ce qui permet de changer la taille du dictionnaire au cours du codage.

I.8.4. Le Codage par plage (Run length Encoding)

Le procédé Run-Length ne relève pas d'une théorie mathématique très complexe. Il s'agit simplement de remplacer des éléments significatifs, successifs et identiques par un seul d'entre eux, suivi du nombre de répétitions [26]. Un exemple de traitement RLE est donné à la figure I.13. Ce procédé peut paraître simpliste et peu performant si on cherche à l'appliquer, par exemple, à un texte, les répétitions de lettres n'apporteraient qu'une compression dérisoire! En revanche, si on l'applique à une image, il est aisé de s'apercevoir que les plages de couleurs (ou niveaux de gris) homogènes sont souvent importantes, même si leur nombre est parfois faible.

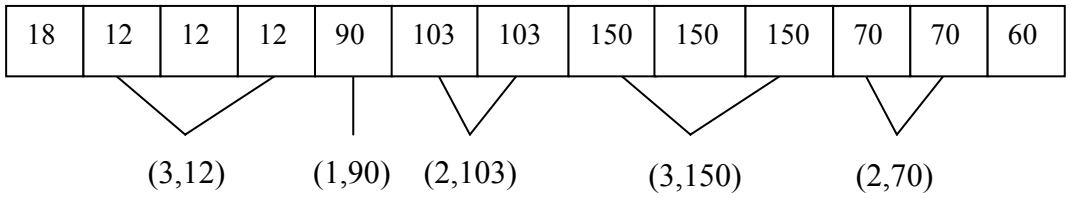


Figure I.13 Exemple de Codage par plage RLE

Si *n* octets successifs sont dans un même état, il est aisé de transmettre l'octet répété et le nombre de répétitions. On pourra ainsi, dans la plupart des cas, coder sur 3 octets les *n* octets composant le signal initial. S'il est relativement simple de coder l'octet à répéter, suivi du nombre de répétitions dans l'octet suivant, cette méthode peut se révéler très pénalisante pour certains cas : à la limite si deux octets consécutifs sont toujours différents, la taille des données compressées sera la double de celle des données initiales. Pour éviter cet inconvénient, les versions les plus avancées du RLE utilisent un code discriminant pour indiquer le début d'une séquence octet à répéter + le nombre de répétitions, les octets isolés restant codés sous leur forme initiale [26].

Dans la norme JPEG (Joint Photographic Experts Group) le ‘‘Codage par plage’’ se focalise sur les répétitions des ‘‘zéros’’ dans une séquence (ou un balayage). Ainsi les coefficients AC quantifiés dans un balayage subissent un codage par plage avant le passage par le codeur de Huffman. La figure I.14 représente l'implantation du ‘‘Codage par plage’’ des coefficients AC (les 63 coefficients de fréquence non nulle pour un bloc 8×8) dans la norme JPEG.

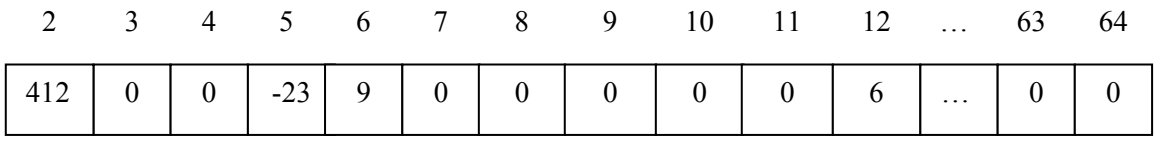


Figure I.14 Codage par plage des coefficients AC dans JPEG

La première ligne indique l'ordre du coefficient AC dans le balayage en zig-zag. La deuxième ligne indique la valeur du coefficient AC quantifié.

(0,412), (2,-23), (0,9), (5,6), ..., EOB

I.8.5. Codage différentiel

La première étape de codage de la matrice quantifiée est le codage différentiel pour le coefficient *DC* (la composante continue du bloc).

La modulation différentielle est basée sur le constat que les données analogiques ont tendance à varier de valeur d'une manière progressive, les sauts importants du signal étant exceptionnels. En effet, les blocs adjacents sont très fortement corrélés. Cette opération transforme donc les valeurs absolues en valeurs relatives. Pour les données graphiques, elle permet de transmettre non plus la valeur moyenne d'un bloc de pixels mais la différence de valeur avec le bloc précédent. Par exemple si la valeur moyenne est codée sur huit bits, un système de codage différentiel peut coder la différence sur quatre bits, ce qui permet de réduire le nombre de bits nécessaires au codage. Pour la compression des images fixes, le problème vient du fait que les valeurs des pixels peuvent varier de manière importante. Le codage différentiel doit donc s'adapter aux petites et aux grandes différences entre les pixels, ce qui limite son efficacité. Par exemple certaines images montrent de longues étendues de pixels de même valeur, elles seront bien compressées, tandis que d'autres, présentant des variations de couleur importantes, seront peu ou pas compressées [16].

I.9. Codage avec perte

Pour la compression avec perte (Lossy) en plus des trois axes de compromis pour la compression sans perte, on ajoute un quatrième axe qui est la qualité du signal : il est utilisé pour caractériser le signal à la sortie du décodeur. Plusieurs mesures sont proposées pour la qualité du signal parmi lesquelles le *SNR* (Signal-to-Noise Ratio), le *PSNR* (Peak-SNR), et le *MSE* (Mean Square Error) [25].

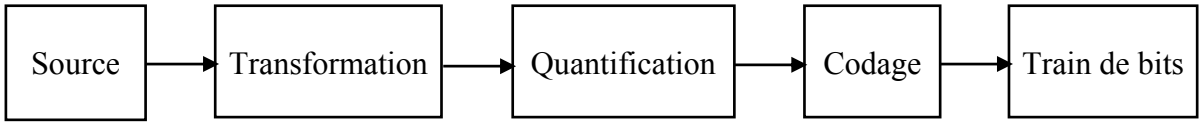


Figure I.15 Schéma classique d'un système de compression avec perte.

La Figure I.15 représente le schéma classique d'un système de compression avec perte.

Les méthodes avec pertes ou irréversibles sont des méthodes qui tirent parti d'une corrélation (ou redondance) existante dans l'image. L'information perdue est due à l'élimination de cette redondance, ceci rend possible une compression plus importante [19]. Pour améliorer le taux de compression, il va falloir perdre quelque chose c'est-à-dire dégrader l'image. Ceci permet

d'atteindre des taux arbitrairement grand au prix d'une dégradation toujours plus importante. L'objectif des algorithmes de compression avec perte est de minimiser cette dégradation pour un taux de compression donné [27].

Dans un premier temps, afin de mieux compacter l'information, la source est transformée en groupe des coefficients. Les transformations les plus utilisées, que ce soit pour les images fixes ou les séquences d'images, sont la Transformée en Cosinus Discrète (DCT), la Transformée en Ondelettes Discrète (DWT) ou la décomposition Pyramidale.

Dans un second temps, les coefficients obtenus après la transformation sont quantifiés. La phase de quantification introduit l'erreur dans le système de codage [24], ce dernier est un des mécanismes utilisés dans les algorithmes de compression, qui produit des pertes d'information [19].

I.9.1. Quantification

Par définition, l'opération de quantification permet une transcription des données depuis un espace de taille ‘‘infini’’ constitué par exemple de l'ensemble des nombres flottants, ou de taille très importante, vers un espace contenant un nombre limité de coefficients [28]. La quantification fait partie de plusieurs méthodes de compression d'image. L'objectif est de réduire la taille des coefficients de façon que cette réduction n'apporte pas de dégradations visuelles à l'image [19].

Plusieurs familles de quantification existent ; elles travaillent soit coefficient par coefficient, soit par groupe de coefficients. Dans le premier cas on parle de quantification scalaire, et dans le second cas la méthode est la suivante : on alloue conjointement un vecteur-représentant à cet ensemble de coefficients. On parle alors de codage de type vectoriel. Le choix de ces représentants vectoriels ou scalaires et leur répartition a donné lieu à de nombreux algorithmes de quantification [28].

I.9.1.1. Quantification Scalaire

La quantification scalaire *SQ*-(Scalar Quantization) est réalisée indépendamment pour chaque élément. D'une manière générale, on peut la définir comme étant l'association de chaque valeur réelle x, à une autre valeur q qui appartient à un ensemble fini de valeurs. La valeur q peut être exprimée en fonction de la troncature utilisée : soit par l'arrondi supérieur, l'arrondi inférieur, ou l'arrondi le plus proche [24].

I.9.1.2. Quantification Vectorielle

La quantification vectorielle *VQ*-(Vector Quantization) a été développée par Gersho et Gray et elle fait aujourd'hui l'objet de nombreuses publications dans le domaine de la compression numérique [29]. Le principe de la quantification vectorielle est issu du travail de Shanon qui montre qu'il était toujours possible d'améliorer la compression de données en codant non pas des scalaires, mais des vecteurs. Un quantificateur vectoriel Q associe à chaque vecteur d'entrée $X_i = (x_j, j = 1...k)$ un vecteur $Y_i = (y_j, j = 1...k) = Q(X_i)$, ce vecteur Y_i étant choisi parmi un dictionnaire (code book) de taille finie. La *VQ* produit de meilleurs résultats que la *SQ*, néanmoins la *VQ* nécessite un codage complexe et de grandes capacités de mémoire [19].

I.10. Formats d'images

Il existe plus d'une cinquantaine de types de formats d'image. Pour chacun d'entre eux la structuration des données et les attributs sont différents. La standardisation d'un format d'image permet de régler l'utilisation, la divulgation et la production de logiciels et de hardware compatibles avec le format standard. Le plus populaire de ces standards est le JPEG, il a été créé vers la fin des années 80. JPEG utilise les principaux modes suivants : baseline, lossless, progressive et hiérarchique. Le mode baseline est le plus utilisé, il supporte le codage avec perte seulement. Le mode lossless est moins populaire il ne supporte pas le mode avec perte.

JPEG2000 est le dernier standard ISO/ITU-T pour le codage des images fixes, est basé sur la DWT, la quantification scalaire, la modélisation du contexte, le codage arithmétique et l'allocation débit post-compression.

Le JPEG2000 possède des fonctionnalités supplémentaires par rapport au format JPEG.

La résistance aux erreurs est une caractéristique particulière du JPEG2000. Après le codage entropique plusieurs caractères de contrôle (segment marks, resynchronising marks) sont insérés dans le flux de bits. Cette démarche est faite pour synchroniser les informations, limiter la taille du segment et éviter la propagation des erreurs.

Une autre fonctionnalité importante du JPEG2000 est la compression par région d'intérêt (ROI). Ceci permet d'avoir des taux de compression différents dans certaines régions de l'image. Les zones importantes peuvent être compressées quasi sans pertes et les zones moins importantes avec un fort taux de compression.

Malgré ses nombreuses fonctionnalités, le JPEG2000 possède quelques inconvénients.

Il nécessite entre deux et six fois plus de cycles de CPU que JPEG et il n'est pas indiqué pour les machines avec faibles ressources, la plupart des appareils numériques (appareils photos, caméscopes, téléphones portable, etc.) et les logiciels qui capturent et traitent les images sont au format JPEG. L'algorithme JPEG est beaucoup moins complexe et il peut être implémenté en hardware [19] [25].

I.11. Conclusion

Dans ce chapitre, nous avons présenté plusieurs techniques de compression d'images sans perte et avec pertes. Tout d'abord, nous avons donné quelques concepts et mesures pour la compression d'images et aussi la définition d'une image médicale. Toutes les approches énumérées précédemment utilisent d'une manière ou d'une autre les corrélations entre pixels voisins dans l'image. Le codage par plage c'est un codage simple, rapide et utilisé par de nombreux formats d'images comme le BMP, TIFF et JPEG par exemple. Suite à cette simplicité, nous avons opté dans notre travail pour le codage par plage pour effectuer la compression des images ou en utilisant les méthodes proposées dans le chapitre III.

Dans le chapitre suivant, nous allons présenter une méthode de compression par décimation en utilisant les réseaux de neurones artificiels.

Chapitre II

Réseaux de neurones

II.1. Introduction

Dans ce chapitre nous allons présenter les réseaux de neurones à apprentissage supervisé, et exposer les méthodes les plus utilisées pour l'apprentissage de ce type de réseaux qui nous permettront d'entamer notre étude.

On présentera plus particulièrement le réseau à couches multiples MLP (Multi Layer Perceptron) en association avec la méthode d'apprentissage de rétropropagation du gradient de l'erreur, communément appelée ''back-propagation''.

C'est, à ce jour, la technique qui donne les meilleurs résultats dans plusieurs domaines d'application en utilisant l'apprentissage supervisé. Le réseau RBF (Radial Basis Function) à apprentissage supervisé est également présenté en détail.

Une problématique importante se situe au niveau de la configuration des réseaux de neurones dont la théorie est inexistante jusqu'à ce jour. C'est un domaine de recherche largement ouvert mais une nouvelle technique provenant des plans d'expériences permettrait toutefois d'optimiser la configuration d'un réseau de neurones selon le contexte du problème [30].

Dans ce chapitre nous allons utiliser les réseaux MLP et RBF pour reconstituer les images au niveau de gris et couleur compressée par la méthode simple qui est la décimation.

II.2. Réseaux de neurones MLP

II.2.1. Architecture et fonctionnement du réseau multicouche

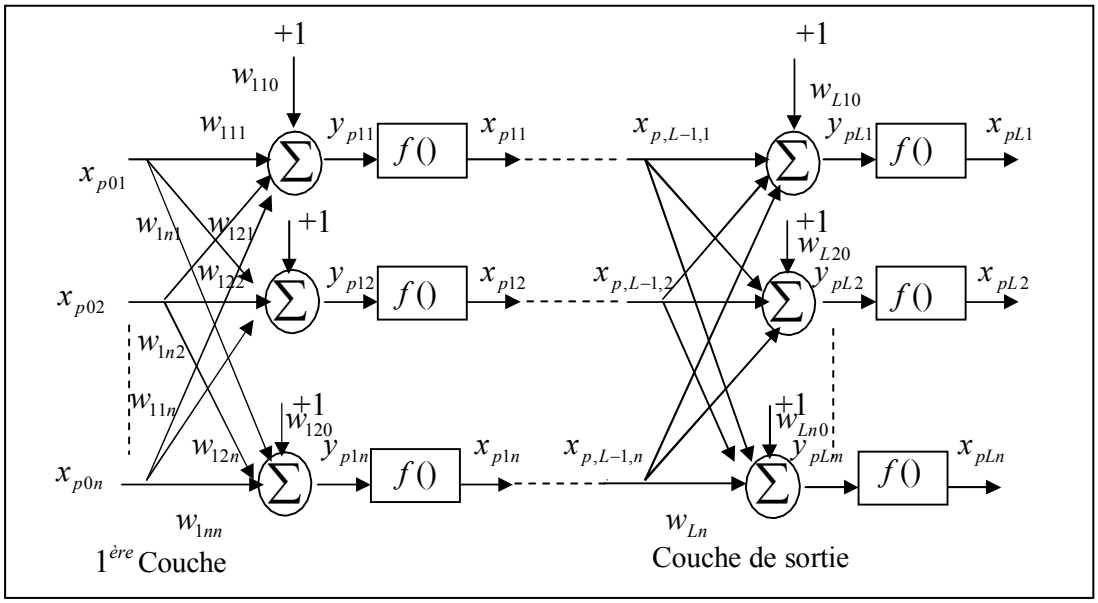


Figure II.1 Structure d'un réseau de neurone multicouche

La topologie d'un tel réseau est formée de plusieurs couches de neurones sans communication à l'intérieur d'une même couche (figure II.1).

- une couche en entrée qui représente les entrées auxquelles sont transmises les données à traiter en provenance d'une source extérieure au réseau ;
- une ou plusieurs couches cachées effectuant le traitement spécifique du réseau ;
- une en sortie qui délivre les résultats.

L'apprentissage est supervisé, c'est-à-dire que l'on présente au réseau, en même temps, une forme et son modèle. La fonction de transfert utilisée est une fonction sigmoïde, dont la dérivabilité joue un rôle important.

L'apprentissage dans ce type de structure consiste à appliquer des couples (entrées, sorties désirées) à l'entrée du réseau.

Une sortie réelle est calculée pour chaque neurone de la $j^{ème}$ couche. Ce calcul est effectué de proche en proche de la couche d'entrée vers la couche de sortie, celle-ci est appelée ''propagation d'avant''. Ensuite l'erreur est calculée puis propagée dans le réseau, donnant lieu à une modification des poids.

On considère un réseau comportant une couche d'entrée à *n* neurones, une couche de sortie à *m* neurones et il comporte une à plusieurs couches cachées.

Supposons qu'on dispose d'un ensemble d'apprentissage composé de *k* pair de vecteurs :

$$(x_1, o_1), (x_2, o_2), ..., (x_k, o_k)$$

avec :

$x_p = (x_{p,0,1}, x_{p,0,2}, ..., x_{p,0,n})^t \in R^n$ Vecteur d'entrée.

$O_p = (O_{p,1}, O_{p,2}, ..., O_{p,m})^t \in R^m$ Vecteur des sorties désiré.

$y_p = (y_{p,L,1}, y_{p,L,2}, ..., y_{p,L,n})^t \in R^n$ Vecteur des sorties réel du réseau.

Où :

$w_{j,k,i}$: La connexion entre le neurone *k* de la couche *j-1* et le neurone *i* de la couche *j*.

$y_{p,j,k}$: L'entrée totale du neurone *k* pour l'échantillon *p* de la couche *j*.

$w_{j,k,0} = \theta_{j,k}$: Le poids fictif du neurone *k* de la couche *j* correspondant à un biais dont

l'entrée est fixé à 1.

L'entrée totale du *k* nœud pour la couche *j* est :

$$y_{p,j,k} = \sum_{i=0}^{n} w_{j,k,i} x_{p,j-1,i} \tag{II.1}$$

La sortie de ce nœud sera :

$$x_{p,j,k} = F(y_{p,j,k}) \tag{II.2}$$

où *F* est une fonction de transfert sigmoïde.

II.2.2. Fonctions de transfert

Différentes fonctions de transfert pouvant être utilisées comme fonction d'activation du neurone sont énumérées au tableau II.1. Les trois les plus utilisées sont les fonctions ''seuil'', ''linéaire'' et ''sigmoïde'' [31].

Comme son nom l'indique, la fonction seuil applique un seuil sur son entrée. Plus précisément, une entrée négative ne dépasse pas le seuil, la fonction retourne alors la valeur 0 (on peut interpréter ce 0 comme signifiant faux), alors qu'une entrée positive ou nulle dépasse le seuil, et la fonction retourne 1 (vrai). Utilisée dans le contexte d'un neurone, cette fonction permet de prendre des décisions binaires.

La fonction linéaire est très simple, elle affecte directement son entrée à sa sortie :

$$a = n$$

La fonction de transfert sigmoïde est quant à elle illustrée à la figure II.2c. Son équation est donnée par :

$$a = \frac{1}{1+e^{-n}}$$

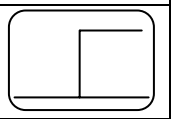
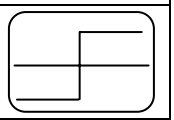
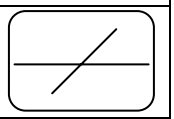
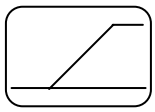
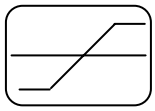
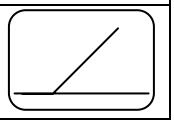
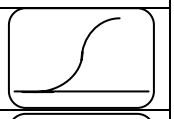
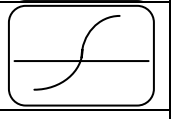
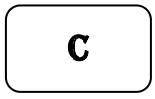
Nom de la fonction	Relation d'entrée / sortie	Icône
Seuil	$a = 0$ si $n < 0$ $a = 1$ si $n \geq 0$	
seuil symétrique	$a = -1$ si $n < 0$ $a = 1$ si $n \geq 0$	
Linéaire	$a = n$	
Linéaire saturée	$a = 0$ si $n < 0$ $a = n$ si $0 \leq n \leq 1$ $a = 1$ si $n > 1$	
Linéaire saturée symétrique	$a = -1$ si $n < -1$ $a = n$ si $-1 \leq n \leq 1$ $a = 1$ si $n > 1$	
Linéaire positive	$a = 0$ si $n < 0$ $a = n$ si $n \geq 0$	
Sigmoïde	$a = \frac{1}{1+e^{-n}}$	
Tangente hyperbolique	$a = \frac{e^{n}-e^{-n}}{e^{n}+e^{-n}}$	
Compétitive	$a = 1$ si n maximum $a = 0$ autrement	

Tableau II.1 Fonctions de transfert *a=f(n)*

Elle ressemble soit à la fonction seuil, soit à la fonction linéaire, selon que l'on est loin ou près de *b* (biais), respectivement. La sigmoïde est un compromis intéressant entre les deux précédentes. Notons finalement, que la fonction ‘‘tangente hyperbolique’’ est une version symétrique de la sigmoïde.

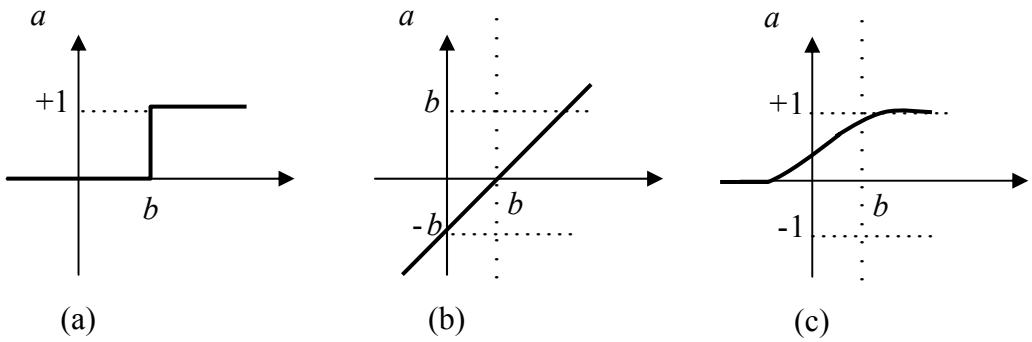


Figure II.2 Fonction de transfert : (a) du neurone ‘‘seuil’’; (b) du neurone ‘‘linéaire’’, et (c) du neurone ‘‘sigmoïde’’

II.2.3. Mise en œuvre des réseaux neuronaux

Nous allons suivre une démarche reprise par Wierenga et Kluytmans (1994) qui est composée de quatre étapes principales :

Etape 1 : fixer le nombre de couches cachées

Mis à part les couches d’entrée et de sortie, l’analyste doit décider du nombre de couches intermédiaires ou cachées. Sans couche cachée, le réseau n’offre que de faibles possibilités d’adaptation; avec une couche cachée, il est capable, avec un nombre suffisant de neurones, d’approximer toute fonction continue (Hornik, 1991). Une seconde couche cachée prend en compte les discontinuités éventuelles.

Etape 2 : déterminer le nombre de neurones par couches cachées

Chaque neurone supplémentaire permet de prendre en compte des profils spécifiques des neurones d’entrée. Un nombre plus important permet donc de mieux coller aux données présentées mais diminue la capacité de généralisation du réseau. Ici non plus il n’existe pas de règle générale mais des règles empiriques.

Etape 3 : choisir la fonction d’activation

Nous considérerons la fonction logistique pour le passage de la couche d’entrée à la couche cachée. Le passage de cette dernière à la couche de sortie sera soit linéaire, soit sigmoïde (logistique) selon nos types de variables.

Etape 4 : choisir l’apprentissage

L’apprentissage par rétro-propagation nécessite la détermination du paramètre d’ajustement des poids synaptiques à chaque itération.

La détermination du critère d’arrêt est aussi cruciale dans la mesure où la convergence peut

passer par des minima locaux.

II.2.4. Algorithmes d'apprentissage

II.2.4.1. Rétropropagation du gradient

L'algorithme de rétro-propagation a été développé en particulier par Rumelhart et Parkenet le Cun. Cet algorithme repose sur la minimisation de l'erreur quadratique entre les sorties calculées et celles souhaitées.

Le terme rétro-propagation du gradient provient du fait que l'erreur calculée en sortie est transmise en sens inverse vers l'entrée.

L'erreur commise sur le $k^{ème}$ nœud de sortie est :

$$\delta_{pk} = O_{pk} - x_{plk} \tag{II.3}$$

Par conséquent l'erreur totale (pour tous les nœuds) est :

$$E_p = \frac{1}{2}\sum_{k=1}^{m} \delta^2{}_{p,k} = \frac{1}{2}\sum_{k=1}^{m} (O_{p,k} - x_{p,l,k})^2 \tag{II.4}$$

Pour minimiser E_p, on calcule son gradient par rapport à chaque poids w, puis on modifie les poids dans le sens inverse du gradient.

Mise à jour des poids de la couche de sortie

$$\nabla E_p = \frac{\partial E_p}{\partial w_{l,k,j}} = \frac{1}{2}\frac{\partial (O_{p,k} - x_{p,l,k})^2}{\partial w_{l,k,j}} \tag{II.5}$$

$$= -(O_{p,k} - x_{p,l,k})\frac{\partial}{\partial w_{l,k,j}}[f(y_{p,l,k})]$$

$$\nabla E_p = -(O_{p,k} - x_{p,l,k})\frac{\partial f(y_{p,l,k})\partial (y_{p,l,k})}{\partial w_{l,k,j}\partial (y_{p,l,k})} \tag{II.6}$$

$$\begin{cases} \dfrac{\partial y_{p,l,k}}{\partial w_{l,k,j}} = \dfrac{\partial}{\partial w_{l,k,j}}\displaystyle\sum_{j=0}^{m}(w_{l,k,j}x_{p,l-1,j}) = x_{p,l-1,j} & \text{(II.7)} \\ \dfrac{\partial f(y_{p,l,k})}{\partial y_{p,l,k}} = f'(y_{p,l,k}) & \text{(II.8)} \end{cases}$$

$$\nabla E_p = -(O_{p,k} - x_{p,l,k}) f'(y_{p,l,k}) x_{p,l-1,j}$$
$$= \delta_{p,k} x_{p,l,k} (1 - x_{p,l,k}) x_{p,l-1,j} \quad \text{(II.9)}$$

La modification des poids est fonction du calcul du gradient. Ainsi, les poids sur la couche des sorties sont mis à jour de la façon suivante :

$$w_{l,k,j}(t+1) = w_{l,k,j}(t) + \Delta_p w_{l,k,j}(t) \quad \text{(II.10)}$$

$$\Delta_p w_{l,k,j}(t) = \mu (O_{p,k} - x_{p,l,k}) f'(y_{p,l,k}) x_{p,l-1,j} \quad \text{(II.11)}$$

Ou :

μ : Pas d'apprentissage $0 < \mu < 1$

Le taux d'apprentissage, un des paramètres de cet algorithme, ne doit pas être trop grand sinon il entraînerait des oscillations de l'erreur autour d'un minimum qu'on ne pourra pas atteindre et si μ est trop petit le temps d'apprentissage serait trop grand.

