N° d'ordre : .....

N° de série : .....

République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Mémoire de Fin d'Étude

Présenté à

L'Université Echahid Hamma Lakhdar El Oued

Faculté de Technologie

Département de Génie Electrique

En vue de l'obtention du diplôme de

MASTER ACADEMIQUE

En Télécommunications

Présenté par

YOUMBAI Mohammed Zouheir

Compression d'images en utilisant la transformé en curvelets

Soutenu le ../05/2016. Devant le jury composé de :

Mr. AJGOU Riadh	Maitre de conférences	Président
Mr. BOUGOUS Chahra	Maitre de conférences	Examinateur
Mr. HIMA Abdelkader	Maitre de conférences	Rapporteur

Année Universitaire 2015/2016

Dedications

I sincerely dedicate my humble work to the one who have always been there for me and never quit supporting me ...my dear parents.

I also dedicated my graduation memoir to all beloved family;

So my best cousins here in El ouedsouf.

So all my friends and colleagues, most particularly, Faouzi DAHA.

Acknowledement

We would like acknowledement and thank allah for all the blessings and guidance we are granted thank to our supervisor Mr. HIMA A., who were more than generous with his expertise and precious time.

Also, we give lots of thanks to all the personnel and teachers, most Mr.CHAMSA staff of Echahid Hamma Lakhdar

Universtity for all the facilities they have provided.

Résumé

La sortie volumineuse de la diversité des images et de l'expansion utilisés dans toutes les applications de la vie et doivent être stockées et la vitesse d'envoi est essentiels indispensables; il faut créer des techniques nouvelles et efficaces pour améliorer la performance de compression d'image pour diminuer l'espace du stockage et l'amélioration des capacités de transmission de l'espace.

Selon à l'observation et examiner les résultats des études précédentes distingue un certain nombre de défauts de mentionner le plus important de la difficulté de la représentation des bords des objets, ce qui a conduit à l'émergence de distorsions dans les images après appliquer la pression de niveau haut.

Et comme un moyen de réduire ce phénomène que nous vivons une nouvelle technique basées sur des ondelettes géométriques appelées Curvelets, associée à une technique de quantification scalaire logarithmique.

Mots-Clés: Compression d'image, ondelettes géométriques, Curvelets, quantification scalaire logarithmique.

Abstract

the voluminous quantity of images that comes from their variety and widespread use in all life applications and the need to store and speed sending, that are the principal things have to invent more efficient image compression techniques to improve timeliness of data transmission and storage capacity.

Through observations of previous study's results we remark some mistakes we mention from them; the difficult representation the borders of the objects which led to appearance the distortions of images after applying high level pressure.

Lastly, we can diminish this phenomenon through repeat new technique based on geometric wavelets called Curvelets, associated with a logarithmic scalar quantization technique.

Key-words: Image compression, geometric wavelets, Curvelets, scalar logarithmic quantization.

ملخص

إن الكم الهائل من الصور الناتج من تنوع و توسع استخدامها في جميع تطبيقات الحياة و ضرورة تخزينها و سرعة إرسالها أمران أساسيان لا غنى عنهما استلزما ابتكار تقنيات جديدة و فعالة للتحسين من أداء ضغط الصور مع الحفاظ على جودتها بغية التقليل من مساحة التخزين و تحسين قدرات الإرسال.

ومن خلال ملاحظة و مراجعة نتائج دراسات سابقة نميز عدد من العيوب نذكر أهمها صعوبة تمثيل حواف الأجسام مما أدي إلى ظهور تشوهات في الصور بعد إجراء عمليات ضغط ذا مستوى عالي.

و كوسيلة للحد من هذه الظاهرة نعيد تجربة تقنية جديدة اعتمدت على مويجات هندسية تسمى الكورفلات مدعومة بعدة تقنيات ذا تكميم سلمي لوغاريتمي.

كلمات مفتاحية : ضغط الصور ، مويجات هندسية ، الكورفلات ، تكميم سلمي لوغاريتمي.