On pose :

$$e_{p,l,k} = (O_{p,k} - x_{p,l,k}) f'(y_{p,l,k}) \quad \text{(II.12)}$$

où :

$e_{p,l,k}$ Erreur de signal du $k^{ème}$ nœud de la couche de sortie.

L'équation des modifications des poids aura donc la forme suivante :

$$w_{l,k,j}(t+1) = w_{l,k,j}(t) + \mu e_{p,l,k} x_{p,l-1,k} \quad \text{(II.13)}$$

Mise à jour des poids des couches cachées

Le procédé peut être appliqué aux couches cachées .Cependant un obstacle survient lors du calcul de l'erreur des sorties des nœuds cachées. Cette limitation provient du fait que les sorties désirées ne sont pas connues.

Pour s'affranchir de cet obstacle, nous devons développer un terme d'erreur à la sortie des nœuds cachés.

Nous n'avons aucune idée à l'avance de ce que peut être la sortie correcte ou désirée pour ces nœuds.

Pour cela, nous développons le terme de l'erreur à la sortie du réseau :

$$\begin{aligned} E_p &= \frac{1}{2}\sum_{k=1}^{m}(O_{p,k} - x_{p,l,k})^2 \\ &= \frac{1}{2}\sum_{k=1}^{m}[O_{p,k} - f(y_{p,l,k})]^2 \\ &= \frac{1}{2}\sum_{k=1}^{m}[O_{p,k} - f(\sum_{j=0}^{n} w_{l,k,j}x_{p,l-1,j})]^2 \end{aligned} \tag{II.14}$$

Nous pouvons exploiter le fait que $x_{p,l-1,j}$ dépend des poids de la couche cachée à travers l'équation suivante :

$$x_{p,l-1,j} = f(\underbrace{\sum_{i=0}^{n} w_{l-1,j,i}x_{p,l-2,i}}_{y_{p,l-1,j}}) \tag{II.15}$$

Pour évaluer le gradient de E_p par rapport aux poids des couches cachées.

$$\begin{aligned} \frac{\partial E_p}{\partial w_{l-1,j,i}} &= \frac{1}{2}\sum_{k=1}^{m}\frac{\partial(O_{p,k} - x_{p,l,k})^2}{\partial w_{l-1,j,i}} \\ &= -\sum_{k=1}^{m}(O_{p,k} - x_{p,l,k})\frac{\partial x_{p,l,k}}{\partial y_{p,l,k}}\frac{\partial y_{p,l,k}}{\partial y_{p,l-1,j}}\frac{\partial y_{p,l-1,j}}{\partial w_{l-1,j,i}} \\ &= -\sum_{k=1}^{m}(O_{p,k} - x_{p,l,k})\frac{\partial x_{p,l,k}}{\partial y_{p,l,k}}\frac{\partial y_{p,l,k}}{\partial x_{p,l-1,j}}\frac{\partial x_{p,l-1,j}}{\partial y_{p,l-1,j}}\frac{\partial y_{p,l-1,j}}{\partial w_{l-1,j,i}} \end{aligned} \tag{II.16}$$

Chacun des facteurs de l'équation (II.16) peut être calculé explicitement :

$$\frac{\partial x_{p,l,k}}{\partial y_{p,l,k}} = \frac{\partial f(y_{p,l,k})}{\partial y_{p,l,k}} = f'(y_{p,l,k})$$

$$\frac{\partial y_{p,l,k}}{\partial x_{p,l-1,j}} = \frac{\partial(\sum_{j=0}^{n} w_{l,k,j}x_{p,l-1,j})}{\partial x_{p,l-1,j}} = w_{l,k,j}$$

$$\frac{\partial y_{p,l-1,j}}{\partial w_{l-1,j,i}} = x_{p,l-2,i}$$

$$\frac{\partial x_{p,l-1,j}}{\partial y_{p,l-1,j}} = \frac{\partial f(y_{p,l-1,j})}{\partial y_{p,l-1,j}} = f'(y_{p,l-1,j})$$

Le résultat est le suivant :

$$\frac{\partial E_p}{\partial w_{l-1,j,i}} = -\sum_{k=1}^{m}(O_{p,k} - x_{p,l,k}) f'(y_{p,l,k}) w_{l,k,j} f'(y_{p,l-1,j}) x_{p,l-2,i} \quad \text{(II.17)}$$

La mise à jour des poids de la couche cachée se fait dans le sens inverse du gradient en utilisant l'équation précédente :

Nous pouvons utiliser la définition de $e_{p,l,k}$ de (II.12) pour écrire :

$$\Delta w_{l,k,j} = \mu f'(y_{p,l-1,j}) x_{p,l-2,i} \sum_{k=1}^{m}(O_{p,k} - x_{p,l,k}) f'(y_{p,l,k}) w_{l,k,j} \quad \text{(II.18)}$$

avec μ : taux d'apprentissage.

$$\Delta_p w_{l,k,j} = \mu f'(y_{p,l-1,j}) x_{p,l-2,i} \sum_{k=1}^{m} e_{p,l,k} w_{l,k,j} \quad \text{(II.19)}$$

Notons que la mise à jour de chaque poids de la couche cachée dépend de toutes les erreurs de signal $e_{p,l,k}$ sur la couche de sortie. En définissant le terme de l'erreur des couches cachées :

$$e_{p,l-1,j} = f'(y_{p,l-1,j}) \sum_{k=1}^{m} e_{p,l,k} w_{l,k,j} \quad \text{(II.20)}$$

Alors l'équation de mise à jour des poids de la couche cachée devient :

$$w_{l-1,j,i}(t+1) = w_{l-1,j,i}(t) + \mu e_{p,l-1,j} x_{p,l-2,i} \quad \text{(II.21)}$$

II.2.4.2. Résumé de l'algorithme de Rétropropagation

1. Appliquer un vecteur d'entrée $x_p = (x_{p,0,1}, x_{p,0,2}, \ldots\ldots\ldots\ldots x_{p,0,n})^t$ aux nœuds d'entrées puis initialiser les poids du réseau;

2. Exécuter l'échantillon d'apprentissage à travers le réseau;

3. Calculer les termes d'erreur de signal de la couche de sortie et les couches cachées en utilisant (II.12) et (II.20) respectivement;

4. Mise à jour les poids de la couche de sortie et couches cachées en utilisant (II.13) Et (II.21) respectivement;

5. Répéter ce processus jusqu'à ce que l'erreur E_P devienne acceptable (aller à 2).

II.2.4.3. Considérations pratiques

- les poids du réseau doivent être initialisés à de petites valeurs aléatoires.
- La valeur du taux d'apprentissage μ à un effet significatif sur les performances du réseau, si ce taux est petit l'algorithme converge lentement, par contre s'il est grand l'algorithme risque de générer des oscillations.
- Généralement, μ doit être compris entre 0 et 1 pour assurer la convergence de l'algorithme vers une solution optimale.
- Il n'existe pas de règles permettant de déterminer le nombre de couches cachées dans un réseau donné n_i le nombre de neurones dans chacune d'elles.
- Théoriquement, l'algorithme ne doit se terminer dès que le minimum de l'erreur commise par le réseau sera atteint, correspondant à un gradient nul, ce qui n'est jamais rencontré en pratique. C'est pourquoi un seuil est fixé à priori afin d'arrêter l'apprentissage.

II.2.4.4. Accélération de l'algorithme avec le momentum

La convergence du réseau par rétropropagation est un problème crucial car il requiert de nombreuses itérations. Pour pallier à ce problème, un paramètre est souvent rajouté pour accélérer la convergence. Ce paramètre est appelé le momentum.

Le momentum est un moyen efficace pour accélérer l'apprentissage et aussi pour pouvoir sortir des minimums locaux.

La règle de mise à jour des poids devient alors :

$$w_{j,k,i}(t+1) = w_{j,k,i}(t) + \mu e_{p,j,k} x_{p,j-1,i} + m[w_{j,k,i}(t) - w_{j,k,i}(t-1)] \qquad \text{(II.22)}$$

avec m : est la constante du momentum.

II.3. Réseaux de neurones à fonction de base radiale (RBF)

Les réseaux de neurones à fonction de base radiale (Radial Basis Functions) sont des réseaux de neurones à une seule couche cachée dont les fonctions d'activation sont des fonctions à base radiale, le plus souvent des Gaussiennes. La fonction d'activation du neurone de la couche de sortie est l'identité. Les entrées sont directement connectées aux neurones de la couche cachée [32].

II.3.1. Formalisme

L'approximation de fonction non linéaire peut également s'effectuer à l'aide d'une somme de fonctions noyaux (''kernels''). Si ces noyaux sont fixés en largeur et position, la sortie dépend linéairement des poids. Pour une fonction continue φ d'une variable vectorielle Y, son estimation par une somme de N_c noyaux s'écrit [33] :

$$\varphi(Y) = w_0 + \sum_{i=1}^{N_c} w_i \Phi(\|Y_i - C_i\|) \tag{II.23}$$

où Φ désigne la fonction noyau, w_i les coefficients de pondération et C_i le $i^{ème}$ noyau.

Le modèle de neurone à fonction radiale de base a été introduit par plusieurs auteurs comme Moody et Darken, Musavi et al. Chaque neurone élémentaire, s'identifie à un noyau dans l'expression II.23 : il calcule la distance entre l'entrée et son centre qu'il fait passer ensuite dans une non linéarité Φ (figure II.3). Dans ce modèle, l'opérateur Distance (ou Norme) remplace l'opération Produit Scalaire du neurone formel.

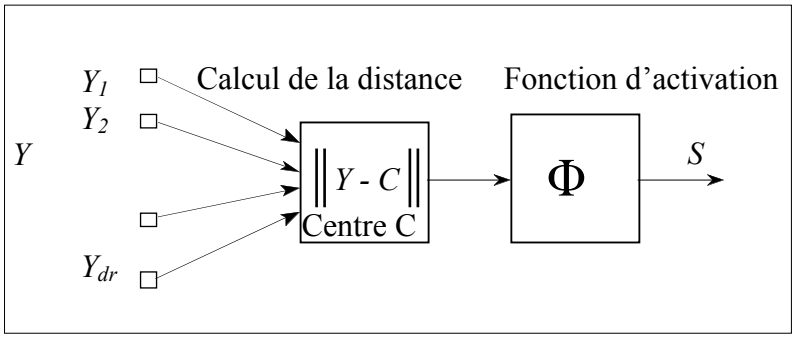


Figure II.3 Neurone élémentaire à noyau

La sortie S du neurone s'écrit finalement sous la forme :

$$S = \Phi(\|Y - C\|) \tag{II.24}$$

Dans cette expression, l'opérateur norme est considérée dans sa forme généralisée définie par :

$$\|X\|^2 = (X^t A X)^t \tag{II.25}$$

où A est une matrice de normalisation définie positive.

La sortie du neurone vaut donc :

$$S = \Phi(((Y-C)^t A(Y-C))^{-1/2}) \quad \text{(II.26)}$$

Si A est la matrice identité, on retrouve la norme euclidienne. Si A est l'inverse de la matrice de variance-covariance d'un groupe d'observation, il s'agit alors de la norme de Mahalanobis. Les noyaux utilisés comme fonction d'activation sont des fonctions définies de R vers R^+, symétriques radialement par rapport à un point (d'où la dénomination de neurones à fonctions radiales de base) parmi lesquelles on peut citer :

- noyau gaussien $\Phi(v) = \exp(-\frac{v^2}{2\beta^2})$
- noyau ''thin plate'' $\Phi(v) = v^2 \log(v)$
- noyau multiquadratique $\Phi(v) = \sqrt{(v^2 + \beta^2)}$

Le noyau gaussien est le plus largement répandu. La valeur que prend sa sortie est d'autant plus importante que l'entrée est plus proche de son centre et elle tend vers zéro lorsque la distance entrée-centre devient importante. Le paramètre β permet de contrôler la vitesse de décroissance de la fonction Φ et il conviendra de le choisir de façon judicieuse.

L'expression complète de la sortie du neurone RBF à noyau gaussien est :

$$\Phi(Y) = \exp(-\frac{\|Y-C\|^2}{2\beta^2}) = \exp(-\frac{1}{2\beta^2}(Y-C)^t A(Y-C)) \quad \text{(II.27)}$$

L'architecture d'un réseau RBF s'organise en deux couches seulement : une couche cachée et une couche de sortie. La première couche, constituée de N_c noyaux élémentaires, effectue une transformation non linéaire de l'espace d'entrée. La couche de sortie calcule une combinaison linéaire des sorties des noyaux élémentaires.

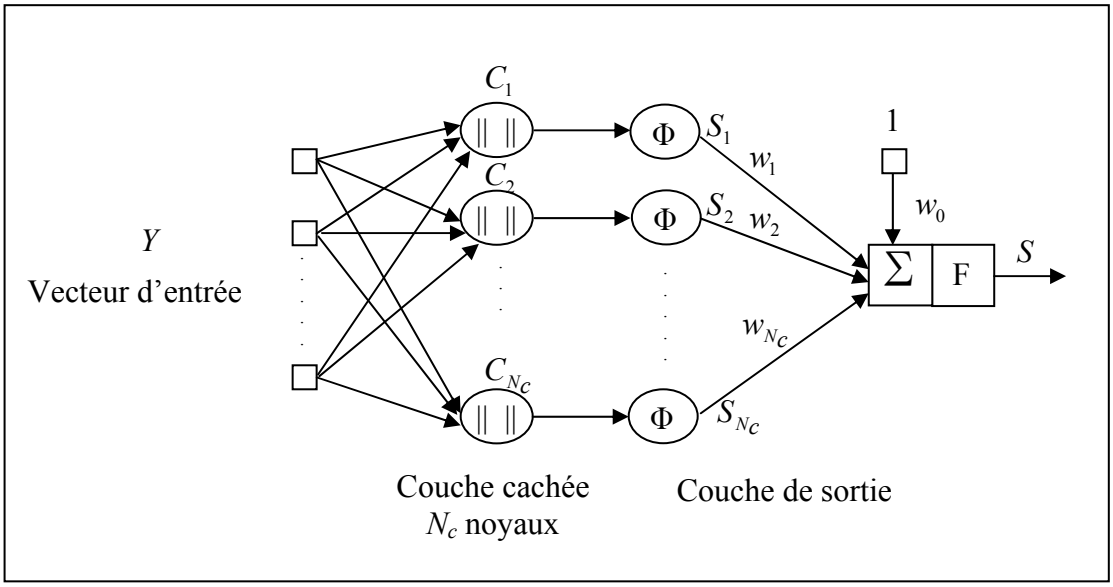


Figure II.4 Architecture d'un réseau RBF

La sortie d'un tel réseau s'exprime sous la forme :

$$S = F(w_0 + \sum_{i=1}^{N_c} w_i \Phi(\|Y - C_i\|)) \tag{II.28}$$

Cette relation s'identifie à la relation II.23, la fonction d'activation en sortie est linéaire *F(x)=x*.

Les perceptrons multicouches MLP et les réseaux RBF sont tous deux en mesure d'approximer n'importe quelle fonction non linéaire, mais il convient de noter quelques différences entre ces deux types de réseaux. La première porte sur leur architecture : tandis qu'elle est figée en deux couches pour les RBF, l'architecture d'un MLP peut comporter plusieurs couches cachées. Le deuxième point fondamental qui les différencie est la nature des réponses qu'ils sont en mesure de fournir. En effet, de par leur fonction d'activation (Gaussiennes pour les RBF et sigmoïdales pour les MLP) [33].

II.3.2. Apprentissage des réseaux RBF

Les paramètres ajustables dans un réseau RBF tel que présenté par la figure II.4 sont :

- la position des centres C_i $(1 \le i \le N_c)$.
- l'optimisation du nombre N_c de noyaux.
- la valeur de l'écart-type associé à chaque noyau.
- les poids de la couche de sortie *w*.

Le type de fonctions noyaux sera toujours fixé avant l'apprentissage et dans notre cas, il s'agira de noyaux Gaussiens.

II.3.3. Choix de la métrique et de la largeur des noyaux

Dans un réseau RBF, l'ajustement de l'écart-type β_i [33] associé à chaque noyau permet de contrôler le chevauchement des différentes fonctions élémentaires. S'il est choisi trop petit, la décroissance de la fonction d'activation autour du centre est rapide; des exemples très proches du centre peuvent conduire à des réponses nulles. A l'inverse, si la largeur du noyau est trop grande, des observations même situées très loin du centre contribuent significativement à la sortie. Dans les deux cas, cela se traduira par une mauvaise généralisation du réseau.

Nous allons distinguer trois cas différents de choix du β_i .avec la matrice A_i (decovariance) sera la matrice identité, conduisant donc à un emploi d'une métrique euclidienne.

Cas 1, l'écart-type sera constant pour tous les noyaux et déduit de la valeur maximale des distances séparant tous les noyaux par la relation :

$$\beta_i = \beta = \frac{d_{\max}}{\sqrt{2N_c}} \qquad \forall i \qquad 1 \le i \le N_c \tag{II.29}$$

Cas 2, la valeur de l'écart-type est directement la valeur moyenne des distances séparant tous les noyaux :

$$\beta_i = \beta = d_{moy} \qquad \forall i \qquad 1 \le i \le N_c \tag{II.30}$$

Cas 3, Chaque noyau possède son propre écart-type. Cette approche est particulièrement intéressante si une méthode de type ‘‘clustering’’ est employée pour le positionnement des noyaux. Si le groupe d'observations rattaché à un noyau est dispersé, son écart-type doit être suffisamment important pour pouvoir englober tous les exemples. Tandis que si les formes sont localisées autour du centre, la largeur du noyau doit être petite. D'où l'idée d'ajuster β_i comme la moyenne des distances entre le centre du noyau et les éléments du groupe ou ‘‘cluster’’ qui lui sont rattachés :

$$\beta_i = \frac{1}{N_{CL_i}} \sum_{Y \in clusteri} (Y_i - C_i)^t (Y_i - C_i) \qquad \text{(II.31)}$$

où N_{CLi} est le nombre d'observations dans le $i^{ème}$ groupe de centre C_i.

Il existe évidemment de nombreuses autres variantes de réseaux de neurones [34] [35] [36] [37] mais elles ne sont que très partiellement utilisées. Ce sont des structures moins connues, citons les réseaux de Hopfield ou de Hamming qui n'est pas du type ''propagation directe''. Ils nécessitent plus de temps de calcul et leur analyse est moins directe. Les réseaux de Kohonen sont, quant à eux, utilisés principalement en classification.
Dans cette étude, nous nous sommes limités aux réseaux de neurones de type MLP et RBF. Ils se prêtent le mieux à notre application. De plus, la relative facilitée avec laquelle on peut analyser leur fonctionnement dans le cadre de la reconstruction d'image compressée.

Les avantages suivants du RBF par rapport au MLP avec l'algorithme de rétropropagation ont été expérimentalement et théoriquement avérés :

- L'apprentissage dans RBF est plus rapide que l'apprentissage telle que le MLP.
- Une meilleure généralisation est réalisée dans RBF.
- RBF ont les propriétés très rapides de convergence comparées aux réseaux multicouche conventionnels avec des fonctions de transfert sigmoïdes.
- Il n'y a aucun problème de minimum local.
- La couche cachée a une interprétation beaucoup plus claire que le MLP avec l'algorithme de rétropropagation. Il est plus facile d'expliquer ce qu'un réseau de RBF a appris que ses contre parties MLP avec l'algorithme de rétropropagation
- Il y a également des inconvénients à employer le RBF, l'un d'entre eux trouvant le nombre approprié de nœuds cachés. L'étude non surveillée pourrait être nécessaire pour s'appliquer d'abord et pour découvrir le nombre de faisceaux. Le nombre de nœuds cachés est alors placé pour être égal à ce nombre. Un trop grand nombre, ou un trop petit nombre des nœuds cachés empêchera RBF d'approximer correctement les données [38].

Dans la littérature scientifique il y a plusieurs applications des réseaux de neurones (MLP RBF) [34] [39] [40] [41] [42].

II.4. Applications des réseaux de neurones sur les images

II.4.1. Prétraitement des données

Un problème intéressant en traitement numérique d'image est la restauration d'images compressée, notre problème consistera donc à récupérer une image proche de l'image originale à partir d'une image compressée pour cela nous proposons deux algorithmes utilisant les réseaux de neurones artificiels à savoir le perceptron multicouche (MLP) et les réseaux de neurones à fonction radiales de base (RBF).

Au début, consistaient à utiliser un filtre passe-bas ce qui avait pour l'avantage de lissage d'image (un filtrage suivi d'une décimation).

La réduction d'une image consiste à diminuer le nombre de pixels qui la composent.

Dans le cas d'une réduction d'un facteur 2 (50% de l'image originale), l'image réduite contiendra ainsi quatre fois moins de pixels que l'image initiale (taux de compression 4). Cela conduit à une disparition de structures contenues dans l'image initiale et, si l'on n'y prend garde, à l'apparition de distorsions indésirables. La réduction est un processus avec pertes, et il faudra s'efforcer de conserver le maximum de l'information contenue dans l'image initiale, tout en maintenant une bonne qualité visuelle.

L'idée de compression est faite par décimation c'est-à-dire l'élimination d'une ligne par ligne et colonne par colonne. Nous exploitons bien ce principe sur l'image filtrée jusqu'un taux de compression égale 4 (facteur de réduction 2).

Si le taux de compression est égal à 16 (facteur de réduction 4) l'élimination peut être faite sur deux lignes et deux colonnes, puisque une image plus réduite, son taux de compression est meilleur que celui de la version précédente, la réduction est relativement identique, le tableau II.2 illustre le principe de la compression avec perte par réduction sur l'image ‘‘Lena’’ au niveau de gris.

avec le taux de compression est $Cr = \frac{Taille\ d'image\ originale}{Taille\ d'image\ compressée}$.

Image	Dimensions	Taille (Ko)	*Cr*
Originale	512×512	262,144	
Réduit par 2	256×256	65,536	4
Réduit par 4	128×128	16,384	16
Réduit par 8	64×64	4,096	64

Tableau II.2 Le taux de compression pour l'image compressée ‘‘Lena’’

Une fois la compression des images est faite avec perte, nous proposons de réaliser une reconstruction par les réseaux de neurones MLP et RBF pour les différents taux de compression obtenus.

Le schéma synoptique du bloc de reconstruction se présente dans l'ordre suivant :

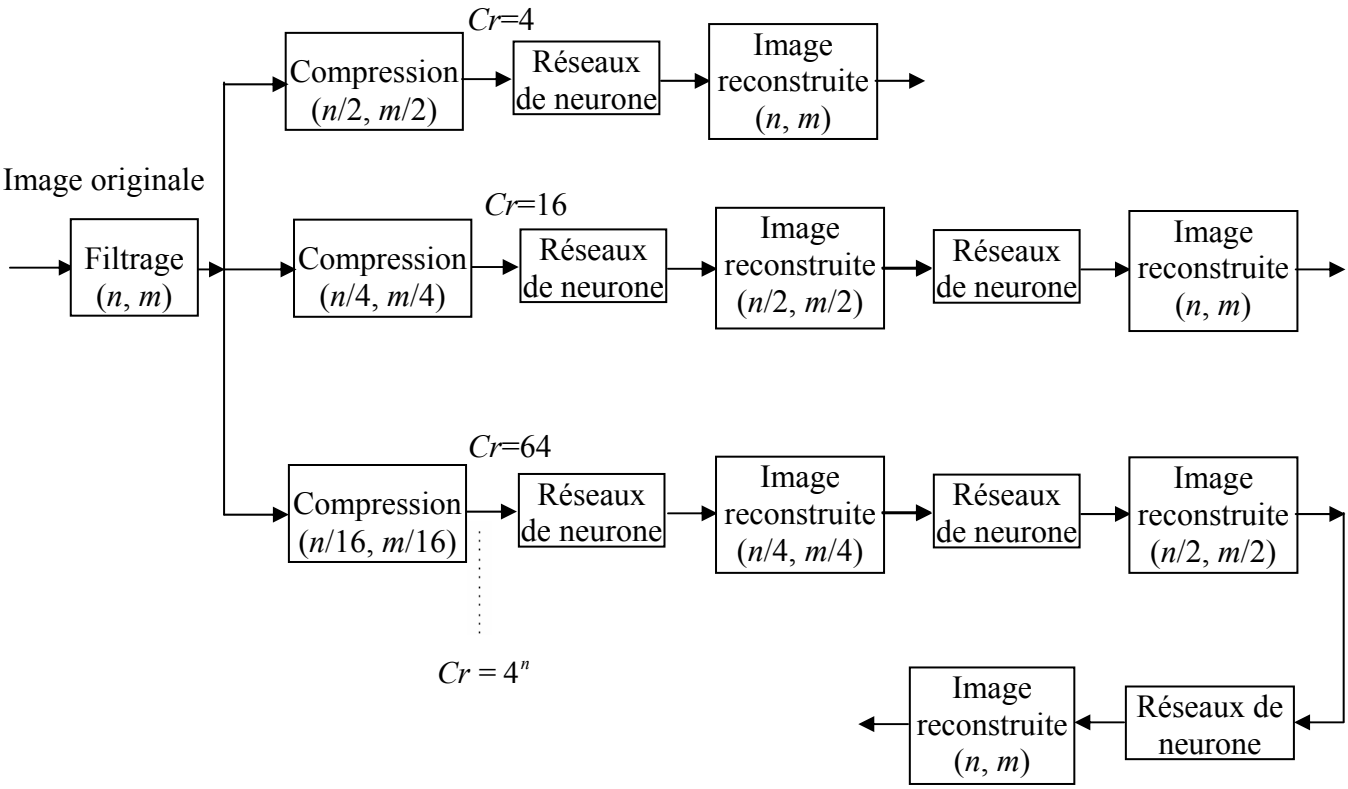


Figure II.5 Algorithme bloc de reconstruction

II.4.2. Base d'apprentissage

Le choix de la base d'apprentissage est très important dans la conception des réseaux de neurone parce qu'un mauvais choix de cette base peut entraîner beaucoup de problèmes sur les performances des réseaux de neurones, Il existe plusieurs méthodes pour créer une base de données servant d'exemples pour le traitement d'image par réseau de neurones, dans notre cas nous avons créé deux types de parcours de base de données : avec recouvrement figures II.6, II.7 et sans recouvrement figures II.8, II.9.

Pour les parcours de la base de données une image compressée de taille 104×176, image reconstituée de 208×352.

Dans la base de données avec recouvrement $N_e = (n-1)(m-1)$ et $N_e = \frac{n \times m}{4}$ dans la base de données sans recouvrement, avec N_e nombre d'exemples, (n, m) largeur et hauteur de l'image compressée.

Si on a une base avec recouvrement, N_e=18025, sinon N_e=4576.

Dans la première base nous allons utiliser comme entrée la courbe illustrée sur la figure II.6 (la taille de bloc est $4\times N_e$) et comme sortie la courbe de la figure II.7 (la taille de bloc est $16\times N_e$), et dans la deuxième base on change l'entrée par la courbe dans la figure II.8 ($4\times N_e$) et on a la sortie le parcours comme montré dans la figure II.9 ($16\times N_e$).

Plus ce nombre est grand, plus la généralisation est bonne.

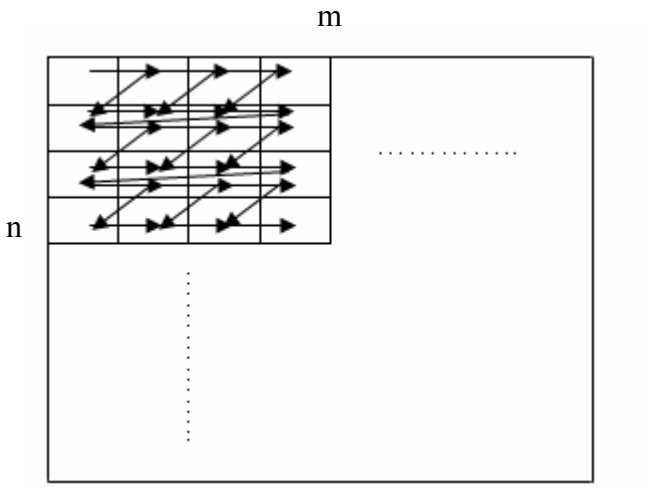


Figure II.6 Parcours de la base d'entrée de réseau de neurone (Courbe avec recouvrement)

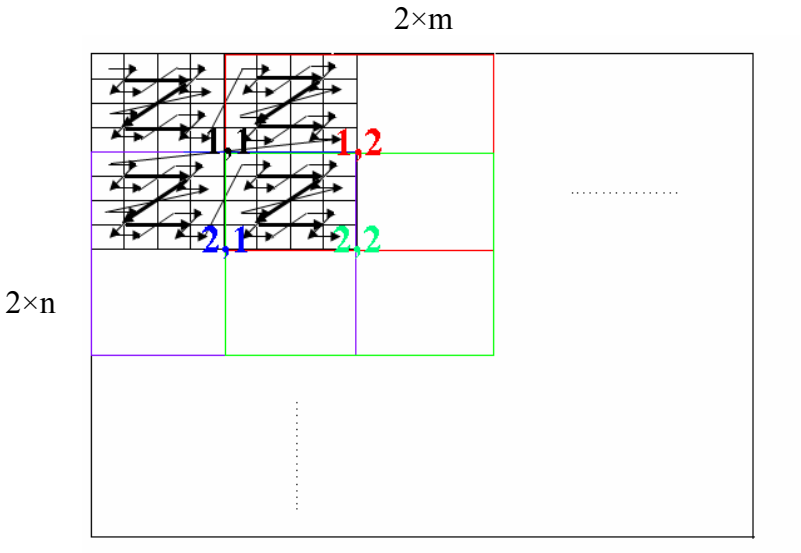


Figure II.7 Parcours de la base de sortie de réseau de neurone (Courbe avec recouvrement)

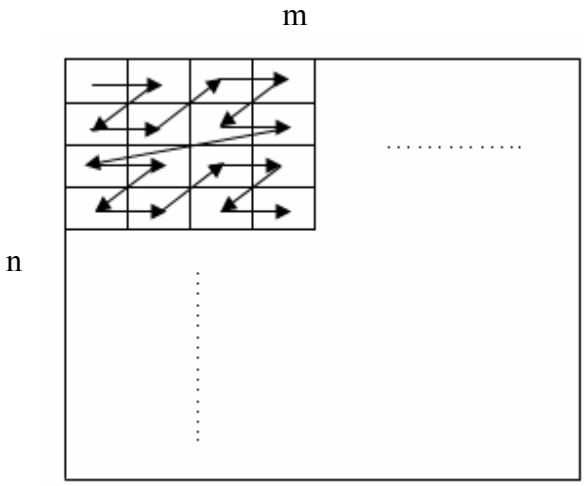


Figure II.8 Parcours de la base d'entrée de réseau de neurone (Courbe sans recouvrement)

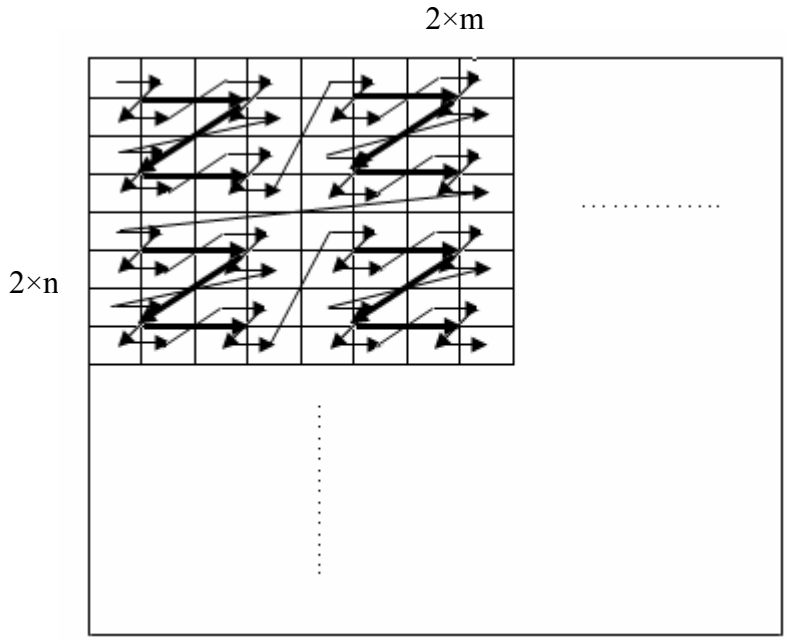


Figure II.9 Parcours de la base de sortie de réseau de neurone (Courbe sans recouvrement)

II.4.3. Architecture du réseau

L'architecture représentée sur la figure II.10 est celle d'une reconstruction d'image en utilisant le réseau de neurones pour lequel on a utilisé deux bases de données, la première avec recouvrement et la deuxième sans recouvrement ; elles sont présentées plusieurs fois au réseau de neurones suivant un parcours de façon à minimiser au maximum les différents critères d'apprentissage (*MAE*, *MSE*, *NMSE*, *PSNR*).

La base de donnée à des fenêtres sur l'image est de taille 2×2 à l'entrée et 4×4 à la sortie.

Afin d'améliorer les performances des réseaux neuronaux, il est préférable de normaliser les données d'entrée et de sortie de telle sorte qu'elles se trouvent dans l'intervalle [0, 1].

Pour le traitement d'image à 256 niveaux de gris, la fonction k est de la forme k =1/255 et la fonction k^{-1} de la forme $k^{-1} = 255$.

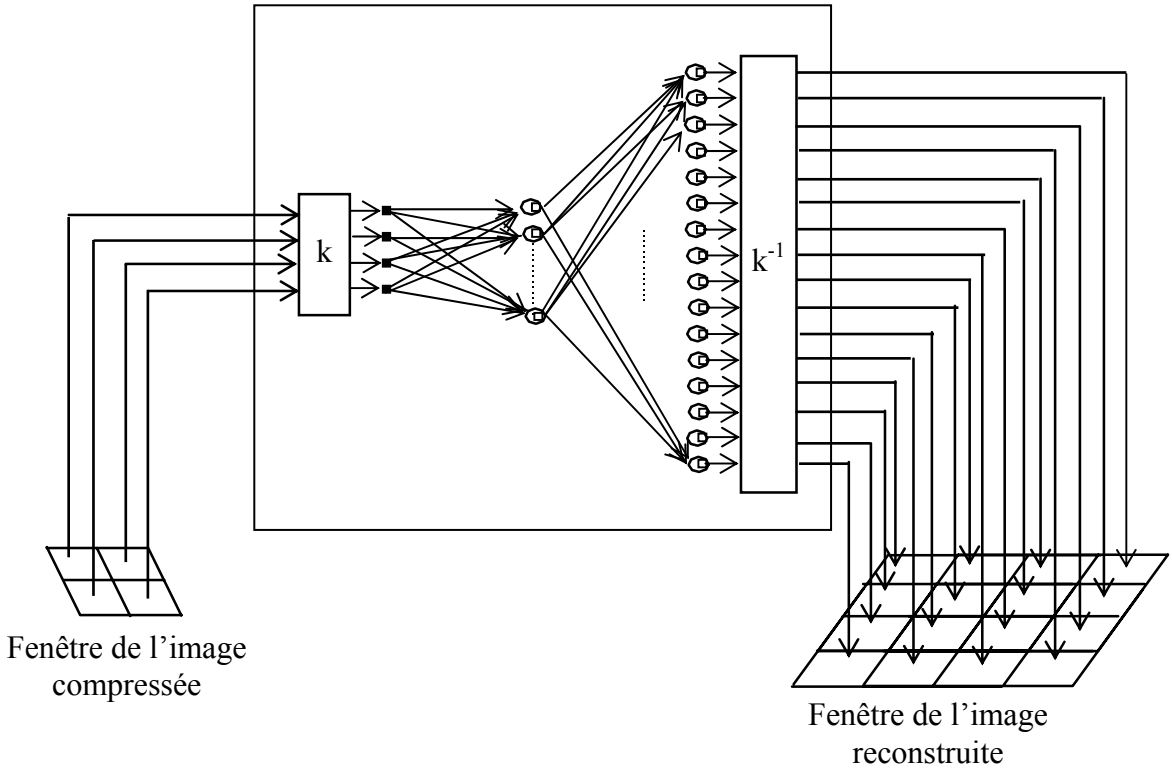


Figure II.10 Architecture du réseau

Lors de l'apprentissage et de la phase de reconstruction d'une image, la fenêtre de la compression neuronale est déplacée pixel par pixel selon le parcours choisi sur l'image. Cette architecture permet de reconstituer des images de tailles différentes.

II.4.4. Critères d'apprentissages

Pour pouvoir vérifier la validité des résultats obtenus par le réseau, il faut trouver un moyen autre que le critère visuel, qui n'est pas très objectif pour nous permettre de comparer les images reconstituées avec les images d'origine.

Pour cela nous avons opté dans notre travail d'utiliser des paramètres quantitatifs décrits dans le chapitre I, à savoir l'erreur absolue moyenne (Mean Absolute Error, *MAE*), l'erreur moyenne quadratique (Mean Square Error, *MSE*), l'erreur moyenne quadratique normalisée (Percent Normalised Mean Square Error, *NMSE*), et enfin le rapport signal sur bruit crête (Peak Signal Noise Ratio, *PSNR*).

II.4.5. Apprentissage des réseaux de neurones

Une fois l'architecture d'un réseau de neurones choisie, il est nécessaire d'effectuer un apprentissage pour déterminer les valeurs des poids permettant à la sortie du réseau de neurones d'être aussi proche que possible de l'objectif fixé.

Cet apprentissage s'effectue grâce à la minimisation d'une fonction, appelée fonction de coût, calculée à partir des exemples de la base d'apprentissage et de la sortie du réseau de neurones; cette fonction détermine l'objectif à atteindre.

II.4.6. Conception du Réseau de neurones

II.4.6.1. Configuration et optimisation des caractéristiques du réseau

Pour obtenir la meilleure configuration du réseau de neurones au niveau de la reconstruction d'image compressée, nous employons deux bases pour optimiser le réseau de neurones lors de l'étape d'apprentissage. Le choix du nombre de neurones de la couche cachée nécessaire au problème n'est pas théoriquement défini.

II.4.7. Réseaux de neurones MLP

II.4.7.1. Présentation des simulations effectuées

Nous cherchons à estimer les performances générales du réseau ainsi qu'à déterminer quelles sont ses faiblesses au niveau pratique. Les tests suivants couvrent la plupart des informations que l'on peut retirer directement du réseau :

1. Comment déterminer les valeurs ω, μ, m, les nombres des neurones dans les trois couches (entrée, cachée, sortie) et le nombre d'itérations optimales.
2. Calculer l'erreur absolue moyenne *MAE*.
3. Calculer l'erreur quadratique moyenne *MSE*.
4. Calculer l'erreur moyenne quadratique normalisée *NMSE*.
5. Calculer le rapport signal sur bruit crête *PSNR*.

Afin de reconstituer une image compressée nous avons procédé selon la démarche suivante :

a. la première étape consiste à déterminer l'architecture optimale du réseau de neurones. Les valeurs de ces paramètres dépendent généralement des données du problème et sont difficiles à estimer à priori. On ne fait varier qu'une des constantes du réseau à la fois, par exemple le nombre de la couche cachée ou le nombre d'itérations pour l'apprentissage, et le taux d'apprentissage, le momentum. Ensuite on recommence l'apprentissage du début avec une nouvelle valeur pour la constante. Ainsi on obtient assez de statistiques pour estimer quelles sont les valeurs optimales des constantes d'apprentissage. Ces tests utilisent les deux bases de données, nous avons entraîné un MLP avec un nombre variable de neurones (2 à 12) dans la couche cachée à l'aide de la base d'apprentissage extraite des images compressée et originales. Les entrées de la base ont été extraites des images compressées, alors que les pixels correspondants sur les images originales ont constitué les sorties désirées.
La méthode d'apprentissage utilisée est la rétropropagation du gradient, cet algorithme itératif est le plus utilisé pour l'apprentissage du perceptron multicouche.

b. Dans la deuxième étape, une fois l'apprentissage terminé, c'est-à-dire la meilleure architecture du réseau obtenu, nous avons appliqué des images de test (celles-ci ont été bien sur filtrées et compressées avant avec différents taux de compression 4, 16, 64) au réseau MLP.

II.4.7.2. Algorithme d'apprentissage

Le choix d'un critère d'erreur (*MAE*, *MSE*, *NMSE*, *PSNR*) pour l'apprentissage du réseau de neurones se fait comme suit :

Soit *E* la fonction coût que l'on veut obtenir (*MAE* ou *MSE* ou *NMSE* ou *PSNR*) :

Tant que l'erreur > *E*, pour chaque base de donnée il faut faire :

- Présenter un exemple en entrée, déterminer les paramètres de réseau de neurones (taux d'apprentissage, momentum).
- Propager les signaux d'activation des entrées vers la sortie
  - Calculer la sortie
- Comparer la sortie calculée avec la sortie désirée
  - Calculer l'erreur (*MAE*, *MSE*, *NMSE*, *PSNR*).
- Rétropropager l'erreur des cellules de sortie vers les cellules d'entrée

- Modifier les poids des connections de chaque couche (mise à jour des poids).

Nous verrons dans la suite, la mise en œuvre détaillée de l'algorithme ainsi que les résultats pratiques obtenus sur des images de ''Lena'' au niveau de gris et couleur.

II.4.7.3. Analyse des résultats

Dans cette section, nous présentons les résultats obtenus sur l'image compressée après l'application du réseau de neurone MLP à la base de donnée. Le nombre de neurone de la couche d'entrée est ainsi fixé à 4 neurones. La couche de sortie est de 16 neurones.

Les courbes données sur les deux figures II.11 et II.12 illustrent comment le nombre de neurones et la base de données (avec et sans recouvrement) agissent sur les différentes fonctions de coût (*MAE*, *MSE*, *NMSE*, *PSNR*).

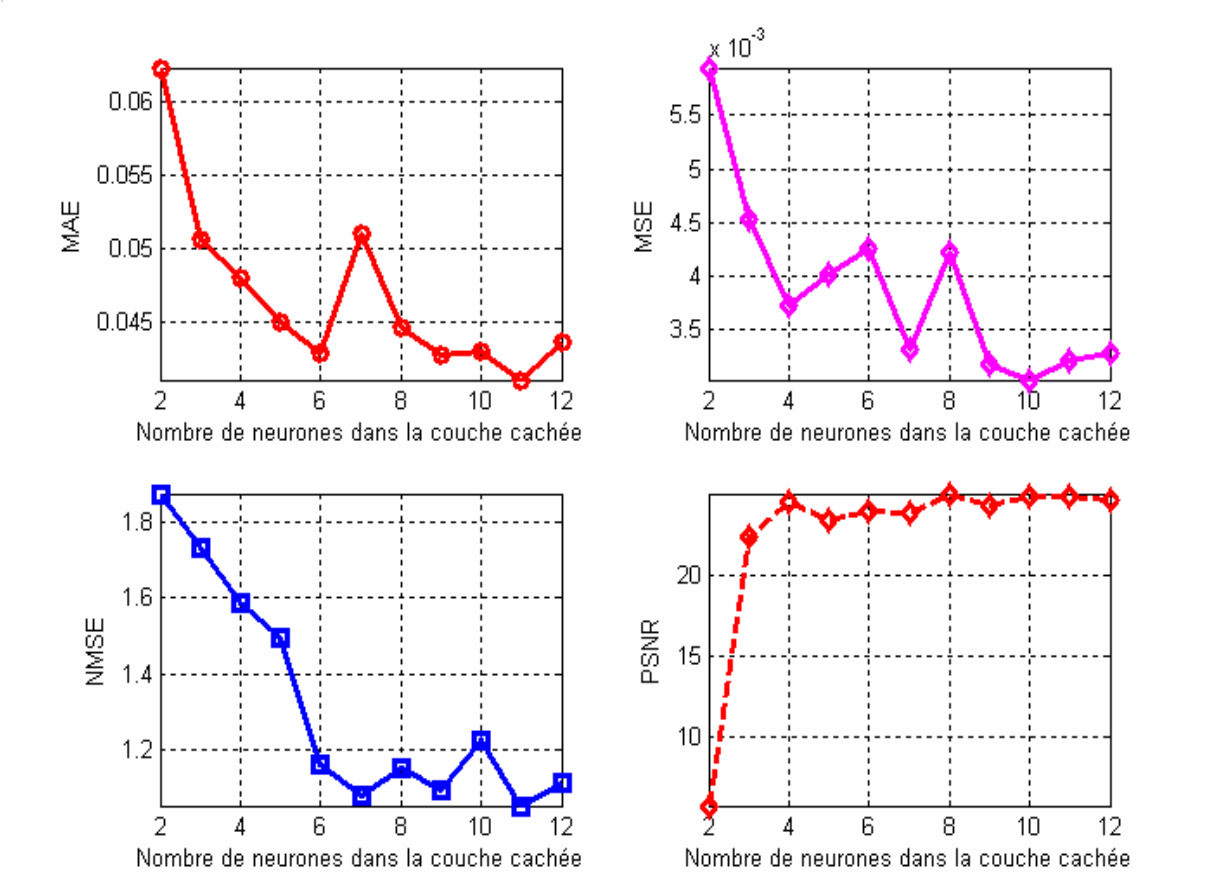


Figure II.11 Evaluation du critère d'erreur (*MAE*, *MSE*, *NMSE*, *PSNR*) en fonction du nombre de neurones dans la couche cachée avec la base de données avec recouvrement

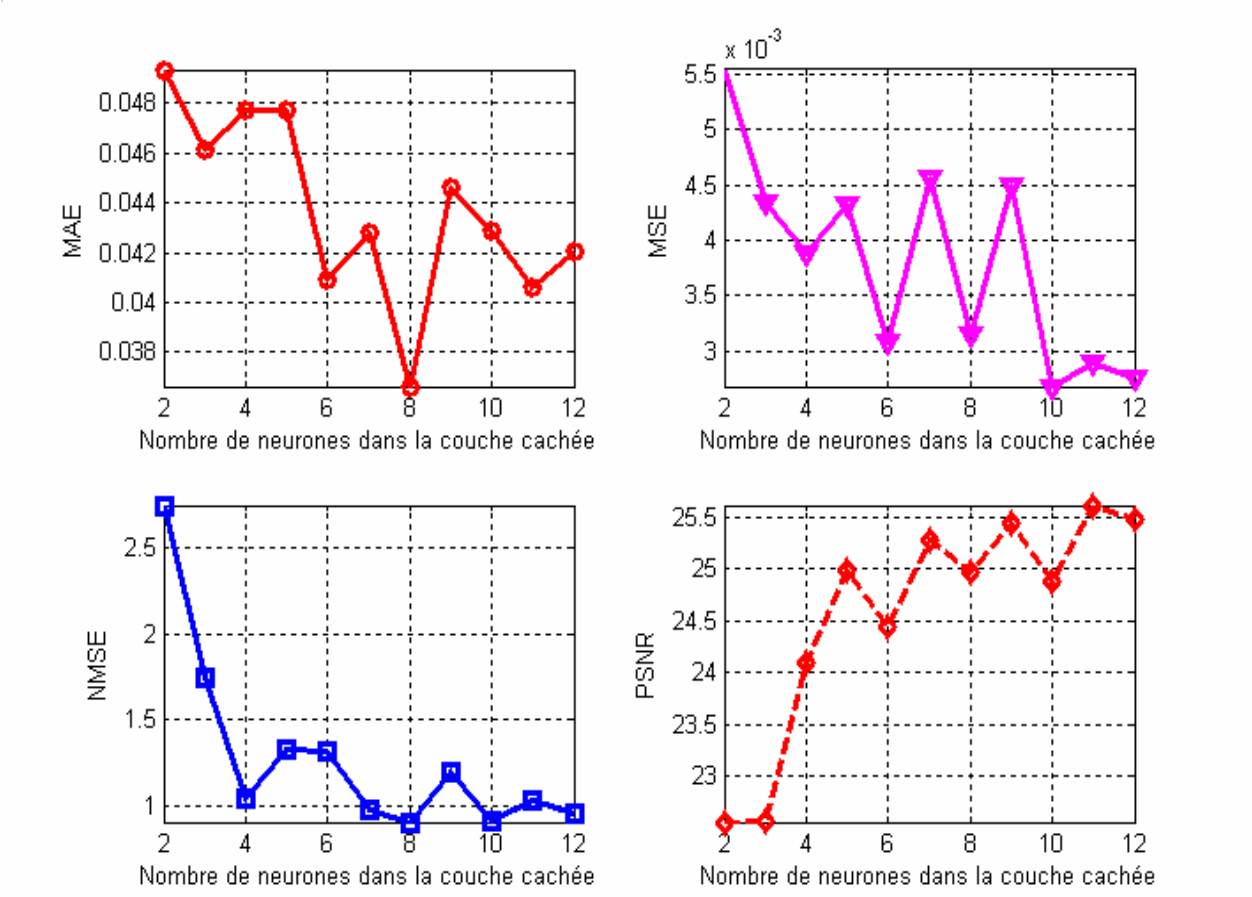


Figure II.12 Evaluation du critère d'erreur (*MAE*, *MSE*, *NMSE*, *PSNR*) en fonction du nombre de neurones dans la couche cachée avec la base de données sans recouvrement

II.4.7.4. Validation des résultats

Pour la validation de la robustesse et la performance du réseau de neurones développé, on passe à l'étape de l'application sur l'image de ''Lena'' son vent utilisé dans la littérature (voir figure II.13 à II.16). Nous avons utilisé les deux parcours des données avec et sans recouvrement qui se trouvent aux figures II.6 à II.9.

Une image originale ''Lena'' au niveau de gris et couleur, compressée par les différents facteurs de réduction (Taux de compression 4,16 et 64), reconstruite par l'application des réseaux de neurones MLP.

Comme on peut le voir, les blocs sont nettement visibles sur l'image reconstruite. L'effet de blocs est par contre diminué par le filtre médian. De plus, la qualité objective de l'image est améliorée, comme signalé dans la légende de la figure II.13 et la figure II.14.