Liste des tableaux

Tableau II.1 : *Résultats du probabilité obtenus par utilisant codage arithmétique.* 27
Tableau II.2 : *Résultats du nombre nécessaire pour chaque caractère.* 27

Tableau III.1 : *Résultats obtenus par utilisant un quantificateur scalaire uniforme.* 32
Tableau III.2 : *Résultats obtenus par utilisant un quantificateur scalaire non-uniforme avec Mu-law.* 35
Tableau III.3 : *Résultats obtenus par utilisant un quantificateur scalaire uniforme optimisé par Lloyd-Max.* 38
Tableau III.4 : *Résultats obtenus par utilisant un quantificateur scalaire non-uniforme avec Mu-law et optimisé par Lloyd-Max.* 41

Liste des figures

Figure I.1 : *Schéma générale de la compression d'image.* 4
Figure I.2 : *Comparaison de l'approximation des ondelettes et des curvelets.* 8
Figure I.3 : *Schéma de construction de la transformée en Curvelets d'une image.* 9

Figure II.1 : *Quantificateur Midtread et Quantificateur Midrise.* 17
Figure II.2 : *Quantificateur uniforme à zone morte.* 18
Figure II.3 : *Quantification scalaire non uniforme de Lloyd-MAX.* 19
Figure II.4 : *La quantification scalaire non uniforme par compression (companding).* 20
Figure II.5 : *Exemples des fonctions de compression et d'expansion.* 21
Figure II.6 : *Fonction companding.* 22
Figure II.7 : *Fonction de compression Mu-law (μ =1, 10, 255).* 23
Figure II.8 : *Etapes de quantification par Companding et quantification μ-law à 8 niveaux. (μ=25)* 24

Figure III.1 : Algorithme de compression et décompression. 31
Figure III.2 : *Evolution du PSNR en utilisantion un quantificateur scalaire uniforme pour différentes images de test.* 33
Figure III.3 : *Evolution du Taux de compression en utilisantion un quantificateur scalaire uniforme pour différentes images de test.* 33
Figure III.4 :*Résultats obtenus par utilisant un quantificateur scalaire uniforme.* 34
Figure III.5 : *Evolution du PSNR en utilisantion un quantificateur scalaire non-uniforme avec Mu-law pour différentes images de test.* 36
Figure III.6 : *Evolution du taux de compression en utilisantion un quantificateur scalaire non-uniforme avec Mu-law pour différentes images de test.* 36
Figure III.7 : *Résultats obtenu par utilisant un quantificateur scalaire non-uniforme Mu law.* 37
Figure III.8 : *Evolution du PSNR en utilisantion un quantificateur scalaire uniforme optimisé par Lloyd-Max pour différentes images de test.* 39
Figure III.9 : *Evolution du Taux de compression en utilisantion un quantificateur scalaire uniforme optimisé par Lloyd-Max pour différentes images de test.* 39

Figure III.10 : *Résultats obtenu par utilisant un quantificateur scalaire uniforme optimisé par Lloyd-Max.* …… 40
Figure III.11 : *Evolution du PSNR en utilisantion un quantificateur scalaire non-uniforme avec Mu-law et optimisé par Lloyd-Max pour différentes images de test.* …… 42
Figure III.12 : *Evolution du Taux de compression en utilisantion un quantification scalaire non-uniforme avec Mu-law et optimisé par Lloyd-Max pour différentes images de test.*…… 42
Figure III.13 : *Résultats obtenu e par utilisant un scalaire non-uniforme avec Mu-law et optimisation par Lloyd max.*…… 43

Liste d'abréviations

DFT	Transformée de fourier.
DCT	Transformée en cosinus discrète.
DST	Transformée en sinus discrète.
DQS	Distorsion du Quantification Scalaire.
EQM	Eerreur quadratique moyenne .
FFT	Transformée de Fourier rapide.
FFT2D	Transformée de Fourier rapide à deux dimensions.
JPEG	Joint Photographic Expert Group.
JPEG2000	Standard de compression d'images fixes récent, introduit par JPEG.
KL	karhunen-Loeve.
PDF	Fonction de distribution de la probabilité.
PSNR	Rapport signal sur bruit crête.
Q(x)	La fonction de quantification.
QS	La Quantification Scalaire.
Q_{MR}	Quantificateur scalaire midtrise.
Q_{MT}	Quantificateur scalaire midtread.
Tc	Taux de compression.
TO	transformation par ondelettes.
BI	Borne inférieure**.**
BS	La borne supérieure.
USFFT	Unequally spaced FFT.