Figure II.13 Reconstruction d'image ''Lena'' compressée avec recouvrement

Images compressées (a), (b) et (c)

Images reconstruites avant filtrage (d), (e) et (f)

Images reconstruites après filtrage (g), (h) et (i)

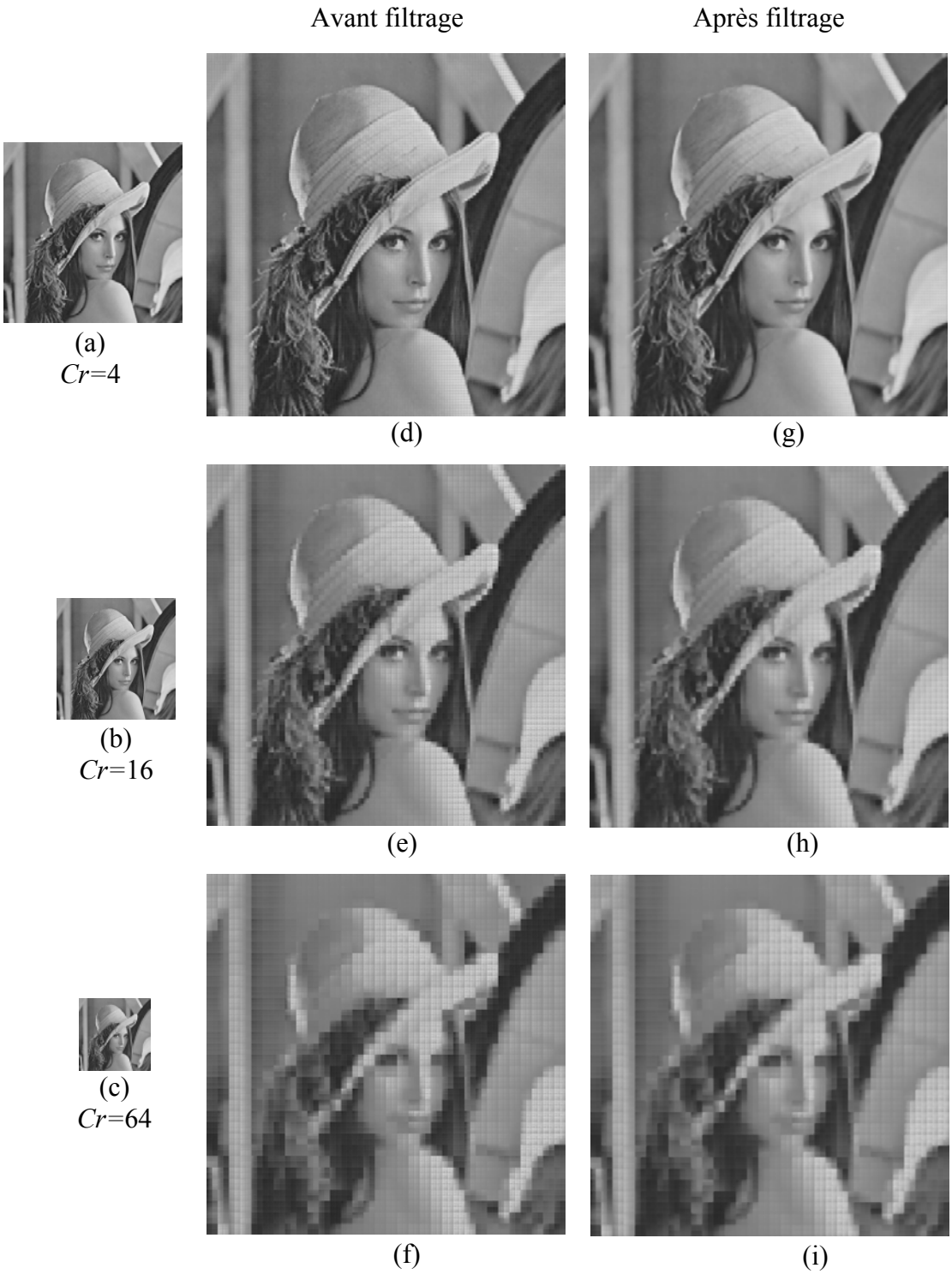


Figure II.14 Reconstruction d'image ''Lena'' compressée sans recouvrement

Images compressées (a), (b) et (c)

Images reconstruites avant filtrage (d), (e) et (f)

Images reconstruites après filtrage (g), (h) et (i)

L'augmentation moyenne du PSNR sur l'image de test lors de l'utilisation de différents taux de compression est donnée dans le tableau II.3 et II.4. L'algorithme réalise une très forte augmentation de *PSNR* surtout à des faibles taux de compression. Par exemple, la différence est de plus de 1.6db lorsque le taux de compression est 4, alors que pour un taux de compression de 16 la différence est approximativement de 0.7db.

	***Cr*=4**		***Cr*=16**		***Cr*=64**	
Mesure	Avant filtrage	Après filtrage	Avant filtrage	Après filtrage	Avant filtrage	Après filtrage
MAE	4.1831	3.4956	7.7944	7.1207	12.3647	11.8587
MSE	50.8380	35.4056	156.3361	135.0772	355.0480	337.0999
NMSE	0.2898	0.2018	0.8911	0.7700	2.0238	1.9215
PSNR	31.0689	32.6401	26.1902	26.8250	22.6279	22.8532

Tableau II.3 Mesure de l'erreur entre l'image originale et celle réduite puis reconstruite par un parcours avec recouvrement pour différents facteurs de compression

Le tableau II.4 donne les différentes mesures d'erreurs entre l'image originale et l'image reconstruite par un parcours sans recouvrement.

	***Cr*=4**		***Cr*=16**		***Cr*=64**	
Mesure	Avant filtrage	Après filtrage	Avant filtrage	Après filtrage	Avant filtrage	Après filtrage
MAE	4.9151	3.7405	8.6331	7.6682	13.1877	12.5421
MSE	64.4858	38.9700	182.6214	149.5189	392.6655	364.2973
NMSE	0.3676	0.2221	1.0410	0.8523	2.2382	2.0765
PSNR	30.0362	32.2235	25.5153	26.3838	22.1906	22.5162

Tableau II.4 Mesure de l'erreur entre l'image originale et celles réduite puis reconstruite par un parcours sans recouvrement pour différents facteurs de compression

Comme pour la première application sur l'image ''Lena'' au niveau de gris, les figure II.15 et II.16 montre la seconde application du réseau de neurones MLP sur l'image ''Lena'' en couleur (RGB).

Figure II.15 Reconstruction de l'image ''Lena'' compressée avec recouvrement

Images compressées (a), (b) et (c)

Images reconstruites avant filtrage (d), (e) et (f)

Images reconstruites après filtrage (g), (h) et (i)

Figure II.16 Reconstruction de l'image ''Lena'' compressée sans recouvrement

Images compressées (a), (b) et (c)

Images reconstruites avant filtrage (d), (e) et (f)

Images reconstruites après filtrage (g), (h) et (i)

Le tableau II.5 donne les différentes métriques pour mesurer la distorsion entre l'image originale et l'image reconstruite par un parcours avec recouvrement.

	***Cr*=4**		***Cr*=16**		***Cr*=64**	
Mesure	Avant filtrage	Après filtrage	Avant filtrage	Après filtrage	Avant filtrage	Après filtrage
MAE	5.0314	4.0454	8.8022	7.6116	13.3514	12.3969
MSE	62.1777	39.5744	181.0149	138.3837	391.8582	342.6685
NMSE	0.3964	0.2523	1.1541	0.8823	2.4984	2.1848
PSNR	30.1945	32.1567	25.5537	26.7200	22.1995	22.7821

Tableau II.5 Mesure de l'erreur entre l'image originale et celle réduite puis reconstruite par un parcours avec recouvrement pour différents facteurs de compression

Le tableau II.6 donne les différentes mesures d'erreurs entre l'image originale et l'image reconstruite par un parcours sans recouvrement après les différents taux de compression.

	***Cr*=4**		***Cr*=16**		***Cr*=64**	
Mesure	Avant filtrage	Après filtrage	Avant filtrage	Après filtrage	Avant filtrage	Après filtrage
MAE	6.1586	4.4885	9.9791	8.4084	14.4744	13.3393
MSE	86.1103	48.0101	222.0663	165.0645	446.0515	390.4374
NMSE	0.4679	0.2921	1.2063	0.9713	2.4482	2.2268
PSNR	28.7803	31.3175	24.6660	25.9543	21.6370	22.2153

Tableau II.6 Mesure de l'erreur entre l'image originale et celle réduite puis reconstruite par un parcours sans recouvrement pour différents facteurs de compression

Les figures II.13 à II.16 illustre bien la reconstruction d'image ''Lena'' (au niveau de gris et couleur) compressée pour des facteurs de compression de l'ordre 4,16 et 64. Nous pouvons conclure que pour un taux de compression *Cr*=64, l'image décompressée est fortement dégradée pour les deux images de ''Lena'' au niveau de gris et couleur.

Nous pouvons conclure aussi que l'utilisation des réseaux de neurone MLP sur l'image de ''Lena'' au niveau de gris et couleurs compressée par décimation donne de très bons résultats, mais avec un temps d'exécution relativement élevé surtout dans le cas où la base de données est avec recouvrement.

II.4.8. Réseau de neurones RBF

L'avantage principal des RBF réside dans l'apprentissage, par ailleurs, pour autant que le nombre de données soit suffisant, les résultats obtenus avec ce modèle sont aussi meilleurs que ceux obtenus avec un MLP (perceptron multicouches).

Les réseaux RBF (Radial Basis Function) sont des réseaux à couches dont l'origine remonte à une technique d'interpolation RBF, le réseau est constitué d'une couche cachée dont les fonctions d'activations ont l'allure d'une courbe gaussienne. La couche de sortie est la plupart du temps constituée de fonctions d'activations linéaires [30].

Dans un réseau de neurones RBF, $f(x)$ est approximée par un ensemble des fonctions d'activations d-dimensionnelles radiale. Ces fonctions de base radiale sont portées sur les points de repères bien placés, appelés les centres des Gaussiennes.

Les centres des Gaussiennes peuvent être considérés comme les nœuds de la couche cachée. Généralement la position des centres des Gaussiennes et la largeur des fonctions de base radiale sont obtenues par un apprentissage non supervisé, tandis que les poids de la couche cachée sont calculés par apprentissage supervisé [43].

Supposons que nous voulons approximer la fonction $f(x)$ avec un ensemble M de fonction radiale de base $\phi_j(x)$, porté sur les centres Gaussiennes C_j donnée par :

$$\phi_j : R^d \rightarrow R : \phi_j(x) = \phi_j(\|x - C_j\|) \tag{II.32}$$

avec $C_j \in R^d$ et $1 \leq j \leq M$

L'approximation de la fonction $f(x)$ peut être exprimée par :

$$\hat{f}(x) = \sum_{j=1}^{M} \lambda_j \phi_j(\|x - C_j\|) \tag{II.33}$$

avec :

$$\varphi_j(\|x - C_j\|) = \exp\left[-\frac{1}{2}\left(\frac{\|x - C_j\|}{\beta_j}\right)^2\right] \tag{II.34}$$

Les Gaussiennes sont définies par les trois paramètres C_j, β_j et λ_j.

II.4.8.1. Algorithme d'apprentissage

L'apprentissage est une phase du développement d'un réseau de neurone durant laquelle le comportement du réseau est modifié jusqu'à l'obtention du comportement désiré.
Notre objectif est l'amélioration des performances du réseau, l'apprentissage consiste donc à ajuster ces poids de manière à satisfaire un critère d'optimisation.
Souvent, l'algorithme d'apprentissage peut être décrit comme suit [43] :

1. Déterminer les centres C_j des Gaussiens,
2. Calculer les largeurs β_j des Gaussiens de *''kernels''*,
3. Calculer les poids λ_j.

Pendant les deux premières étapes seulement les entrées de l'ensemble de données de l'apprentissage sont employées, c'est pour cette raison qu'on l'appelle apprentissage non supervisé.
Les centres des Gaussiennes sont estimés selon un arrangement de quantification vectoriel, comme par exemple le compétitive learning, la mise à jour des centres Gaussiennes est comme suit [43] :

$$C_i(t+1) = C_i(t) + \alpha(t)\left\|Y_i - C_i\right\| \tag{II.35}$$

où α est un facteur décroissant d'adaptation en fonction du temps, avec $0 < \alpha(t) < 1$.

La largeur des Gaussiennes est définie par la formule suivante :

$$\beta_i = \frac{1}{N_{CL_i}} \sum_{Y \in clusteri} (Y_i - C_i)^t (Y_i - C_i) \tag{II.36}$$

La mise à jour des largeurs des Gaussiennes est donnée par la règle suivante :

$$\forall j, \beta_j = q\beta_j^c \tag{II.37}$$

Avec q est un constant à choisir.

Les paramètres sont ainsi adaptés selon une règle non supervisée.

Dans la littérature, il existe plusieurs algorithmes pour le calcul des centres des Gaussiennes et les poids [33], [38], [44].

Une fois les centres et les largeurs sont calculés, le calcul des poids λ_j se fait par un apprentissage supervisé.

La configuration de base de ce réseau sera constituée d'une couche d'entrée composée d'autant de neurones qu'il y a de variable pour les vecteurs d'entrée, d'une couche cachée et d'une couche de sortie évoluera entre 0 et +1. Dans le cas de notre travail, nous choisirons un réseau RBF à une couche cachée de 2 jusqu'à 12 neurones, 4 neurones présents à l'entrée et 16 à la sortie.

Pour minimiser la fonction de coût on fait l'apprentissage de réseaux de neurones RBF sur les différents critères d'erreurs (*MAE*, *MSE*, *NMSE*, *PSNR*).

Les figures II.17 et II.18 présentent l'évolution des performances des réseaux de neurones RBF en fonction du nombre de neurones dans la couche cachée pour la première et la deuxième base de données avec et sans recouvrement.

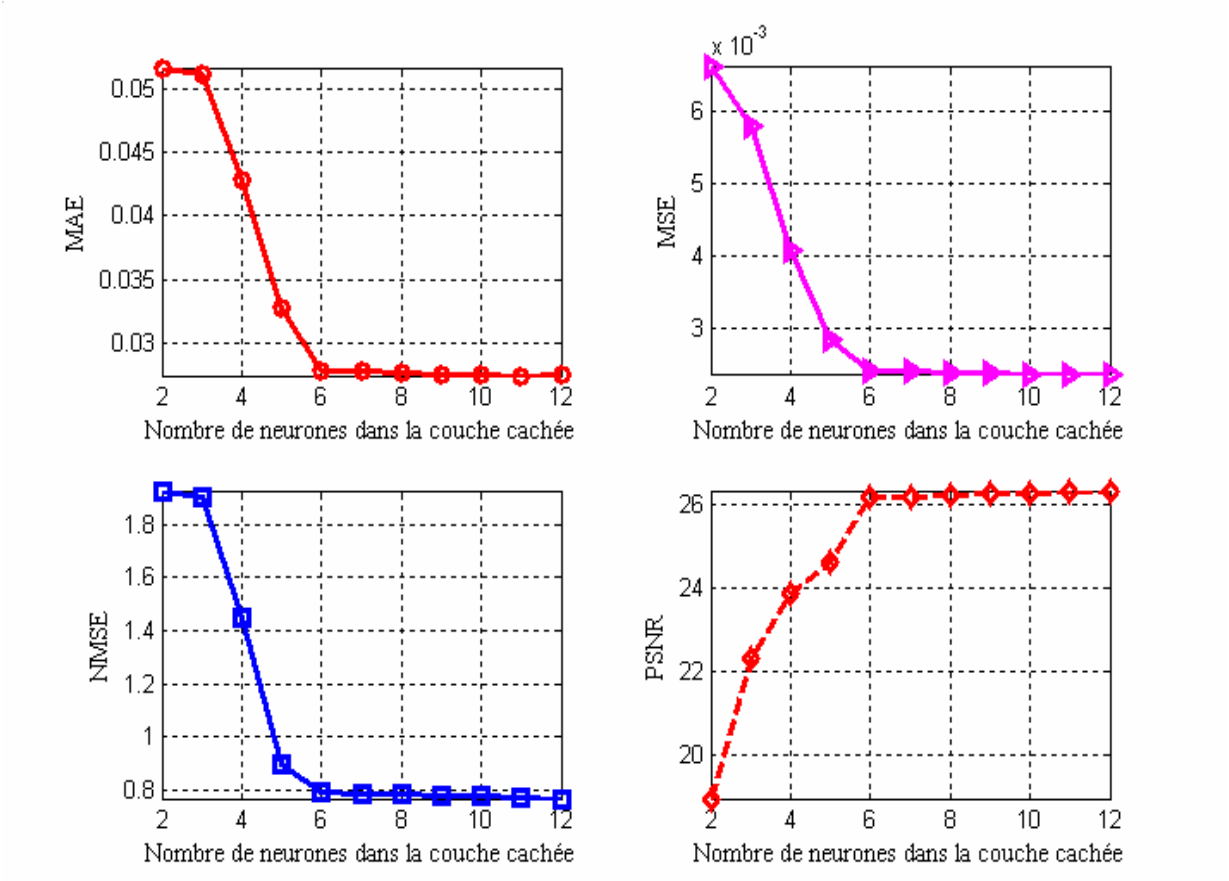


Figure II.17 Evaluation du critère d'erreur (*MAE*, *MSE*, *NMSE*, *PSNR*) en fonction de nombre de neurones dans la couche cachée (base de donnée avec recouvrement)

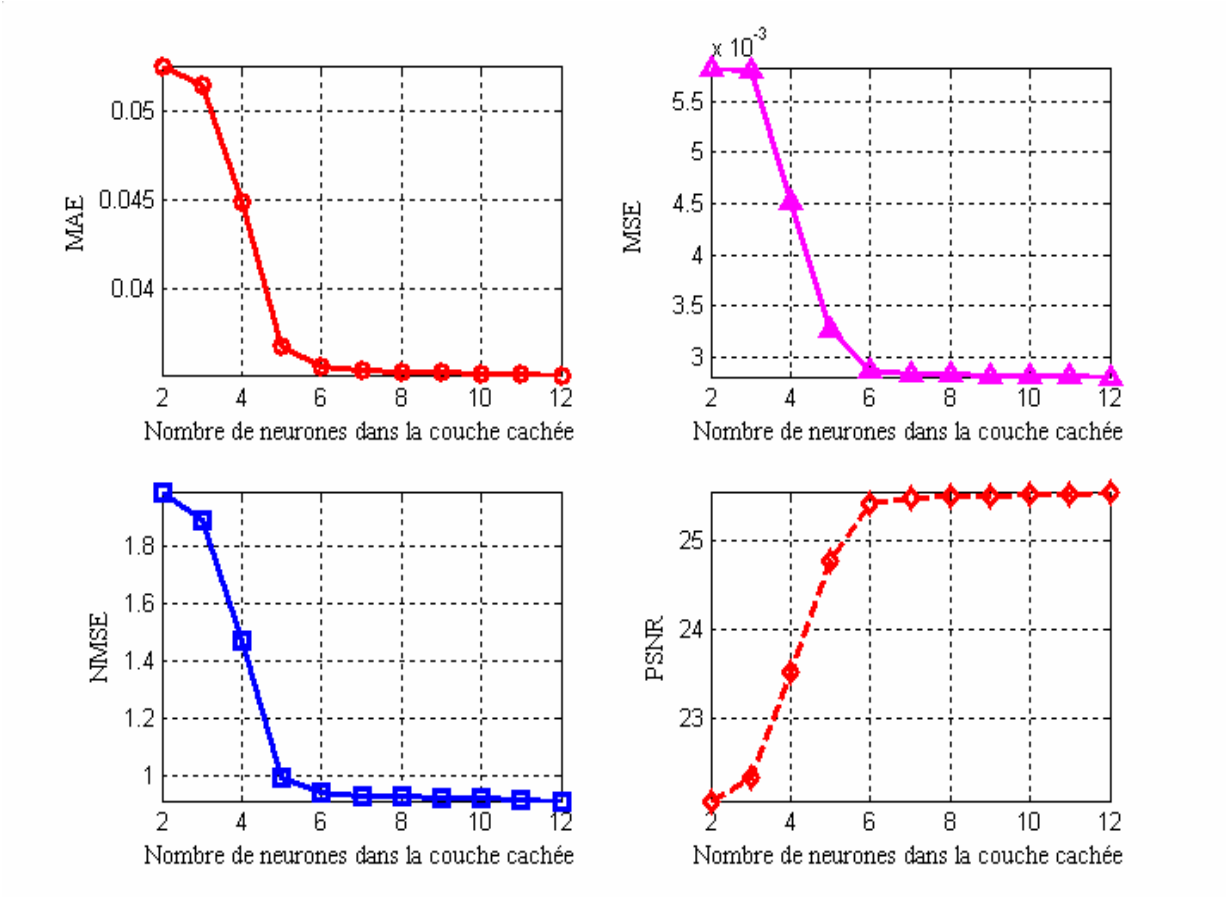


Figure II.18 Evaluation du critère d'erreur (*MAE*, *MSE*, *NMSE*, *PSNR*) en fonction de nombre de neurones dans la couche cachée de la base sans recouvrement

II.4.8.2. Validation du modèle

En plus de l'ensemble d'apprentissage, un second ensemble appelé ensemble de test, est utilisé à la fin de chaque phase d'apprentissage. On mesure non seulement l'erreur d'apprentissage mais aussi l'erreur de test, c'est-à-dire l'erreur totale commise sur tous les exemples de l'ensemble de test. Cette erreur de test est calculée une fois que la phase d'optimisation des poids est terminée.

Nous avons mené une étude empirique qui consiste à répéter plusieurs expériences avec un réseau RBF utilisant à chaque fois, l'une des bases de donnes avec et sans recouvrement.

Le réseau RBF converge rapidement avec un nombre d'itération ne dépassant pas 100 et une erreur maximale de l'ordre 10^{-3}.

Il faut noter que la précision des estimations d'un réseau RBF ne dépend pas uniquement du nombre de neurones de la couche cachée, mais aussi des largeurs des Gaussiennes utilisées par la fonction d'activation des neurones de la couche cachée.

L'analyse des résultats dans les figures II.17 et II.18 montre que les résultats ne sont pas forcément nettement améliorés si on augmente le nombre de neurones de la couche cachée, c'est-à-dire les courbes d'erreur se stabilise à une valeur fixe après le sixième neurone.

Le tableau II.7 représente les mesures des performances de réseau de neurones RBF.

Mesure	Performance avec recouvrement	Performance sans recouvrement
MAE	0.0274	0.0351
MSE	0.0023	0.0028
NMSE	0.7702	0.9101
PSNR	26.3827	25.5284

Tableaux II.7 Des mesures des performances de réseaux de neurones

Les images reconstituées sont montrées dans les figures II.19 à II.22 respectivement pour les deux bases de données sur l'image ‘‘Lena’’ au niveau de gris et couleur.

Figure II.19 Reconstruction d'image ''Lena'' compressée avec recouvrement

Images compressées (a), (b) et (c)

Images reconstruites avant filtrage (d), (e) et (f)

Images reconstruites après filtrage (g), (h) et (i)

Figure II.20 Reconstruction d'image ''Lena'' compressée sans recouvrement

Images compressées (a), (b) et (c)

Images reconstruites avant filtrage (d), (e) et (f)

Images reconstruites après filtrage (g), (h) et (i)

Figure II.21 Reconstruction d'image ''Lena'' compressée avec recouvrement

Images compressées (a), (b) et (c)

Images reconstruites avant filtrage (d), (e) et (f)

Images reconstruites après filtrage (g), (h) et (i)

Figure II.22 Reconstruction d'image ‘‘Lena’’ compressée sans recouvrement

Images compressées (a), (b) et (c)

Images reconstruites avant filtrage (d), (e) et (f)

Images reconstruites après filtrage (g), (h) et (i)

Les tableaux II.8 à II.11 illustrent également l'évaluation des performances des mesures sur l'image de test ‘‘Lena’’ obtenus pour chaque expérience.

	***Cr*=4**		***Cr*=16**		***Cr*=64**	
Mesure	Avant filtrage	Après filtrage	Avant filtrage	Après filtrage	Avant filtrage	Après filtrage
MAE	3.0404	3.1180	6.5685	6.4517	10.9895	10.858
MSE	35.6461	29.4265	117.2718	113.9921	289.7547	288.1041
NMSE	0.2032	0.1677	0.6685	0.6498	1.6516	1.6422
PSNR	32.6107	33.4434	27.4389	27.5621	23.5105	23.5353

Tableau II.8 Mesure des performances sur l'image ‘‘Lena’’ au niveau de gris avec un parcours avec recouvrement

	***Cr*=4**		***Cr*=16**		***Cr*=64**	
Mesure	Avant filtrage	Après filtrage	Avant filtrage	Après filtrage	Avant filtrage	Après filtrage
MAE	3.1204	3.1110	6.8728	6.7228	11.3399	11.1859
MSE	36.1906	29.5450	124.0389	118.9380	299.5912	295.7651
NMSE	0.2063	0.1684	0.7070	0.6780	1.7077	1.6859
PSNR	32.5448	33.4260	27.1952	27.3776	23.3655	23.4213

Tableau II.9 Mesure des performances sur l'image ‘‘Lena’’ au niveau de gris avec un parcours sans recouvrement

	***Cr*=4**		***Cr*=16**		***Cr*=64**	
Mesure	Avant filtrage	Après filtrage	Avant filtrage	Après filtrage	Avant filtrage	Après filtrage
MAE	3.1157	3.1994	6.5713	6.3834	10.6857	10.5076
MSE	36.4960	29.4641	114.1384	108.6239	273.9635	269.0230
NMSE	0.2407	0.1997	0.7389	0.7102	1.7424	1.7195
PSNR	32.5083	33.4379	27.5565	27.7716	23.7539	23.8329

Tableau II.10 Mesure des performances sur l'image ‘‘Lena’’ en couleur avec un parcours avec recouvrement

Mesure	***Cr*=4**		***Cr*=16**		***Cr*=64**	
	Avant filtrage	Après filtrage	Avant filtrage	Après filtrage	Avant filtrage	Après filtrage
MAE	3.3357	3.3031	6.7744	6.5566	10.9191	10.7278
MSE	38.7307	31.1669	119.9922	112.7070	282.4558	275.8563
NMSE	0.2539	0.2113	0.7752	0.7375	1.7956	1.7648
PSNR	32.2502	33.1939	27.3393	27.6113	23.6213	23.7240

Tableau II.11 Mesure des performances sur l'image ''Lena'' en couleur avec un parcours sans recouvrement

Nous avons présenté dans cette partie une méthode basée sur les réseaux de neurones pour résoudre le problème de la reconstruction des images compressées. Notre choix s'est porté sur les réseaux de neurones MLP et RBF.

Les réseaux de neurones ont été testés sur des images compressées pour plusieurs taux de compression et ainsi pour deux parcours d'image (avec et sans recouvrement).

Les résultats obtenus par l'utilisation de parcours d'image sans recouvrement montrent bien que le temps de traitement devient nettement inférieur à celui de parcours avec recouvrement bien que les résultats de reconstruction restent similaires.

Les résultats obtenues sur l'image ''Lena'' au niveau de gris et couleur avec le réseau de neurones RBF sont meilleurs par rapport à ceux obtenus avec le réseau de neurone MLP (le gain de *PSNR* de l'ordre de 2.5db au niveaux de gris et 3.4db en couleur) en plus, le temps d'apprentissage est nettement plus faible par rapport à celui du réseau MLP.

Donc suite aux remarques décrites ci-dessus, nous avons utilisé les réseaux RBF pour la reconstruction des images médicales compressées sans recouvrement.

II.5. Compression d'image médicale

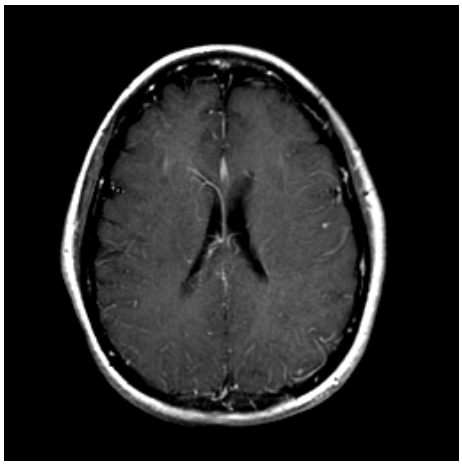
Les hôpitaux stockent chaque jour un grand nombre d'images ; les services de sécurité gèrent des bases de données contenant les empreintes de beaucoup de personnes ; et surtout l'Internet a ouvert l'accès à d'innombrables images pour le grand public. Le besoin de comprimer les images est dû à leur nombre croissant et au volume de stockage qu'elles occupent.

Le but principal de compression des images est de réduire la quantité de bits nécessaire pour décrire tout en gardant un aspect visuel acceptable, des images reconstruites. Pour résoudre ce

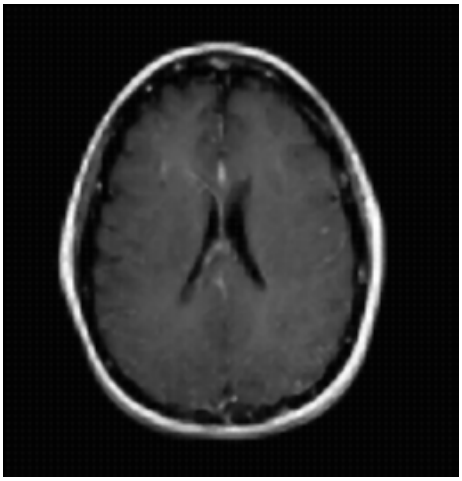
problème nous avons appliqué les réseaux à fonction radiale de base (RBF) sur les images médicales compressées pour les restituer.

Pour cela, nous avons validé la méthode en utilisant des images médicales correspondant à diverses modalités (scanner, IRM, rayon X) , les résultats obtenus montrent que la méthode est extensible à n'importe quel type d'image.

Les résultats obtenus pour les images médicales sont présentés sur les figures II.23 à II.27.



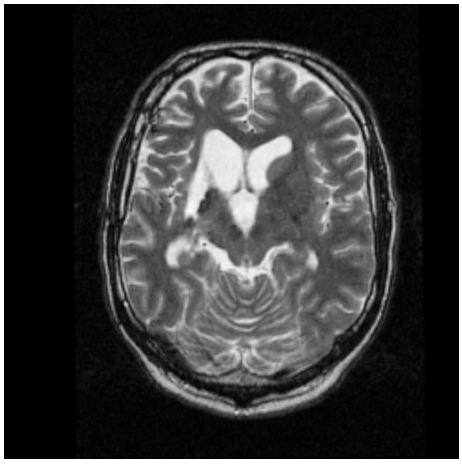
(a)
Image originale
IRM1 (256×256)



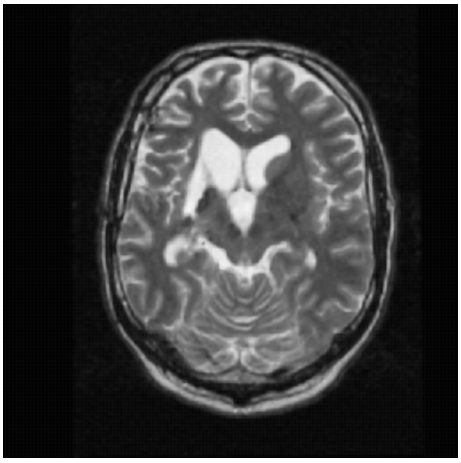
(b)
Cr=4

(c)
Cr=16

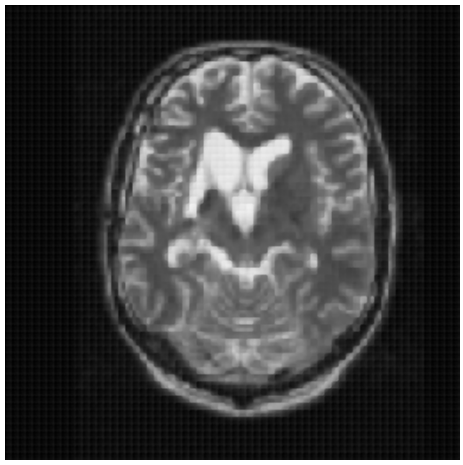
Figure II.23 Reconstruction de l'image compressée
Images reconstruites (b) et (c)



(a)
Image originale
IRM2 (512×512)



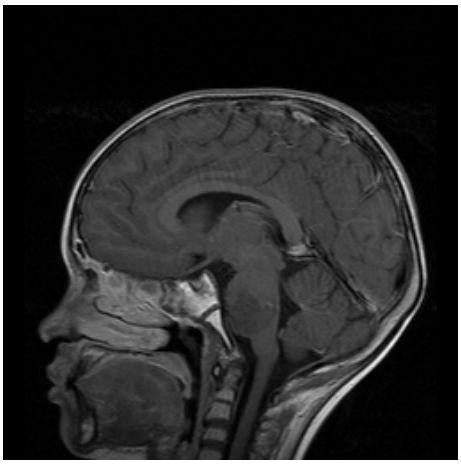
(b)
Cr=4



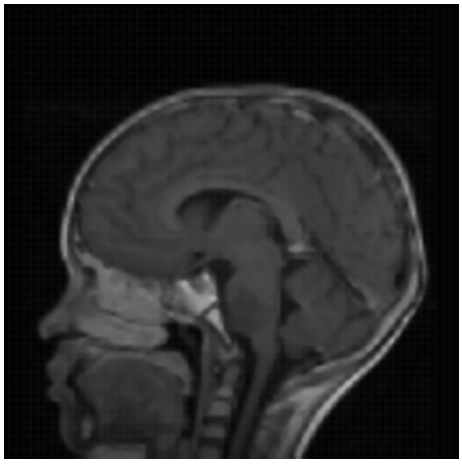
(c)
Cr=64

Figure II.24 Reconstruction de l'image compressée

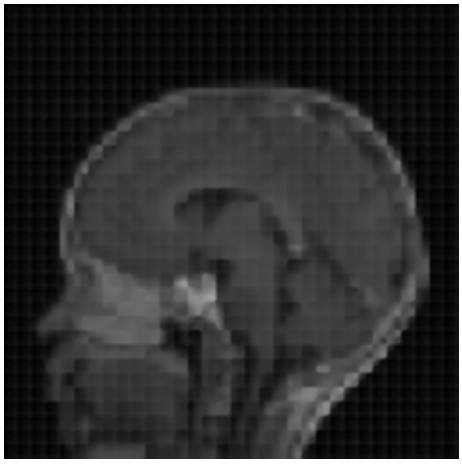
Images reconstruites (b) et (c)



(a)
Image originale
Cerveau (au niveau de gris)
(256×256)

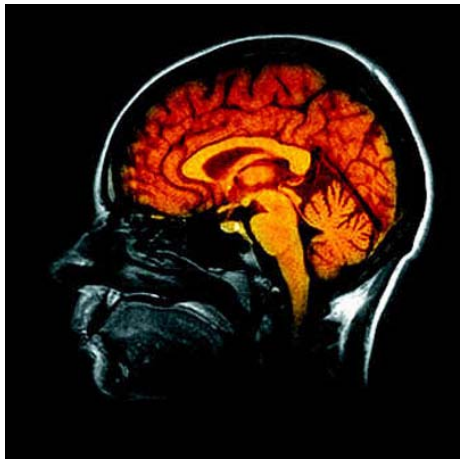


(b)
Cr=4

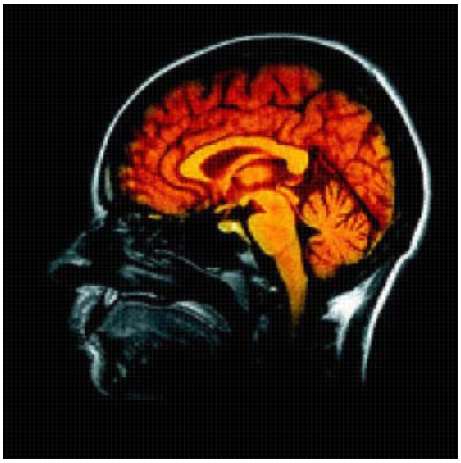


(c)
Cr=16

Figure II.25 Reconstruction d'image compressée
Images reconstruites (b) et (c)



(a)
Image originale
Cerveau (RGB)
(400×400)



(b)
Cr=4

(c)
Cr=16

Figure II.26 Reconstruction d'image compressée

Images reconstruites (b) et (c)

(a)
Image originale
rayon X
(864×544)

(b)
Cr=4

(c)
Cr=16

Figure II.27 Reconstruction de l'image compressée

Images reconstruites (b) et (c)

Le tableau II.12 illustre bien les résultats obtenus avec les différentes images médicales pour les différents critères d'erreurs.

Image	***Cr***	***MAE***	***MSE***	***NMSE***	***PSNR***
IRM1	4	6.14	110.1559	2.8808	27.7107
	16	13.9633	505.5444	13.2209	21.0932
IRM2	4	4.4376	35.7413	0.5676	32.5991
	16	10.3698	184.1563	2.9243	25.4789
Cerveau (gris)	4	5.1036	65.1099	2.7045	29.9943
	16	10.3784	217.111	9.0183	24.7640
Cerveau (RGB)	4	6.2711	90.7234	3.4803	28.5536
	16	14.2033	393.8254	15.1504	22.1778
Rayon X	4	3.6206	19.1524	0.5988	35.3086
	16	7.8127	90.5546	2.8311	28.5617

Tableau II.12 Les compressions expérimentées

Les résultats obtenus dans le cadre des images médicales montrent bien l'efficacité de l'utilisation des réseaux RBF pour reconstituer les images compressées et surtout dans le cas des images de grandes dimensions (Rayon X).

II.6. Conclusion

Dans ce chapitre nous avons effectué une étude générale sur les réseaux de neurones à apprentissage supervisé spécialement les réseaux MLP et RBF.

D'après les résultats obtenus, l'algorithme de réseau de neurones RBF semble être le plus efficace pour la reconstruction des images médicales compressées par décimation. Cet algorithme est le plus rapide et permet d'obtenir de meilleurs résultats avec un nombre de neurones moins important que ceux du réseau MLP.

Chapitre III

Transformée de Cosinus Discret (DCT)

III.1. Introduction

Le développement des applications informatiques s'est accompagné d'un accroissement énorme de l'utilisation d'images, notamment dans le domaine du multimédia, des jeux, de l'imagerie médicale,…Les images posent, par leur taille importante, de nombreux problèmes quant à leur transmission et à leur stockage. Pour gagner en vitesse aussi bien qu'en place, il est nécessaire de ‘‘compresser’’ l'image [16].

A la fin des années 80, deux importants groupes de normalisation, le CCITT (Consultative Committe for International Telegraph and Telephone) et l'ISO (International Standards Organization) décidèrent de créer, une norme internationale pour la compression d'images fixes. La mise en place d'un standard international était devenue nécessaire pour archiver ou pour faciliter l'échange des images dans des domaines aussi variés que les photos satellites, l'imagerie médicale, la télécopie couleur ou la cartographie… C'est ainsi que fut créé le groupe JPEG (Joint Photographic Experts Group) à l'origine de la norme qui porte son nom.

Le codage JPEG est aujourd'hui largement utilisé dans les secteurs de l'informatique et de la communication (appareils photo numériques, scanners, imprimantes, télécopieurs…). Nous décrirons en première partie de ce chapitre cette norme de compression.

Nous avons, dans une deuxième partie, proposé une méthode de compression d'images en comparant par la suite les résultats obtenus avec ceux des méthodes : CBTC-PF et JPEG. A la fin, de ce chapitre, on présente quelques applications sur les images médicales.

III.2. Codage par Transformée

La plupart des algorithmes de compression avec pertes opèrent sur une transformation de l'image qui projette les données dans un autre espace plus propice à la quantification et au codage entropique. L'idée est qu'en effectuant une transformée linéaire sur l'image originale, on obtient un ensemble de coefficients dans l'espace transformé dont les composantes sont moins corrélées entre elles, plus compactes (c'est à dire que l'énergie du signal est concentrée dans un faible nombre de composantes au lieu d'être uniformément répartie). Les données transformées sont ensuite quantifiées pour subir un codage entropique en phase finale [18].

L'intérêt du codage par transformée réside entre autres dans le fait que la quantification des coefficients entraîne une perte d'information ‘‘répartie’’ sur les données initiales. Cela

permet de se référer plus facilement aux outils perceptifs humains et donc à l'évaluation subjective de la qualité de reconstruction de l'image [18].

Les techniques de compression dans le domaine fréquentiel s'appuient sur une transformation de l'image vers un nouvel espace de représentation d'énergie fortement décorrélée. Cette décorrélation provoque une nouvelle représentation de l'image par la redistribution de l'énergie dans un nombre restreint de coefficients transformés. Cette énergie de l'image transformée est distribuée sous la forme de tranches énergétiques de basse, moyenne et haute intensités. Les transformations d'espace les plus courantes sont la DCT et les ondelettes [19].

L'un des avantages de la DCT bidimensionnelle est qu'elle est séparable. C'est-à-dire qu'une DCT en deux dimensions se calcule comme deux DCT monodimensionnelles successives ce qui permet d'accélérer le calcul.

L'hypothèse de stationnarité du signal oblige à effectuer ces transformations non pas sur toute l'image mais sur des petits blocs de taille réduite non recouvrant par exemple une DCT sur des blocs 8×8. La conséquence de ce codage par bloc est que la quantification des coefficients issus des blocs transformés par DCT introduit la présence d'effets de blocs gênants et surtout pour des quantifications à très bas débit. Le fait de traiter l'image en blocs indépendants est une limitation des méthodes de transformée par bloc. En effet, les échantillons des blocs ''voisins'' ne sont pas forcément décorrélés avec les autres blocs, bien au contraire. On peut donc s'attendre à ce que les coefficients transformés issus de blocs adjacents présentent une certaine corrélation [18]. L'altération de la qualité issue par une compression avec perte, nous incite à essayer d'évaluer l'image décompressée. Il est à noter, que les critères d'évaluation de la qualité sont nombreux. Cependant, nous avons retenus le rapport signal sur bruit (*PSNR*), car, il est le plus utilisé dans la littérature spécialisée du domaine.

III.3. Norme JPEG

III.3.1. Principe de la compression JPEG

Le principe de l'algorithme JPEG [16] (figure III.1) pour une image en niveaux de gris (une image couleur est décomposée en un plan luminance et deux plans chromatiques qui sont traités de manière identique) est le suivant : La matrice des pixels de l'image est décomposée séquentiellement de gauche à droite et de haut en bas en blocs de 8×8 pixels. Tous les blocs subissent le même traitement. Une transformée en cosinus discret bidimensionnel (DCT pour

Discret Cosine Transform) est réalisée sur chaque bloc. Cette transformation concentre l'information de l'image en haut et à gauche de la matrice, correspondant aux basses fréquences. Les coefficients de la transformée sont ensuite quantifiés à l'aide d'une table de 64 éléments correspondant à un facteur de qualité. Cette table permet de choisir un pas de quantification important pour certaines composantes jugées peu significatives visuellement. En effet, les informations pertinentes d'une image, caractérisée par son signal bidimensionnel *Img*(*x,y*), sont concentrées dans les fréquences spatiales les plus basses. On introduit ainsi un critère perceptif qui peut être rendu dépendant des caractéristiques de l'image et de la taille de l'image désirée.

La perte de qualité se situe principalement dans l'étape de quantification et le sous échantillonnage des chrominances.

A la phase finale, des codages sans distorsion (différentiel, entropique, algorithme de Huffman ou arithmétique) sont ensuite réalisés en utilisant les propriétés statistiques des images. Pour cela, on commence par ordonner les coefficients suivant un balayage en zigzag pour placer d'abord les coefficients correspondant aux fréquences les plus basses. Cela donne une suite de symboles. Les codes les plus fréquents seront remplacés par d'autres codes comportant un nombre de bits le plus petit possible.

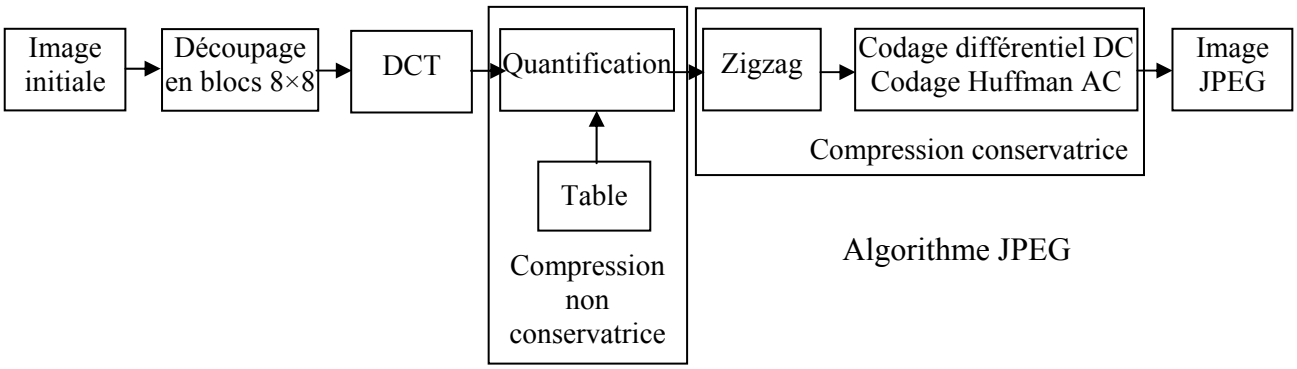


Figure III.1 Principe de la compression JPEG

Dans les sections qui suivent, chaque étape est décrite en détail.

III.3.1.1. Traitement des images couleur

Habituellement, les images couleur sont stockées sous le format RGB (Red Green Blue : Rouge Vert Bleu). Or le système RGB n'est pas le mieux adapté pour appliquer le codage JPEG. En effet, l'œil humain est plus sensible à la luminance qu'à la chrominance d'une image. Aussi une transformation est réalisée pour passer à un espace couleur, plus adapté à la compression. C'est l'espace YCbCr (Luminance, chrominance par rapport au bleu

et chrominance par rapport au rouge, la dernière chrominance étant déduite à partir des autres composantes) qui a été choisi [16].

III.3.1.2. DCT et IDCT

L'étape de DCT permet de représenter l'image dans une nouvelle base plus propice à la compression. Dans cette représentation, l'information est plus concentrée [16].

III.3.1.2.1. Calcul de la DCT et de l'IDCT

Chaque bloc 8×8 est soumis à une transformation par DCT (Discrete Cosine Transform). Le premier coefficient de la DCT, le DC (Direct Component), est proportionnel à la moyenne des valeurs du bloc. Les 63 autres coefficients sont appelés AC (Alternative Component). Ce nouveau domaine transformé permet une décorrélation très forte de l'information. Sur ce bloc de coefficients, les énergies sont groupées en basse, moyenne et haute fréquences. Avec la DCT, chaque colonne est une fonction cosinus de fréquence différente. La variance est alors concentrée sur les composantes de basse fréquence, et les composantes de haute fréquence seront annulées par quantification [19].

La DCT travaille sur une matrice carrée représentant un bloc de l'image globale *Img*(*x,y*). Le nombre d'échantillons sur l'axe des *X* doit donc être égal au nombre d'échantillons sur l'axe des *Y*. Pour fixer les idées, nous disons que notre bloc est de taille (*N*×*N*) 8 par 8 pixels préconisés dans la norme) [16].

La DCT et la IDCT fournissent, intuitivement, chacune une matrice carrée. Leurs formules sont données ci-dessous :

$$DCT(i,j)=\frac{1}{\sqrt{2N}}c(i)c(j)\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}Img(x,y)\cos\left(\frac{(2x+1)i\pi}{2N}\right)\cos\left(\frac{(2y+1)j\pi}{2N}\right), \qquad \text{(III.1)}$$

$$IDCT(i,j)=\frac{1}{\sqrt{2N}}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}c(i)c(j)DCT(i,j)\cos\left(\frac{(2x+1)i\pi}{2N}\right)\cos\left(\frac{(2y+1)j\pi}{2N}\right), \qquad \text{(III.2)}$$

avec $$c(i)=\begin{cases}\frac{1}{\sqrt{2}} & si\ i=0\\ 1 & si\ i>0\end{cases}$$

Il est clair que l'application de la IDCT sur touts les blocs DCT mène à avoir l'image *Img*.

Les formules (III.1) et (III.2) montrent que pour calculer la DCT en un point, il est indispensable de parcourir toutes les valeurs du bloc relatif. La double somme oblige, d'un

point de vue algorithmique, à effectuer N^2 calculs de termes. Il en est de même pour le calcul de la IDCT.

Il est évident que la DCT est conservative si l'on ne tient pas compte des erreurs d'arrondis qu'elle introduit.

Lorsqu'on travaille avec le signal *Img*(*x,y*), les axes *X* et *Y* représentent les dimensions horizontales et verticales de l'image (domaine spatial). Lorsqu'on travaille avec la transformée en cosinus discret *DCT*(*i,j*), les axes représentent les fréquences du signal en deux dimensions [16].

III.3.1.3. Quantification

La quantification est la phase non conservatrice du processus de compression JPEG (excepté les arrondis effectués). Elle amène, moyennement à une diminution de la précision de l'image, mais aussi à réduire le nombre de bits nécessaires au stockage. Pour cela, elle réduit chaque valeur de la matrice DCT en la divisant par un nombre (quantum) fixé par une table (matrice 8×8) (figure III.2, III.3) de quantification suivant l'équation (III.3) :

$$Valeur\ quantifiée(i,j) = \frac{Valeur\ DCT(i,j)}{quantum(i,j)}, \qquad \text{(III.3)}$$

Cette valeur est arrondie à l'entier le plus proche. Ultérieurement, lors de la restitution de l'image (décompression), il suffira de réaliser l'opération inverse (déquantification) en multipliant chaque valeur de la matrice quantifiée par le quantum correspondant, pour retrouver une matrice DCT déquantifiée, à partir de laquelle sera établie la matrice des pixels de sortie.

Comme l'œil est moins sensible aux hautes fréquences qu'aux basses fréquences, la diminution de précision sera d'autant plus forte que les fréquences sont élevées, car les informations qu'elles contiennent sont moins pertinentes. La valeur du quantum sera d'autant plus élevée que l'élément correspondant de la matrice DCT contribue peu à la qualité visuelle de l'image, donc qu'il se trouve éloigné dans la séquence zigzag.

C'est pourquoi les matrices de quantification comportent généralement des valeurs constantes selon des diagonales ascendantes (*i, j, i-1, j+1*), mais croissantes d'une diagonale à la suivante : cet accroissement constitue le pas du quantum, et est lié au ''facteur de qualité''.

Les nombreux tests réalisés ont conduit à retenir en pratique des facteurs de qualité compris entre 1 (l'image reste excellente) et 25 (dégradation encore acceptable). Il est important d'observer que, même avec des pas de quantum faibles, la quantification conduit à une valeur nulle un certain nombre d'éléments situés dans le coin inférieur droit de la

matrice : ces éléments ne seront pas restitués par la déquantification, d'où le caractère non conservateur de cette opération.

Bien que la spécification JPEG n'impose aucune contrainte sur la matrice de quantification, l'organisme de standardisation *ISO* a développé un ensemble standard de valeurs de quantifications utilisables par les programmeurs de code JPEG. Ce choix a été rendu possible grâce aux tests intensifs des matrices via des observateurs.

Voici celles préconisées par la norme JPEG (figure III.2 pour la luminance et figure III.3 pour les chrominances) :

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

Figure III.2 Matrice de quantification pour la luminance (recommandation JPEG)

17	18	24	47	99	99	99	99
18	21	26	66	99	99	99	99
24	26	56	99	99	99	99	99
47	66	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99

Figure III.3 Matrice de quantification pour les chrominances (recommandation JPEG)

Habituellement, la matrice de quantification *Q* est obtenue à partir de formules plus ou moins simples, permettant de ‘‘choisir’’ la perte de qualité. La formule suivante est un exemple qui donne de bons résultats :

$Q = q(i, j)$ avec $q(i, j) = 1 + K(1 + m(i + j))$

Avec i l'indice de ligne, j l'indice de colonne, m une constante (souvent égale à 1) et K le facteur de qualité (choisi entre 1 et 25).

III.4. Les courbes de scanning

III.4.1. Définition et intérêt

L'ordonnancement des coefficients constituant un bloc DCT représente une étape primordiale dans notre stratégie de compression proposée. La littérature spécialisées du domaine évoque plusieurs ''Scanning'' courbes à savoir :

Scan de : Hilbert, zigzag, Regazzoni, lexicographique, L'entrelacement de bits [45].