Sommaire

Introduction générale **1**

CHAPITRE I : Transformation en curvelets

I.1. Introduction **3**

I.2. Images numériques et compressions **3**

I.3. Principe générale de la compression des images **4**

I.4. Mesure les performances en compression **6**

I.4.1. La qualité de reconstitution de l'image (débit-distorsion et critères de qualité) 6

I.4.2. Le taux de compression 6

I.4.3. La rapidité du codeur et décodeur (complexité) 6

I.5. Classification des méthodes de compression **6**

I.5.1. Compression avec perte 7

I.5.2. Compression sans perte 7

I.6. Transformée en ondelettes **7**

I.7. Géométrie **8**

I.8. Curvelet , Notion **8**

I.9. Curvelets première génération **10**

I.10. Curvelets rapides seconde génération **10**

I.11. Conclusion **13**

II.1. Introduction **15**

CHAPITR II: Quantification et codage

II.2. Quantification scalaire **15**

II.2.1. Quantification scalaire uniforme 16

II.2.2. Quantificateur uniforme a zone morte 17

II.2.3. Quantificateurscalaire non uniforme 18

II.2.4. Algorithme de Lloyd-MAX 18

II.2.5. Quantificateur scalaire non uniforme par compression (companding) 20

II.2.6. Quantificateur scalaire non uniforme par Mu-law companding 22

II.3. Quantification et l'entropie **24**

II.4. Codage de l'information **26**

II.4.1. Codeur de Huffman 26

II.4.2. Codage arithmétique 26

II.5. Conclusion **28**

CHAPITRE III: Résultats et discussion

III.1. Introduction **30**

III.2. Algorithme de compression **31**

III.3. Présentation et discussions **31**

III.3.1. Compression en utilisation un quantificateur scalaire uniforme 32

..... 34

III.3.2. Compression en utilisation un quantificateur scalaire non-uniforme avec Mu-law companding 35

III.3.3. Compression en utilisation un quantificateur scalaire uniforme avec optimisation des partitions par Lloyd-Max 38

III.3.4. Compression en utilisation un quantificateur scalaire non-uniforme avec Mu-law et optimisation des partitions par l'algorithme de Lloyd-Max 41

III.5. Conclusion **44**

Conclusion générale **45**

Bibliographie **46**

Introduction générale

L'analyse d'images est une discipline qui a pour objectif de développer des méthodes de calcul automatisé, afin de les utiliser dans différents domaines : télécommunications et du multimédia, médical, géographie, vision par ordinateur…etc.

Avec le développement de l'outil informatique, plusieurs techniques de traitement des images ont vu le jour, alors on est appelé à s'intéresser à compression d'image de façon particulière.

En effet, cette dernière nous permet de réduire la taille d'une image dans le but d'augmenter la capacité des supports de stockage (limitée en capacité) en gardant le maximum de qualité

Dans la littérature, nous trouvons plus d'une vingtaine de formats de compression, spécifiquement dans la compression d'image (gif, .jpeg, .bmp...), ayant chacun leur propre méthode de transformation, ou cumulant plusieurs algorithmes, mais tous sont complémentaires. Parmi lesquelles, nous citons : la transformée en ondelettes qui est à la base des standards de compression.

Depuis quelque année, des nouvelles transformations par ondelettes ont été développées comme les Curvelets, Contourlets et Bandlets qui intègrent de notion de directionnalité et qui permettent de rechercher des objets de manière optimale.