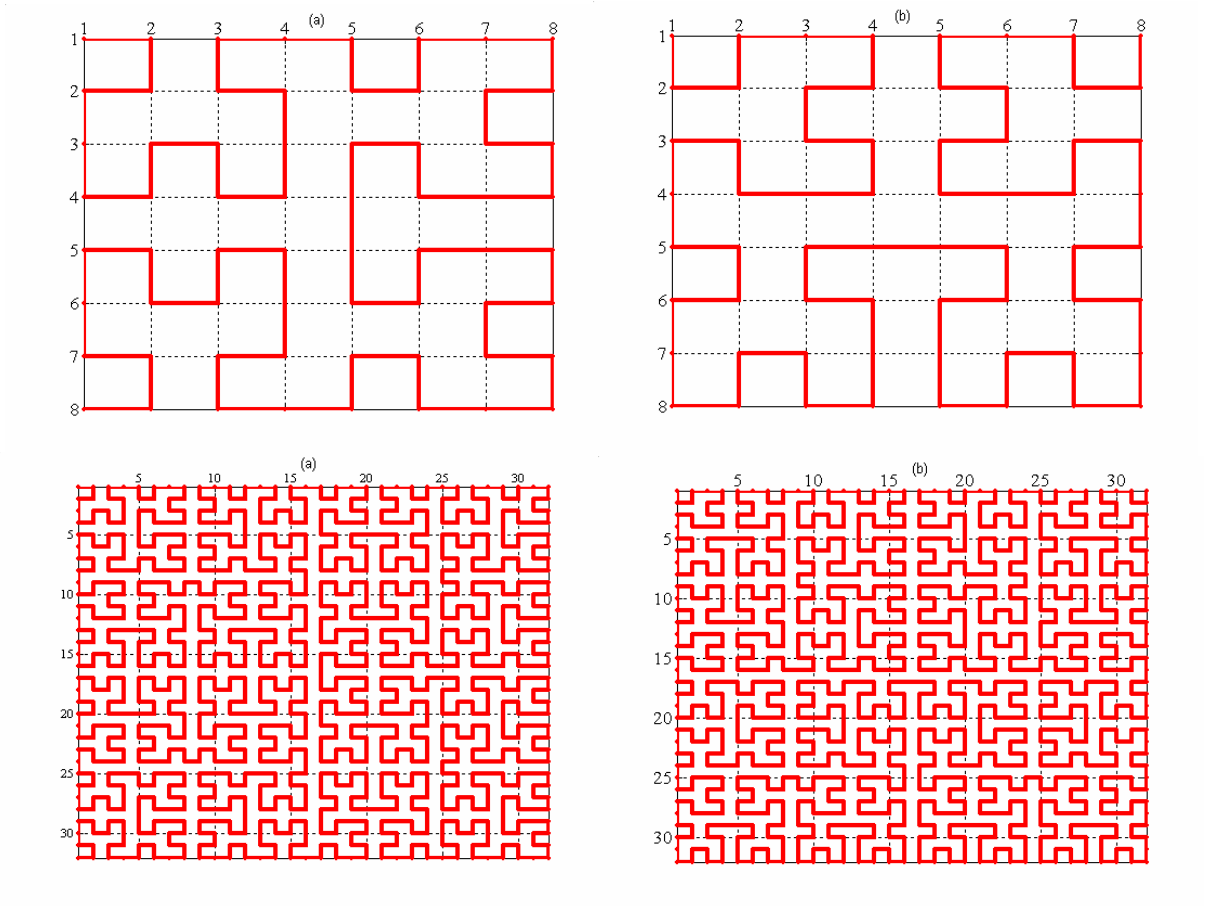


Figure III.4 Courbe de Hilbert

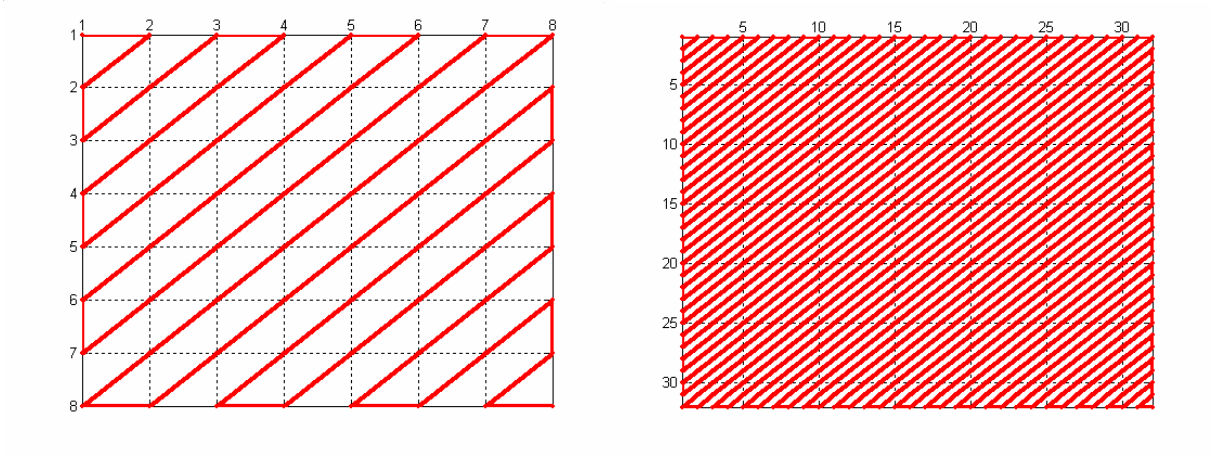


Figure III.5 ‘‘zig-zag scan’’

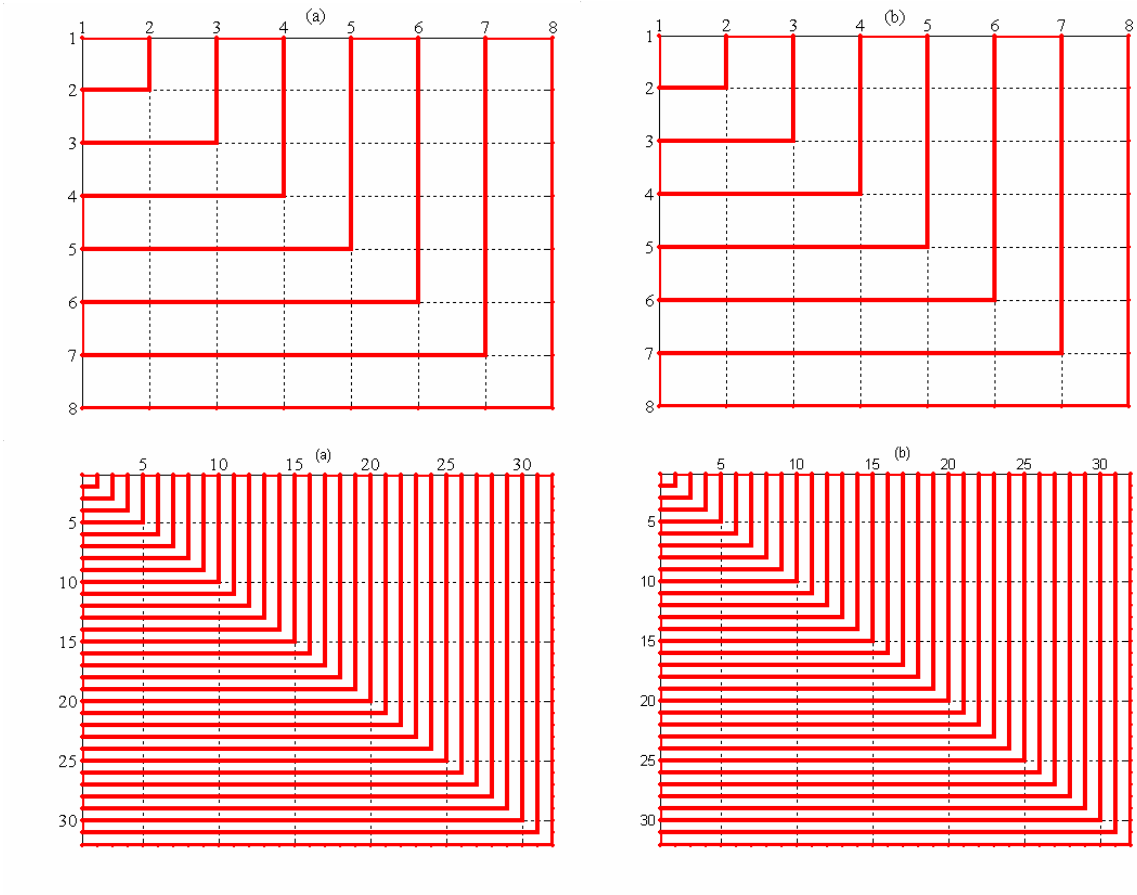


Figure III.6 Courbe proposée par Regazzoni

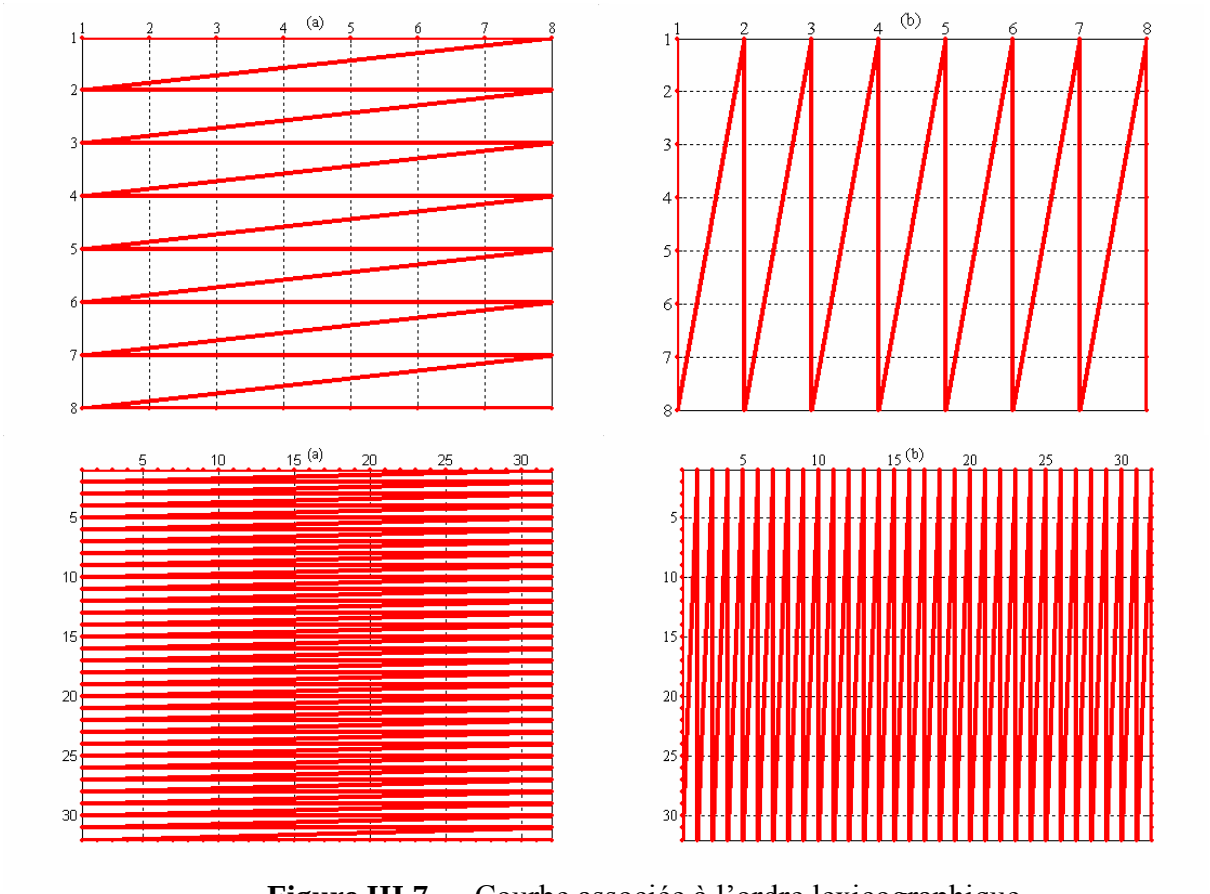


Figure III.7 Courbe associée à l'ordre lexicographique

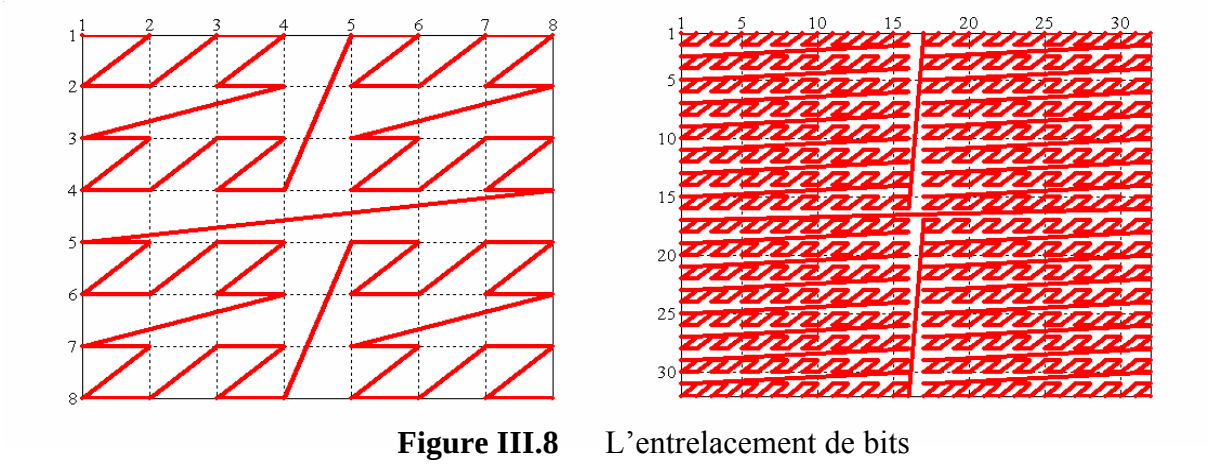


Figure III.8 L'entrelacement de bits

Dans notre algorithme nous avons travaillé avec les huit ‘‘Scans’’ (bloc 8×8) ou (bloc 32×32) mentionnés précédemment de manière à avoir d’une façon adaptative le meilleur résultat en terme *PSNR-Cr* en exploitant pour chaque bloc DCT la courbe adéquate.

III.5. Méthode proposée

III.5.1. Compression d’image

On présente dans la suite la compression d’images de l’approche proposée. Cette compression est présentée dans la figure III.9, les images couleur comprennent trois plans qui subissent chacun le même procédé entre autre, avec un changement d’espace de couleur RGB vers YCbCr, Cette transformation permet une compression plus efficace car l’information est principalement condensée dans le plan Y, on peut donc comprimer plus efficacement Cb et Cr.

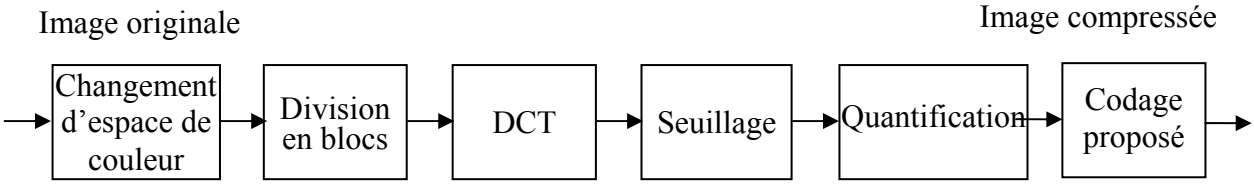


Figure III.9 Schéma de Principe de la compression

La décompression de l’image effectue les traitements dans l’ordre inverse (figure III.10).

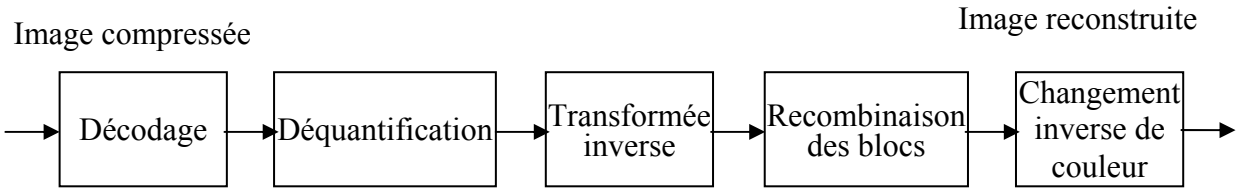


Figure III.10 Décompression d’image

Les différentes étape de notre méthode se résument en :

- Prétraitement et division en blocs : une soustraction de la valeur moyenne sur chaque plan est appliquée sur l’image couleur originale, Ceci mène à une image reconstruite à valeur moyenne nulle, le découpage consiste à diviser l’image en blocs sur lesquels sont appliquées indépendamment les étapes suivantes. La principale raison de ce découpage est de limiter le

nombre de pixels à traiter à la fois pour diminuer le temps de calcul et la complexité des circuits électroniques. Il peut résulter de cette division un effet visuel appelé effet de blocs à des taux de compression élevés, la frontière des blocs devient visible car ils ont été comprimés indépendamment. La taille des blocs est variable selon les méthodes. Dans notre cas nous avons destiné deux blocs 8×8 ou 32×32.

- L'application de DCT : on calcule la DCT de chaque bloc, ce qui permet de transformer les pixels d'un bloc 8×8 (32×32) d'une image en un autre bloc de 8×8 (32×32) contenant les composantes fréquentielles correspondantes. On trouve des coefficients qui caractérisent les basses fréquences et d'autres qui caractérisent les hautes fréquences.

Vu que la variation des intensités des pixels dans un bloc 8×8 (32×32) est très lente, la majorité de l'énergie se situe dans les fréquences basses. La transformée en cosinus discrète DCT permet donc de concentrer cette énergie dans quelques coefficients.

Les deux motivations principales à l'utilisation d'une transformation sont :

- l'obtention d'une représentation de l'image qui se prête bien à la quantification et au codage.
- la possibilité d'ajuster les erreurs de quantification selon la sensibilité au système visuel humain.

- Seuillage : les coefficients obtenus sont exposés à un seuillage qui permet la mise en évidence de deux classes de coefficients :

Les coefficients inférieurs ou égaux à *TH* (qui sont mis à zéro) et les coefficients dominants supérieurs à *TH*.

Plus le seuil *TH* est élevé, plus le taux de compression est grand, mais une distorsion d'image importante est inévitable.

- Quantification : les vecteurs de coefficients non nuls sont en suite quantifiés par un quantificateur linéaire de taille 8 bits, l'objectif de cette quantification est de réduire le nombre de bits nécessaires à la représentation des coefficients de façon que cette réduction n'apporte pas de dégradation visuelle notable à l'image.

La quantification Q des coefficients X est effectuée selon la formule :

$$Q = \left\lfloor \frac{X - X_{\min}}{X_{\max} - X_{\min}} \times (2^p - 1) \right\rfloor \qquad \text{(III.4)}$$

L'espace quantifié Q est constitué de valeurs scalaires réparties suivant un intervalle régulier (entre 0 et 255).

où $\lfloor \ \rfloor$ représente la valeur plus proche.

X_{min} valeur minimale de X.

X_{max} valeur maximale de X.

p nombre de bits de quantification.

La déquantification se fait selon :

$$X_{dq} = \frac{Q}{(2^p - 1)} \times (X_{\max} - X_{\min}) + X_{\min} \tag{III.5}$$

Où X_{dq} est le coefficient reconstruit.

Les données sont maintenant prêtes à être transformées en un train binaire (bit stream), pour être utilisées par la suite par le codeur sans perte qui sera détaillé dans la section III.5.2 qui suit.

Pour la reconstruction de l'image traitée on applique le décodage, la déquantification, et la transformée en cosinus discret inverse.

Le *PSNR* est calculé après l'addition de la valeur moyenne précédemment retranchée aux images originales et reconstruites.

Pour valider notre approche nous avons fait une étude comparative avec deux méthodes récentes : CBTC-PF [13], JPEG.

L'évaluation d'un algorithme de compression s'effectue communément suivant : la mesure de qualité (*MSE*, *PSNR*,..), (taux de compression, *bpp*) ainsi que temps d'exécution. La comparaison a été faite en utilisant le *bpp* et le *PSNR* qui sont définis respectivement par [13] :

$$bpp = \frac{Taille\ d'image\ couleur\ compressée\ en\ bits}{nombre\ de\ pixels} \tag{III.6}$$

$$PSNR = 10 \times log_{10} \frac{255^2 \times 3}{MSE(R) + MSE(G) + MSE(B)} \tag{III.7}$$

Le *PSNR* est contrôlé d'avance en utilisant la dichotomie. Celle ci est détaillée en ce qui suit [46] :

1. Initialisation par :

- Fixer en avance une valeur désirée dite $PSNR_d$ à atteindre par *PSNR*.
- Sélectionner l'intervalle de recherche $[TH_{min}, TH_{max}]$.
- Donner la précision de convergence ε .

- Calcule $TH = \frac{TH_{\min} + TH_{\max}}{2}$.

2. Calcul :

Seuillage

- Mettre à zéros tous les coefficients inférieurs ou égaux à *TH*.

Calculer *PSNR* en fonction de *TH*

- calculer le *PSNR* de l'image seuillée pour la valeur courante de *TH*.

3. Mise à jour du coefficient de *TH*

- si ($PSNR<PSNR_d$) alors $TH_{min}=TH$, sinon $TH_{max}=TH$.
- $TH = \frac{TH_{\min} + TH_{\max}}{2}$

4. Condition de terminaison

- si $\left|PSNR - PSNR_d\right| > \varepsilon$ aller à Etape de 2, Sinon STOP.

Notre objectif est d'obtenir une valeur de *PSNR* plus proche (acceptable) prévue d'avance.

Deux problèmes classiques rencontrés dans la compression de l'image qui sont [18] :

- les pertes irréversibles sont introduites lors de la phase de quantification qui consiste à réduire l'espace de représentation des données.
- la transformée a pour objectif de projeter les données originales dans un espace plus propice à la compression. En d'autres termes, il s'agit de décorréler le signal de manière à minimiser l'information redondante.

Le calcul de la valeur de *TH* est donc à modifier, par conséquent l'étape de calcul peut se résumer comme suit :

Calcul :

- seuillage.

$XDCT=0$ si $\left|XDCT\right| \le TH$.

avec :

- Quantification.

$$XDCTQ = \left\lfloor \frac{XDCT - XDCT_{\min}}{XDCT_{\max} - XDCT_{\min}} \times 254 + 1 \right\rfloor$$

$XDCT_{min}$ est la valeur minimale de l'image après la transformée.

$XDCT_{max}$ est la valeur maximale de l'image après la transformée.

XDCTQ l'image après la transformée et la quantification.

- Les traitements dans l'ordre inverse
  - ✓ Déquantification

$$XDCTDQ = \frac{(XDCTQ - 1)}{254}(XDCT_{\max} - XDCT_{\min}) + XDCT_{\min}$$

XDCTDQ l'image déquantifiée.

Ce nouvel algorithme nous apporte une solution à ce problème que nous allons préciser.

III.5.2. Compression sans perte

L'étape du codage est décrite comme suit : L'image est découpée dans notre approche suivant deux tailles de bloc, le découpage 8×8 ou 32×32 pixels, le scanning s'effectue selon huit chemins différents possible (Hilbert (a,b), zigzag, Regazzoni (a,b), horizontale, verticale, entrelacement de bits), présentés précédemment.

Ensuite, on applique sur chaque bloc de 8×8 pixel le codeur RLE classique (sur l'ensemble des coefficients AC) qui permet d'encoder de manière efficace les longues suites de zéros répétitives, alors toute suite de zéros est remplacée par un couple comme (zéros size).

Mais le découpage en 32×32 pixels contient 1024 éléments, pour obtenir le codage sur 8 bits nous avons proposé les étapes de partition suivante :

$$R = \begin{cases} [0\ S] & si\ S \le 255 \\ [0\ 0\ (S\text{-}255)] & si\ 255 < S \le 510 \\ [0\ 0\ 0\ (S\text{-}510)] & si\ 510 < S \le 765 \\ [0\ 0\ 0\ 0\ (S\text{-}765)] & si\ 765 < S \le 1020 \\ [0\ 0\ 0\ 0\ 0\ (S\text{-}1020)] & si\ 1020 < S \le 1024 \end{cases}$$

où *S* détermine le nombre des zéros et *R* plage de codage.

Par exemple : Après découpage en bloc de 32×32 pixel et réarrangement de parcours d'image, on obtient un vecteur de 1024 éléments :

[255 13 **0 0 ….0 0 0 0 0 0** 70 30 1 50 0….0]

700 éléments des zéros

Après le codeur RLE on obtient le vecteur suivant :

[255 13 **0 0 0 190** 70 30 1 50].

Ce codage remplace une suite de valeurs par un bloc spécial *B* (figure III.11), le bloc est composé de (*D*, *PDNZ*, *M*), où (*D*) est l'activation du bloc de 1 bits, (*PDNZ*) position de la dernière valeur non nulle, avec *PDNZ=NZ*-1, où *NZ* est le nombre de la dernière valeur non nulle, et (*M*) est une plage de 3 bits pour le choix de la meilleure courbe de scanning. Le but de cette étape est de trouver le meilleur chemin pour obtenir un taux de compression élevé, lorsque *M*='111' correspond a la huitième courbe. Cependant, le nombre de bits pour représenter (*PDNZ*) est de 6 bits dans le cas découpage en 8×8 pixels, de 10 bits pour le deuxième cas (32×32 pixels).

Le bloc *L* coûte 8×*NZ* bits, chaque coefficient de *L* est codé sur 8 bits.

La figure III.11 présente la disposition du bloc *B* et leurs contraintes.

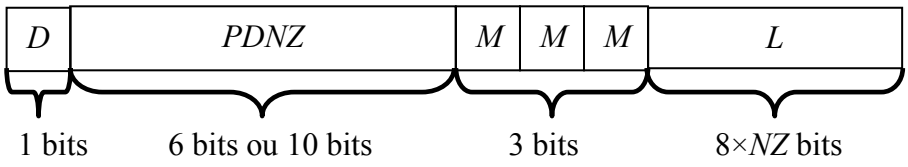


Figure III.11 La représentation du *B*

Sur chaque bloc, on effectue les étapes suivantes :

- Si les éléments de la matrice de l'image sont nuls (*L*=[0 0 0 … 0]), donc *D*='0', le bloc *B* est désactivé (en code *B*='0' sur 1 bit), si non, on teste la valeur de *PDNZ* par chaque scanning ensuite on choisit la valeur minimale de *PDNZ* qui donne le meilleur scanning, pour déterminer le taux de compression optimal, ensuite on définit le nombre de la courbe qui correspond à la valeur de *M* avec *D*='1' (bloc activée).

L'équation (III.8) représente l'espace total *TB* de bloc *B* de l'image compressée.

$$TB = \sum_{i=1}^{x_b} (m + 8 \times NZ_i) \times D_i + x_b \quad \text{(III.8)}$$

Dans l'étape de décompression, les équations suivantes déterminent les valeurs minimale et maximale de chaque coefficient du bloc *B* :

$$PD_j = \sum_{i=1}^{j-1} (m + 8 \times NZ_i) \times D_i + j \quad \text{(III.9)}$$

$$\sum_{i=1}^{j-1} (m + 8 \times NZ_i) \times D_i + j + 1 \le PNZ_j \le \sum_{i=1}^{j-1} (m + 8 \times NZ_i) \times D_i + (m-3) \times D_j + j \quad \text{(III.10)}$$

$$\sum_{i=1}^{j-1} (m + 8 \times NZ_i) \times D_i + (m-3) \times D_j + j + 1 \le PM_j \le m \times \sum_{i=1}^{j} (D_i) + 8 \times \sum_{i=1}^{j-1} (NZ_i \times D_i) + j \quad \text{(III.11)}$$

$$m \times \sum_{i=1}^{j} (D_i) + 8 \times \sum_{i=1}^{j-1} (NZ_i \times D_i) + j + 1 \le PL_j \le \sum_{i=1}^{j} (m + 8 \times NZ_i) \times D_i + j \quad \text{(III.12)}$$

où x_b est le nombre de bloc, m est 9 pour découpage des blocs en 8×8 est 13 pour 32×32 et j=2,…,x_b.

PD_i, PNZ_i, PM_i, PL_i sont les positions de D_i, NZ_i, M_i, L_i.

III.5.3. Les Résultats de la simulation

La simulation numérique est une étape importante dans l'étude et la mise en ouvre de traitement et compression d'image. En effet, elle permet, d'une part, la validation des études théoriques et, d'autre part, l'optimisation du système par l'analyse de l'impact des divers paramètres sur les performances.

Nous présentons dans cette partie, les résultats issus de la simulation numérique en (langage Matlab).

III.5.3.1. Résultats sur les images aux niveaux de gris

Dans la figure III.12, nous comparons la compression de l'image ‘‘Lena’’ (niveau de gris) obtenue par JPEG face à ceux de notre approche [12], ceci est effectué pour deux taux de compression différents, avec deux types de découpage.

On fixe la valeur du *PSNR* à 24db et à 30db, ce qui donne des taux de compression *Cr* de 59 et 32 obtenus en utilisant JPEG, par contre lorsque l'on calcule le *Cr* entre deux types de découpage 8×8 et 32×32 donnera un taux de compression de 38.96, 19.76 et 123.9, 32.75 respectivement, comme on peut voir dans le tableau III.1.

(a)JPEG
Cr 59
PSNR 24.16db

(b)JPEG
Cr 32
PSNR 29.7db

(c) bloc 8×8
Cr 38.9595
PSNR 24.15db

(d) bloc 8×8
Cr 19.7630
PSNR 29.95db

(e) bloc 32×32
Cr 123.9012
PSNR 24.9db

(f) bloc 32×32
Cr 32.746
PSNR 29.74db

Figure III.12 Comparaison des compressions de JPEG et avec notre méthode d'image de ''Lena''

	PSNR	***Cr***	***bpp***
JPEG	24.16	59	0.14
	29.7	32	0.25
Bloc 8×8	24.15	38.9595	0.2053
	29.95	19.763	0.4048
Bloc 32×32	24.9	123.9	0.0646
	29.74	32.746	0.2443

Tableau III.1 Comparaison des résultats.

On conclut que notre l'algorithme dépasse d'une manière significatif le standard JPEG. Ceci est en fixant bien sure le même *PSNR*. Voir tableau III.1.

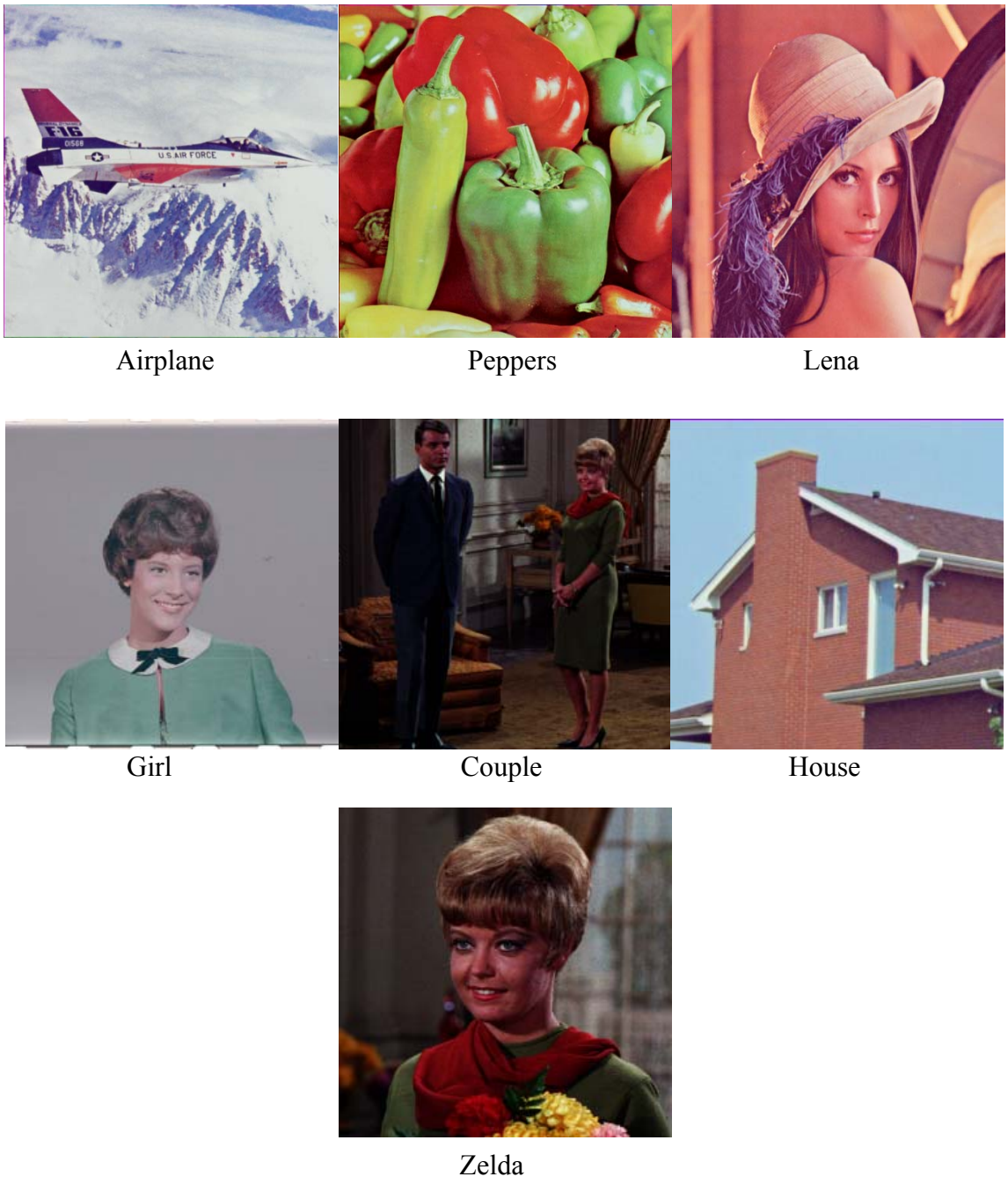
Une autre comparaison a été effectuée avec la méthode proposée dans [13]. Notre approche a donné un *PSNR* 31.73db avec un *bpp* de 0.39bits/pixel tandis que celle de [13] a donné un *PSNR* de 31.59db avec 0.64bits/pixel. Ces résultats démontre la supériorité de notre algorithme.

TH 38.9063
PSNR 31.7336db
bpp 0.3960
Cr 20.2038

Figure III.13 Débit de compression (*bpp*) atteint par le codeur

III.5.3.2. Résultats sur les images couleur

Les images [47] que nous avons utilisé sont des images couleur (RGB) de 24 bits de différentes tailles : ‘‘Airplane’’, ‘‘Peppers’’ et ‘‘Lena’’ de 512×512 ; ‘‘Girl’’, ‘‘Couple’’, ‘‘House’’ et ‘‘Zelda’’ de 256×256. Elles sont présentées sur la figure III.14.



Airplane Peppers Lena

Girl Couple House

Zelda

Figure III.14 Les différentes images originales

Pour évaluer numériquement les performances du codeur, nous avons calculé le rapport signal sur bruit crête à crête (*PSNR*) et le nombre de bits par pixels, les résultats mentionnés sur le tableau III.2.

RGB				
image	***TH***	***PSNR***	***bpp***	***Cr***
Bloc 8×8				
Airplane	53.125	30.3695	1.4608	16.4292
Peppers	46.25	30.1549	1.4072	17.0548
Lena	34.375	31.9014	1.7781	13.4977
Girl	30.1563	35.1254	1.4939	16.0654
Couple	28.75	32.4355	2.0923	11.4708
House	38.75	31.7901	1.6842	14.2501
Zelda	34.2188	31.3095	1.8712	12.8261
Moyenne	37.9464	31.8695	1.6840	14.5134
Bloc 32×32				
Airplane	44.2188	30.6064	1.1870	21.5013
Peppers	34.6875	30.1580	1.4020	17.1179
Lena	30	31.8981	1.5496	15.4880
Girl	21.9531	34.9971	1.4784	16.2338
Couple	23.9844	32.4386	2.3673	10.1381
House	31.25	31.7871	1.5559	15.4251
Zelda	28.9063	31.3051	1.8684	12.8452
Moyenne	30.7143	31.8843	1.6298	15.5356

Tableau III.2 Les performances mesurées en appliquant l'approche directement sur les images (RGB) reconstituées

Le tableau III.2 illustre les taux de compression et les *PSNR* obtenus pour les deux tailles de découpage 8×8 et 32×32.

En observation générale, la qualité des images restaurées obtenues avec le découpage 32×32 est meilleure pour la reconstruction.

Les images reconstruites selon la méthode proposée sont illustrées sur la figure suivante :

Airplane
PSNR 30.6db
bpp 1.187

Peppers
PSNR 30.16db
bpp 1.4

Lena
PSNR 31.9db
bpp 1.55

Girl
PSNR 35db
bpp 1.478

Couple
PSNR 32.44db
bpp 2.37

House
PSNR 31.79db
bpp 1.55

Zelda
PSNR 31.3db
bpp 1.87

Figure III.15 Les images reconstituées

Les images montrées sur la figure III.16 sont issues lors de l'application de notre approche, mais après un changement d'espace de couleur RGB vers YCbCr. Cette transformation permet une compression plus efficace. Les résultats obtenus sont rapportés dans le tableau III.3.

YCbCr					
Image	***TH***	***PSNR YCbCr***	***PSNR RGB***	***bpp***	***Cr***
Bloc 8×8					
Airplane	38.4375	34.5033	30.3366	0.9804	24.4793
Peppers	25.1563	34.2323	30.0185	1.4490	16.5627
Lena	26.25	35.8639	31.5752	1.2880	18.6338
Girl	22.4219	39.5803	34.7889	0.7446	32.2335
Couple	22.6563	36.6668	32.2671	1.3755	17.4478
House	27.3438	36.0813	31.5784	1.3520	17.7510
Zelda	25.6250	35.5420	31.0732	1.3460	17.8301
moyenne	26.8415	36.0671	31.6626	1.2194	20.7055
Bloc 32×32					
Airplane	35.25	34.7524	30.5854	0.7381	32.5167
Peppers	19.8438	35.0318	30.9369	1.4685	16.3427
Lena	21.5625	36.3101	31.9513	1.0054	23.8705
Girl	18.6719	39.7850	35.1323	0.6914	34.7142
Couple	18.6719	36.7123	32.3664	1.3588	17.6631
House	22.3438	36.4097	31.8760	1.0879	22.0610
Zelda	22.1875	35.6334	31.1993	1.1250	21.3333
moyenne	22.6473	36.3764	32.0068	1.0679	24.0716

Tableau III.3 Les performances mesurées sur les images (YCbCr) reconstituées

Nous pouvons voir que le taux de compression obtenu dans l'espace YCbCr est supérieur à celui de l'application directe sur l'espace RGB et ceci est évidemment en termes *PSNR* en moyenne.

La figure III.16 présente les images reconstruites dans l'espace YCbCr.

Airplane
PSNR 30.58db
bpp 0.7381

Peppers
PSNR 30.94db
bpp 1.468

Lena
PSNR 31.95db
bpp 1

Girl
PSNR 35.13db
bpp 0.6914

Couple
PSNR 32.37db
bpp 1.3588

House
PSNR 31.879db
bpp 1.0879

Zelda
PSNR 31.199db
bpp 1.125

Figure III.16 Les images reconstruites

Nous trouvons dans les tableaux III.2 et III.3 les résultats de codage des images citées ci-dessus avec des débits différents. Nous constatons que la transformée de l'espace de couleur RGB vers YCbCr améliore les différentes métriques (*PSNR*, *bpp*, *Cr*).

Pour valider notre travail, il est intéressant de comparer les résultats de notre méthode avec ceux la méthode (CBTC-PF). Cette comparaison est illustrée dans le tableau III.4.

CBTC-PF			**YCbCr**	
image	***PSNR***	***bpp***	***PSNR***	***bpp***
Airplane	30.36	1.04	30.5854	0.7381
Peppers	30.15	1.5	30.9369	1.4685
Lena	31.93	1.17	31.9513	1.0054
Girl	35.13	0.6	35.1323	0.6914
Couple	32.44	1	32.3664	1.3588
House	31.79	1.2	31.8760	1.0879
Zelda	31.31	1.12	31.1993	1.1250
moyenne	31.8729	1.09	32.0068	1.0679

Tableau III.4 Comparaison des résultats avec CBTC-PF [13]

Des résultats rapportés dans le tableau III.4 montrent clairement que notre méthode et la méthode CBTC-PF sont comparables. Cependant, on peut voir que notre technique est légèrement supérieur en termes *PSNR-bpp* en moyenne.

III.5.3.3. Application sur les images médicales

Les méthodes de compression basées sur la Transformée en Cosinus Discrète (*DCT*) sont largement répandues dans des applications d'imagerie médicale (stockage, transmission) pour garantir l'interactivité rapide. Ces applications imposent des contraintes spécifiques sur la qualité d'image, le débit binaire, le coût et la vitesse de compression décompression.

Pour mesurer l'évaluation de la qualité des images médicales compressées, ceci est réalisé en terme du *PSNR* (le rapport signal sur bruit crête) (Peak Signal Noise Ratio). La mesure du *PSNR* jusqu'à aujourd'hui est considérée comme le critère d'évaluation de la qualité le plus utilisé en traitement de l'image, cependant le *PSNR* est une mesure quantitative et nécessite parfois une évaluation subjective de la dégradation.

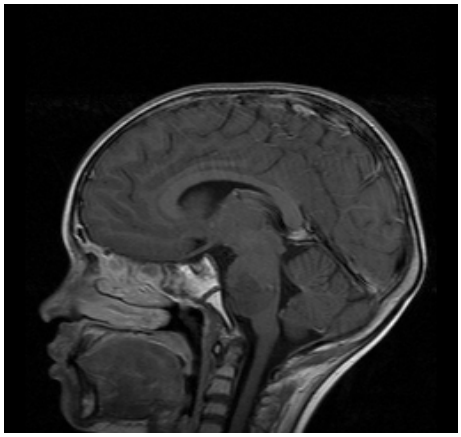
La compression des images est souvent associée aux différents algorithmes d'encodage. Nous nous sommes intéressés dans notre cas à deux algorithmes d'encodage JPEG et CBCT-PF.

Actuellement, la compression dans un service de radiologie est toujours effectuée sans perte. Ce type de compression avec une reconstruction exacte de l'image de départ, garantit

l'intégrité des données et demeure l'outil des praticiens pour des raisons évidentes de diagnostic.

La figure III.17 présente une image IRM. La figure III.17 (a) présente l'image originale (size 256×256). Les deux autres images ont été obtenues par la méthode proposée, en utilisant deux différents seuils *TH* pour un *PSNR* fixé d'avance.

Comme on peut le voir, un seuil *TH* de 35.47 aboutit à une image sur laquelle on remarque certaines dégradations faibles, mais la qualité visuelle est bonne. Cependant, le taux de compression est modeste. Pour comprimer avec un taux de 30, il faut augmenter le seuil à 63.75. L'augmentation du taux de compression est suivie par une dégradation visuelle de la qualité des images reconstruites et ceci est essentiellement du à l'effet de blocs.



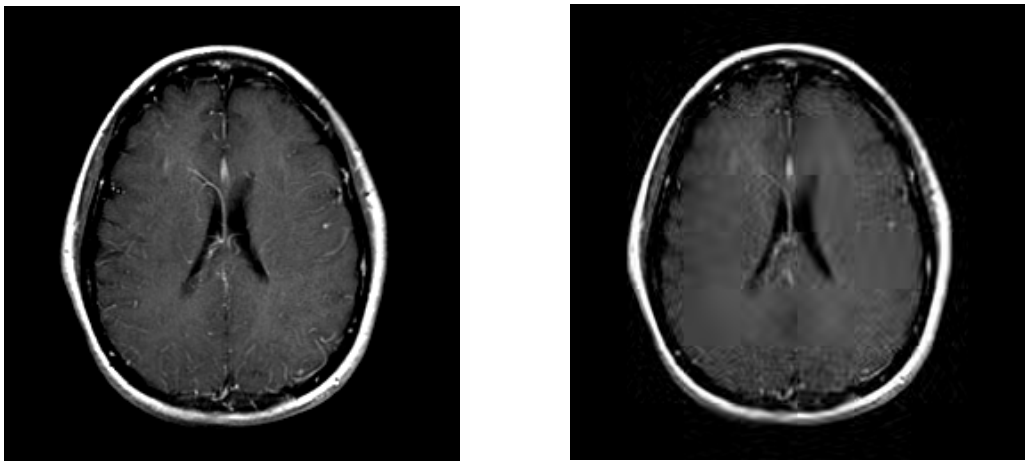
(a) image cerveau (gris)

(b) *Cr* 13.9743
bpp 0.5725
PSNR 31.3db

(c) *Cr* 30.4677
bpp 0.2626
PSNR 28.3db

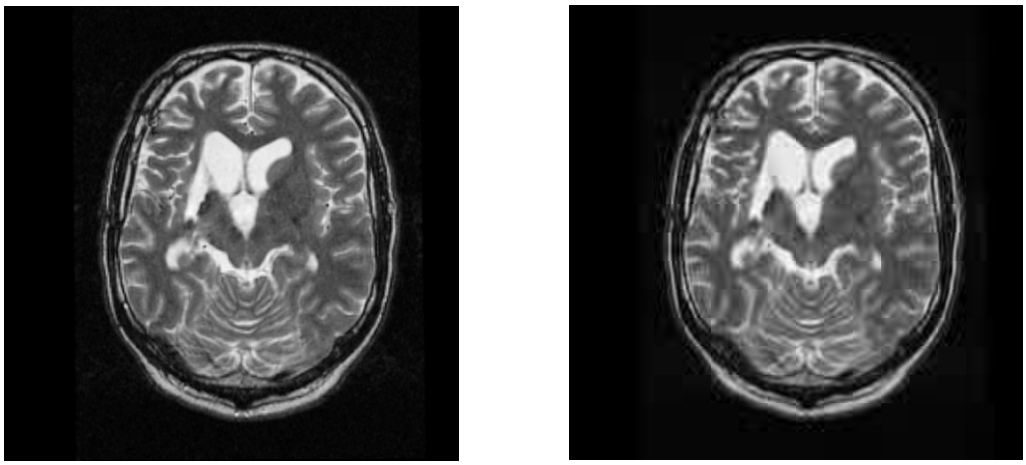
Figure III.17 Compression d'une image cerveau (gris) par l'approche proposée
(a) Image originale (b) et (c) Images reconstruites

Dans les figures suivantes sont représentées les images reconstruites. Le *PSNR* est supérieur à 30db pour l'ensemble des images (IRM, rayon X) de tests.



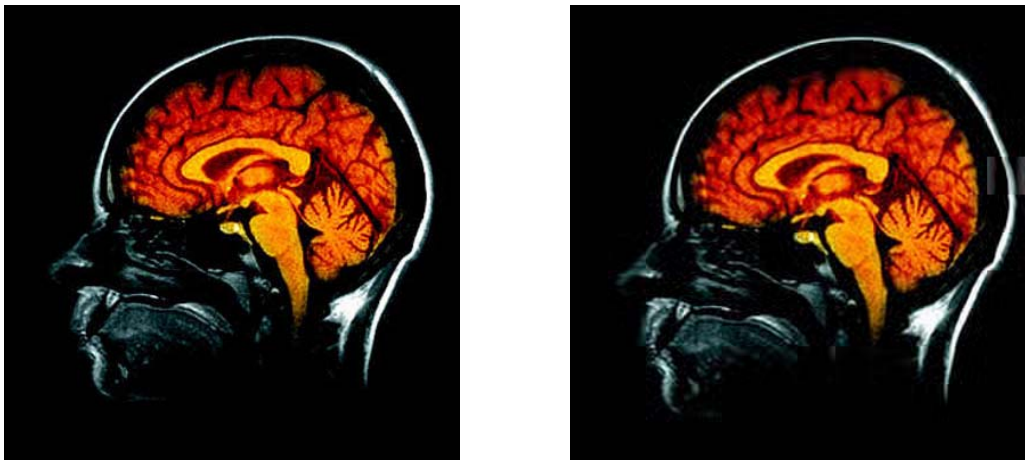
(a) (b)

Figure III.18 (a) Image originale (size 256×256), (b) Image IRM 1 reconstruite



(a) (b)

Figure III.19 (a) Image originale (size 512×512), (b) Image IRM 2 reconstruite



(a) (b)

Figure III.20 (a) Image originale (size 400×400), (b) Image cerveau (couleur) reconstruite

(a) (b)

Figure III.21 (a) Image originale (size 864×544), (b) Image rayon X reconstruite

Les résultats obtenus pour les cinq images de tests sont présentés dans le tableau qui suit :

Image	***PSNR***	***Cr***	***bpp***
Cerveau (gris)	31.3	13.9743	0.5725
	28.3	30.4677	0.2626
IRM 1	31.5342	10.8521	0.7372
IRM 2	32	37.2417	0.2148
Cerveau (couleur)	30.8744	19.3081	1.2430
Rayon X	36.3185	34.8696	0.2294

Tableau III.5 les performances mesurées sur les images médicales

Malgré l'importance des travaux dans le domaine de la compression des images médicales avec pertes contrôlées, l'évaluation de la qualité des images comprimées reste un problème particulièrement délicat. En effet, la compression ne doit en aucun cas induire une modification de diagnostic ni choquer l'œil du médecin.

Dans cette partie, nous avons présenté l'application de notre algorithme de reconstruction sur les images médicales que nous avons développées précédemment. Notre algorithme de décompression rapide est adapté à une modalité médicale.

En effet, de tels outils nous permettent de prouver la validité des tests. Ainsi, l'utilisation des tests sur les images médicales nous a permis d'obtenir des résultats fiables.

III.6. Conclusion

Dans ce chapitre, nous avons proposé une technique de compression d'image, après le découpage de l'image en blocs, l'algorithme de compression réalise un changement d'espace plus propice à la compression, par l'intermédiaire d'une *DCT* (Discret Cosine Transform). Une étape de quantification et seuillage (partie destructrice du procédé) est ensuite appliquée, affectant les valeurs les moins significatives visuellement égales à zéro à la fin une étape de compression sans perte est appliquée.

Les résultats que nous avons obtenus et que nous avons présentés dans ce chapitre sont assez satisfaisants du point de vue de l'amélioration des performances du codeur (*PSNR*, taux de compression *Cr*, nombre de bits par pixel *bpp*) par rapport à celui pour le (CBTC-PF ou JPEG).

Chapitre IV

Etude comparative

IV.1. Introduction

Dans ce chapitre nous allons exposer les résultats comparatifs obtenus par l'application du premier schéma de compression proposée (décimation) en utilisant en premier lieu l'MLP et en deuxième lieu RBF.

Concernant la deuxième approche proposée [12] basée sur la DCT et le nouvel encodeur sans perte. Un autre travail comparatif est effectué pour situer cette dernière par rapport à la norme JPEG et celle proposée dans l'article [13].

Finalement on présente une étude comparative globale entre toutes les méthodes précédemment évoquées.

IV.2. Méthodes proposées

IV.2.1. Méthode basée sur la décimation et les réseaux de neurones

Les tableaux ci-dessous représentent les meilleurs résultats obtenus par les réseaux de neurones (MLP, RBF) qui sont appliqués sur l'image ''Lena'' au niveau de gris et couleur. Pour comparer entre les deux méthodes citées précédemment on se base sur le taux de compression (*Cr*) et le *PSNR*.

	***Cr*=4**		***Cr*=16**		***Cr*=64**	
Mesure	MLP	RBF	MLP	RBF	MLP	RBF
MAE	3.4956	3.1110	7.1207	6.7228	12.3647	11.1859
MSE	35.4056	29.5450	135.0772	118.9380	355.0480	295.7651
NMSE	0.2018	0.1684	0.7700	0.6780	2.0238	1.6859
PSNR	32.6401	33.4260	26.8250	27.3776	22.6279	23.4213

Tableau IV.1 Comparaison entre les résultats des réseaux de neurones (MLP, RBF) sur l'image ''Lena'' au niveau de gris

Application des réseaux de neurones sur l'image ''Lena'' au niveau de couleur (RGB)

	***Cr*=4**		***Cr*=16**		***Cr*=64**	
Mesure	MLP	RBF	MLP	RBF	MLP	RBF
MAE	4.0454	3.3031	7.6116	6.5566	12.3969	10.7278
MSE	39.5744	31.1669	138.3837	112.7070	342.6685	275.8563
NMSE	0.2523	0.2113	0.8823	0.7375	2.1848	1.7648
PSNR	32.1567	33.1939	26.7200	27.6113	22.7821	23.7240

Tableau IV.2 Comparaison entre les résultats des réseaux de neurones (MLP, RBF) sur l'image ''Lena'' au niveau de couleur

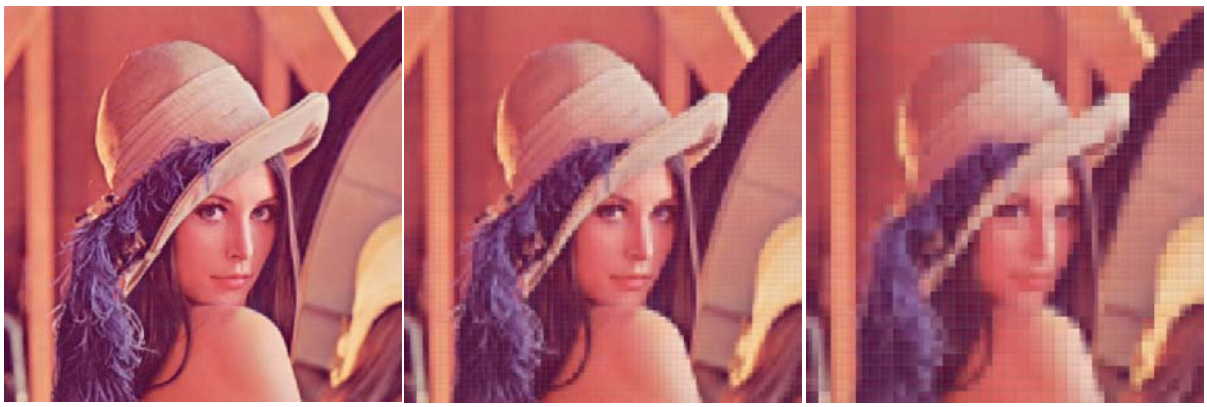
D'après les résultats obtenus, on peut dire que les meilleurs résultats sont acquis par l'utilisation des réseaux de neurones RBF avec un parcours sans recouvrement.

Les figures IV.1 et IV.2 représentent la reconstruction d'image ‘‘Lena’’ au niveau de gris et couleur par le réseau de neurones RBF.

(a) (b) (c)

Figure IV.1 Reconstruction d'image ‘‘Lena’’ au niveau de gris par les réseaux RBF avec un parcours sans recouvrement

Images reconstruites (a), (b) et (c)



(a) (b) (c)

Figure IV.2 Reconstruction d'image ‘‘Lena’’ au niveau de couleur par les réseaux RBF avec un parcours sans recouvrement

Image reconstruite (a), (b) et (c)

IV.2.2. Méthode basée sur la DCT et les nouveaux scanning adaptatif et l'encodeur sans perte

On peut comparer les méthodes essentiellement sur deux plans : d'une part l'efficacité en coût de codage qui permet d'atteindre des taux de compression importants et d'autre part la complexité d'implémentation de l'algorithme et sa rapidité d'exécution. On s'intéresse ici essentiellement au premier aspect.

Le tableau IV.3 donne Les résultats des *PSNR*, *Cr* et *bpp* obtenus par JPEG et la méthode proposée (découpage 8×8 et 32×32).

	PSNR	***Cr***	***bpp***
JPEG	24.16	59	0.14
	29.7	32	0.25
Bloc 8×8	24.15	38.9595	0.2053
	29.95	19.763	0.4048
Bloc 32×32	24.9	123.9	0.0646
	29.74	32.746	0.2443

Tableau IV.3 Résultats des *PSNR*, *Cr* et *bpp* obtenus par JPEG et la méthode proposée.

La figure IV.3 représente la reconstruction d'image ‘‘Lena’’ au niveau de gris par JPEG et la méthode proposée (découpage 32×32).

(a) bloc 32×32
Cr 123.9012
PSNR 24.9db

(b)JPEG
Cr 59
PSNR 24.16db

(c) bloc 32×32
Cr 32.746
PSNR 29.74db

(d)JPEG
Cr 32
PSNR 29.7db

Figure IV.3 Résultats d'application de JPEG et notre méthode sur l'image de ''Lena''

D'après le tableau IV.3 et la figure IV.3 on observe que la méthode proposée permet d'obtenir de meilleurs résultats que ceux de la méthode de codage JPEG. La comparaison des *PSNR* obtenus ainsi que les qualités visuelles des images compressées aux mêmes taux démontrent l'efficacité de l'algorithme proposé. Donc plus le débit est réduit, plus l'écart entre les deux méthodes est important.

Le tableau IV.4 représente l'étude comparative entre la méthode proposée et la méthode CBTC-PF qui est présentée dans l'article [13].

CBTC-PF [13]			DCT-scanning-sans perte [12]	
Image	***PSNR***	***bpp***	***PSNR***	***bpp***
Airplane	30.36	1.04	30.5854	0.7381
Peppers	30.15	1.5	30.9369	1.4685
Lena	31.93	1.17	31.9513	1.0054
Girl	35.13	0.6	35.1323	0.6914
Couple	32.44	1	32.3664	1.3588
House	31.79	1.2	31.8760	1.0879
Zelda	31.31	1.12	31.1993	1.1250
Moyenne	31.8729	1.09	32.0068	1.0679

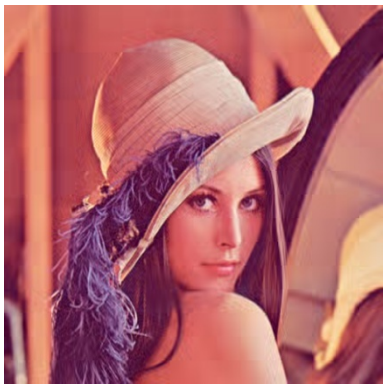
Tableau IV.4 Comparaison des résultats de la méthode proposée et ceux de la méthode CBTC-PF

Il apparaît que les résultats de notre approche sont particulièrement performants par rapport au CBTC-PF.

Les différentes images reconstruites par la méthode proposée sont illustrées par la figure suivante :

Airplane
PSNR 30.58db
bpp 0.7381

Peppers
PSNR 30.94db
bpp 1.468



Lena
PSNR 31.95db
bpp 1

Girl
PSNR 35.13db
bpp 0.6914

Couple
PSNR 32.37db
bpp 1.3588

House
PSNR 31.879db
bpp 1.0879

Zelda
PSNR 31.199db
bpp 1.125

Figure IV.4 Les images reconstruites

IV.2.3. Étude comparative globale

Le tableau IV.5 illustre bien les résultats obtenus avec les différentes images par les méthodes proposées (DCT, RBF).

Image	Méthode proposée à base de (DCT-scanning-sans perte)		Méthode proposée à base de RBF par décimation					
	PSNR	***Cr***	***PSNR***	***Cr***	***PSNR***	***Cr***	***PSNR***	***Cr***
Airplane	30.5854	32.5167	31.493	4	25.8491	16	22.3207	64
Peppers	30.9369	16.3427	32.8826	4	26.6843	16	22.4191	64
Lena	31.9513	23.8705	33.4379	4	27.7716	16	23.8329	64
Girl	35.1323	34.7142	34.1303	4	27.7701	16	23.7489	64
Couple	32.3664	17.6631	31.4988	4	26.0193	16	22.7230	64
House	31.8760	22.0610	32.1058	4	26.1484	16	22.552	64
Zelda	31.1993	21.3333	32.2661	4	26.6184	16	22.9640	64
Cerveau	30.8744	19.3081	28.5536	4	22.1778	16	18.6509	64
Moyenne	31.8653	23.4762	32.0460	4	26.1299	16	22.4014	64

Tableau IV.5 Comparaison entre les méthodes proposées à base de DCT-scanning-sans perte et les réseaux de neurones par décimation

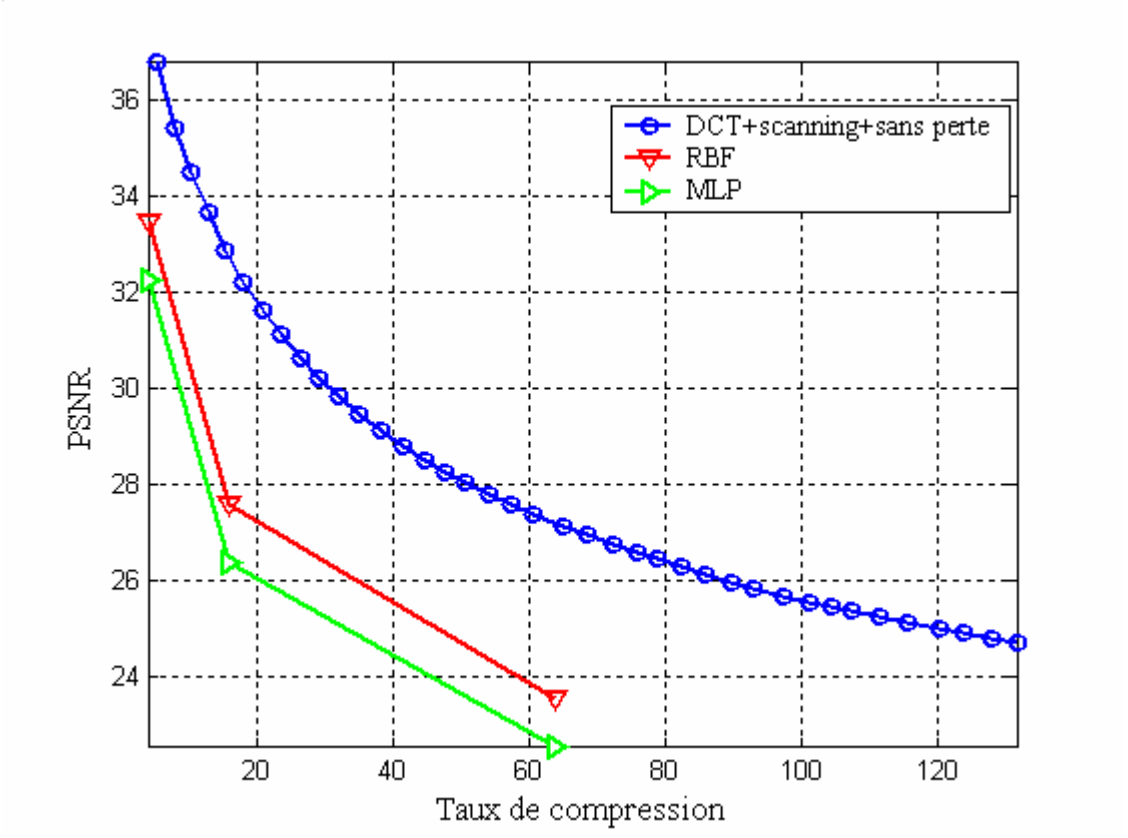


Figure IV.5 Evolution de *PSNR* en fonction de taux de compression de l'image ''Lena'' au niveau de gris

D'après les résultats présentés dans le tableau IV.5 et les courbes dans la figure IV.5 on peut dire que la méthode proposée donne un taux de compression supérieur par rapport aux MLP ou RBF.

IV.3. Conclusion

Dans ce chapitre, nous avons présenté la comparaison de deux méthodes correspondant à deux concepts différents. La première méthode proposée concerne le réseau de neurones (MLP, RBF), dans la deuxième méthode nous avons proposé une méthode de compression basée sur la DCT et un nouvel encodeur sans perte.

Pour la compression d'images fixes, la méthode proposée apparaît donc comme un outil efficace. Elle permet en effet d'obtenir une qualité de compression supérieure aux méthodes évoquées précédemment (MLP, RBF, CBTC-PF et JPEG).

Conclusion générale

Conclusion générale

Au cours de ce travail nous avons étudié une problématique liée à la compression des images. Nous avons développé des algorithmes pour faire face à ce problème. Une étude comparative avec les méthodes utilisées dans la littérature dans le domaine de la compression (JPEG, CBTC-PF) a été faite pour valider notre approche.

Notre première contribution a porté sur l'utilisation des réseaux de neurones MLP et RBF pour la reconstruction des images au niveau de gris et couleur compressées par la méthode de décimation. Cette dernière consiste à diminuer le nombre de pixels en éliminant une ligne sur deux et une colonne sur deux.

Le développement de cette approche a été fondé sur deux étapes. La première étape, de réfère à la proposition de l'architecture optimale pour le réseau de neurones (MLP et RBF), que nous avons utilisé pour la reconstruction des images compressée. Pour atteindre notre objectif, nous avons utilisé deux types de bases de données, avec et sans recouvrement et différents critères quantitatifs pour l'évaluation de la qualité de l'image reconstruite. Lors de cette étape, nous avons constaté que l'initialisation des paramètres du réseau de neurones joue un rôle très important dans la convergence de l'apprentissage.

Nous pouvons dire, que le choix de la base d'apprentissage est très important dans la conception d'un réseau de neurones.

Dans la deuxième étape, nous avons testée la robustesse et la performance des réseaux de neurones développées (MLP, RBF), afin de l'utiliser sur les images médicales.

Les résultats obtenus sur les images au niveau de gris et couleur avec le réseau de neurones RBF sont nettement meilleurs par rapport à ceux obtenus avec le réseau de neurones MLP. L'augmentation du taux de compression est suivie par une dégradation visuelle de la qualité des images reconstruite et ceci est dû essentiellement à l'effet de blocs.

Notre deuxième contribution a porté sur la création d'une technique basée sur la *DCT* ainsi que les nouveaux scannings adaptatifs et l'encodeur sans perte. Les résultats que nous avons obtenus par cette approche sont meilleurs par rapport à ceux obtenus par MLP, RBF, CBTC-PF et JPEG.

Les résultats obtenus, après l'application de l'approche sur différentes images, ont montré sa grande capacité de compression tout en préservant la qualité à un seuil très acceptable. D'avantage cette technique proposée offre la contrôlabilité d'avance de la qualité souhaité avant même l'application de la compression.

Les perspectives

Par ailleurs, de nombreuses pistes sont possibles pour améliorer et développer de nouvelles solutions dans le domaine de la compression.

- Pour la première méthode proposée (MLP, RBF) : l'application d'une autre méthode de réduction par exemple la Spline au lieu de la décimation.
- Concernant la deuxième méthode proposée basée sur la DCT et les nouveaux scanning adaptatifs et encodeur sans perte [12] : l'utilisation de cette dernière dans différents domaine tels que : Cryptage, Tatouage numérique, la compression vidéo et la compression des images médicales 3D.

Bibliographie

Bibliographie

[1] N. Benamrane, Z. B. Daho, J. Shen, ‘‘*Compression des images médicales fixes par réseau de neurones*’’, Université des Sciences et de la Technologie d’Oran, Départ. Informatique.

[2] A.W. Wong, R.K. Taira, and H.K. Huang, ‘‘*Implementation of a digital archive system for a radiology department*’’, in Proc. SPIE Conf. on Medical Imaging VI: PACS Design and Evaluation 1645, pp. 182-190, 1992.

[3] András CZIHÓ, ‘‘*Quantification Vectorielle et Compression D’image. Application à L’imagerie Médicale*’’, thèse doctorat de l’université de Rennes 1, 5 mai 1999.

[4] Robina Asraf, Muhammad Akbar, Noman Jafri, ‘‘*Statical Analysis of Difference image for Absolutely Lossless Compression of Medical Image*’’, 1-4244-0033-3/06, 2006 IEEE.

[5] S. Benierbah and M. Khamadja, ‘‘*Compression of colour images by inter-band compensated prediction*’’, IEE Proc.-Vis. Image Signal Process, Vol. 153, No. 2, April 2006.

[6] Sumathi Poobal, and G. Ravindran, ‘‘*Comparison of Compression Ability Using DCT and Fractal Technique on Different Imaging Modalities*’’, PWASET VOLUME 20, APRIL 2007 ISSN 1307-6884.

[7] AL BOVIK, ‘‘*Hanbook of image and video processing*’’, 2000 by Academic Press.

[8] Majid Rabbani, Rajan Joshi, ‘‘*An overview of the JPEG2000 still image compression standard*’’, Signal Processing: Image Communication 17 (2002) 3–48.

[9] Rafael C. Gonzalez and Richard E. Woods, ‘‘*Digital Image Processing*’’, Second Edition 2002 by Prentice-Hall, Inc.

[10] David Salomon, ‘‘*Data Compression Fourth Edition*’’, Springer-Verlag London Limited 2007.

[11] Dragotti, P.L., Poggi, G., and Ragozini, A.R.P, ‘‘*Compression of multispectral images by three-dimensional SPIHT algorithm*’’, IEEE Trans. Geosci. Remote Sens., 2000, 38, (1), pp. 416–428.

[12] F. Douak, R. Benzid and N. Benoudjit, ‘‘*Color image compression based on DCT transform combined to an adaptive block scanning*’’, submitted to Signal processing, Image communication Journal, 2008.

[13] Bibhas Chandra Dhara, Bhabatosh Chanda, ‘‘*Color image compression based on block truncation coding using pattern fitting principle*’’, Pattern Recognition 40 (2007) 2408-2417.

[14] Ahmed BEN ATITALLAH, ‘‘*Etude et Implantation d’Algorithmes de Compression D’images dans un Environnement Mixte Matériel et Logiciel*’’, thèse doctorat de l’université bordeaux 1, école doctorale des sciences physiques et de l’ingénieur, 11 Juillet 2007.

[15] M. Domariski and K. Rakowsi, ‘‘*A Simple Technique for Near-Lossless Coding of Color Images*’’, 0-7803-5482-6/99/IEEE, International Symposium on Circuits and Systems, pp. 299-302, May 28-31 2000, Geneva, Switzerland.

[16] Sylvain FASSINO, ‘‘*Agrandissement D’Images et de Séquences Vidéo*’’, thèse de Institut national polytechnique de Grenoble, 13 juillet 2004.

[17] Tinku Acharya and Ping-Sing Tsai, ‘‘*JPEG2000 Standard for Image Compression Concepts, Algorithms and VLSI Architectures*’’, Canada 2005 by John Wiley & Sons, Inc.

[18] Cédric VALADE, ‘‘*Compression D’images Complexes avec Pertes : Application à L’imagerie Radar*’’, thèse doctorat de L’école nationale supérieure des Télécommunications, 8 décembre 2006.

[19] José Marconi M. Rodrigues, ‘‘*Transfert Sécurisé D’images par Combinaison de Techniques de Compression, Cryptage et Marquage*’’, thèse doctorat de l’université Montpellier II Mention Informatique, 31 Octobre 2006.

[20] **‘‘***contribution à la restauration d’image par un modèle de réseaux de neurones*’’, ARIMA Volume 1-2002.

[21] Olivier LE CADET, ‘‘*Méthodes d’ondelettes pour la segmentation d’images. Applications à l’imagerie médicale et au tatouage d'images*’’, thèse doctorat de L’INPG de L’institut national polytechnique de Grenoble, Le 28 septembre 2004.

[22] P.Cosman, R.M.Gray, R.A.Olshen, ‘‘*Evaluating Quality of Compressed Medical Images: SNR, Subjective Rating and Diagnostic Accuracy*’’, Proc. of the IEEE, Vol. 82, pp. 919-932, June 1994.

[23] Delphine Le Guen, ‘*‘Etude de Schémas de Régulation à Double Contrainte, Débit-Qualité Locale, pour la Compression Embarquée D’images Satellitales*’’, thèse doctorat de l’université de rennes 1 Mention : Traitement du Signal et Télécommunications, 4 janvier 2001.

[24] Théodore TOTOZAFINY, ‘‘*Compression D’images Couleur pour Application a la Télésurveillance Routière par Transmission Vidéo à Très Bas Débit*’’, thèse doctorat de

l'université de Pau et des pays de L'Adour, Ecole Doctorale des Sciences Exactes et de Leurs Applications, 3 juillet 2007.

[25] Omar Hammami, ''*Etudes d'optimisation algorithmiques de JPEG2000 (EIRE)*'', ministère de l'industrie dans le cadre du Réseau National de la Recherche en Télécommunications (RNRT), février 2004.

[26] Moussa AMMAR, ''*Optimisation D'un Schéma De Codage D'image A Base D'une TCD. Application A Un Codeur JPEG Pour L'enregistrement Numérique A Bas Débit*'', thèse doctorat de l'Ecole Nationale Supérieure des télécommunications, 14 Janvier 2002.

[27] E. LE PENNEC, ''*Compression D'Image*'', Laboratoire de Probabilités et Modèles Aléatoires UMR 7599, Université Paris.

[28] Cédric Sibade, ''*Compression de Données pour les Systèmes de Traitement de Document Grand Format*'', thèse doctorat de l'Université de Marne-la-Vallée (spécialité informatique), 15 décembre 2003.

[29] G. Cazuguel, A. CZIHÓ, B. Solaiman, C. Roux, ''*Medical Image Compression and Feature Extraction using Vector Quantization, Self-Organizing Maps and Quadtree Decomposition*'', Dépt. Image et Traitement de l'Information B.P.832, 29285 Brest Cedex, France, Laboratoire de Traitement de l'Information Médicale, 1999.

[30] Richard Lepage, ''*Reconnaissance D'algues Toxiques Par Vision Artificielle Et Réseau De Neurones*'', Mémoire De Recherche de maître ès sciences appliquées de L'université De Québec à Rimouski, septembre 2004.

[31] Marc Parizeau, ''Réseaux de Neurones GIF-21140 et GIF-64326'', université de Laval, 2006.

[32] L. BAGHLI, ''*Contribution à la Commande de la Machine Asynchrone, Utilisation de la Logique Floue, des Réseaux de Neurones et des Algorithmes Génétiques*'', thèse doctorat de l'université Henri Poincaré, Nancy I, 14 Janvier 1999.

[33] Latifa Oukhellou, ''*Paramétrisation et Classification de Signaux en Contrôle Non Destructif. Application à la Reconnaissance des Défauts de Rails par Courants de Foucault*'', Thèse Doctorat en Sciences de l'Université Paris XI Orsay, 4 juillet 1997.

[34] G. Dreyfus, ''*Neural Networks Methodology and Applications*'', ESPCI, Laboratoire d'Électronique, 2005.

[35] Jean-Pierre Mano, ''*Etude de L'Emergence Fonctionnelle Au Sein D'Un Réseau De Neuro-Agents Coopératifs*'', thèse doctorat de l'université de Toulouse III, 30 mai 2006.

[36] Michael A. Arbib, ''*The Handbook of Brain Theory and Neural Networks*'', Second Edition, 2003 Massachusetts Institute of Technology.

[37] Howard Demuth and Mark Beale, ‘‘*Neural Network Toolbox User’s Guide*’’, Version 4, September 2000.

[38] N. K. Kasabov, ‘‘*Foundations of Neural Networks, Fuzzy Systems, and Knowledge Engineering*’’, MIT Press, 1998.

[39] YU HEN HU and JENQ-NENG HWANG, ‘‘*Handbook of NEURAL NETWORK SIGNAL PROCESSING*’’, CRC Press LLC, 2002.

[40] Philippe LUCIDARME, ‘‘*Apprentissage et Adaptation pour des Ensembles de Robots Réactifs Coopérants*’’, thèse doctorat de l’université Montpellier II, 7 novembre 2003.

[41] Maysoun DAMES, ‘‘*Méthodologie de Modélisation et D’optimisation D’opérations de Dispersion Liquide-Liquide en Cuve Agitée*’’, thèse doctorat de l’institut national polytechnique de Toulouse, 21 juillet 2005.

[42] Andrzej CICHOCKI and Shun-ichi AMARI, ‘‘*Adaptive Blind Signal and Image Processing Learning Algorithms and Applications*’’, John Wiley & Sons, Ltd, 2002.

[43] N. Benoudjit, E. Cools, M. Meurens, M. Verleysen, **‘‘***Chemometric calibration of infrared spectrometers: selection and validation of ariables by non-linear models*’’, Chemometrics and Intelligent Laboratory Systems 70 (2004) 47–53.

[44] Y. H.HU and J. N. HWANG, ‘‘*Handbook of Neural Network Signal Processing*’’, CRC PRESS, 2001.

[45] Patrick LAMBERT, ‘‘*Etudes méthodologiques du filtrage et de la segmentation d'images multi-composantes*’’, HABILITATION A DIRIGER DES RECHERCHES en Electronique, Electrotechnique ET Automatique, 12 juillet 2002.

[46] BENZID Redha, ‘‘*Ondelettes et Statistiques d’Ordre Supérieur Appliquées aux Signaux Uni et Bidimensionnels*’’, thèse doctorat de l’université de Batna, 15 Septembre 2005.

[47] http://sipi.usc.edu/database/.

Résumé

Ce travail vise à reconstruire des images compressées dont l'objectif est de réduire la quantité de bits nécessaires pour les décrire tout en gardant un aspect visuel acceptable des images reconstruites d'une part, et de transmettre plus rapidement l'information d'autre part. Dans ce contexte nous avons proposé deux méthodes de compression, pour des applications spécifiques (particulièrement dans le domaine médical). La première basée sur les réseaux de neurones (MLP, RBF) selon une méthode simple qui est la décimation. La deuxième repose sur une méthode de compression à base de transformée de cosinus discret (DCT), les nouveaux scannings adaptatifs et l'encodeur sans perte. Nous avons montré que les algorithmes proposés, sur lesquelles notre travail est basé, possèdent des propriétés très intéressantes pour atteindre notre objectif. Les résultats que nous avons obtenus, par la deuxième technique sont meilleurs par rapport à ceux obtenus par MLP, RBF, CBTC-PF et JPEG. De plus cette technique présente l'avantage de contrôler la qualité souhaitée avant même d'effectuer la compression.

Mots clés :

Compression d'image, Réseaux de neurones, MLP, RBF, DCT, Scannings adaptatifs, Encodeur sans perte.

Abstract

This work aims at rebuilding compressed images whose objective is to reduce the quantity of bits necessary to describe them while keeping an acceptable visual aspect of the rebuilt images on the one hand, and to transmit information more quickly on the other hand. In this context we proposed two methods of compression, for specific applications (particularly in the medical field).The first based on the networks of neurons (MLP, RBF) according to a simple method which is the decimation. Second rests on a method of compression containing of discrete cosine transform (DCT), the new adaptive scanning and the lossless coder. We showed that the algorithms proposed, on which our work is based, have very interesting properties to achieve our goal. The results which we obtained, by the second technique are better compared to those obtained by MLP, RBF, CBTC-PF and JPEG. Moreover this technique presents the advantage of controlling the quality desired before even carrying out compression.

Key words :

Image Compression, Neural Networks, MLP, RBF, DCT, Adaptive Scanning, Lossless Coding.