L'objectif de ce mémoire est de présenter avec détails la transformation avec Curvelets pour améliorer les performances de la compression d'images fixes (PSNR) par rapport aux performances des ondelettes classiques. Le travail effectué est consacré à la phase de quantification qui représente l'étape la plus importante dans l'algorithme de compression d'image.

Nous avons partagé notre travail en trois chapitres. Le premier chapitre donnera quelques notions de base utilisées communément en compression d'image, et sera consacré à la transformation en Curvelets, aux propriétés principales et aux outils des Curvelets qui amélioraient les ondelettes classiques. Ensuite, on passera à la présentation des techniques de quantification et codage. Ça fera alors l'objet du chapitre II. Enfin, on finalisera notre mémoire par présenter la méthode de compression utilisée, basée sur une transformation en Curvelets.

Une conclusion générale clôturera ce mémoire.

CHAPITRE I :

Transformation en Curvelets

I.1. Introduction

Avec le développement de l'outil informatique, on effectue des échanges de volumes importants d'information. Or la gestion d'une telle masse pose des problèmes de stockage et de transfert. Pour cela, des études ont été menées afin de mettre en évidence des algorithmes de compression et de décompression de données. Leur but est de changer le format des informations de telle sorte qu'elles occupent moins de volume.

Au cours des dernières décennies, le transformée en ondelettes ont eu un immense succès dans le domaine du traitement d'images, et ont été utilisées pour de nombreux tels que la compression et la restauration d'images. Ces problèmes ont souvent pour préalable la recherche d'une représentation de l'image qui soit la plus parcimonieuse possible, au sens ou un petit nombre de paramètres permet d'obtenir une approximation précise de l'images.

Depuis quelque année, des nouvelles transformations ont été développées comme les curvelets, contourlets et bandelets qui intègrent de notion de directionnalité et qui permettent de chercher des objets de manière optimale.

Dans ce chapitre présente les concepts généraux de compression d'images fixes, les étapes principales de l'algorithme de compression avec pertes et le principe de la transformée en Curvelets et sa mise en œuvre numérique, ainsi que ses principales propriétés, qui leur permettent de représenter les régularités géométriques de façon plus efficace.

I.2. Images numériques et compressions

Une image numérique, telle qu'on peut la voir sur un écran d'ordinateur, est une mosaïque de pixels (Picture éléments) dont la couleur est choisie dans un ensemble fini : il s'agit d'un objet naturellement discret. Il s'identifie à une matrice a h colonnes et v lignes dont les éléments appartiennent à un ensemble fini E. Typiquement, pour une image en niveau de gris, E est constitué des entiers compris entre 0 et 255 et correspond à l'intensité lumineuse de chaque pixel. Ces 256 valeurs distinctes se codent avec 8 bits (2^8=256), d'où le nom d'image 8 bits. Pour les images couleurs, chaque pixel est caractérisé par 3 intensités lumineuses, celles des canaux rouge, vert et bleu, définissant des images 3×8=24 bits.

Le stockage en mémoire d'une image couleur requiert donc $h \times v \times 24$ bits. Ceci représente rapidement une grande quantité : une image de 4 mégapixels nécessite ainsi $4 \times 2^{20} \times 24$ bits soit 12 mégaoctets, ne permettant le stockage que d'une dizaine de photos sur une carte 128 mégaoctets. Les modems téléphoniques fournissaient des contraintes encore

plus grandes puisque qu'une image couleur de 320× 240 non compressée ne nécessitait pas moins de 4 minutes pour être transmise.

L'objectif de la compression est de réduire la quantité de mémoire nécessaire pour le stockage d'une image et de réduire le temps de transmission de celle-ci [1], ou de manière équivalente trouver la représentation la plus économique (plus petite) possible en gardant le maximum de qualité dans l'image représentée. Pour quartier la quantité de compression effectuée sur une image numérique, on définit son taux de compression comme le rapport entre le nombre de bits pour représenter l'image compressé et le nombre de bits pour représenter l'image initiale. Avec cette définition, plus le taux de compression est faible, meilleure est la compression. S'il est possible de revenir à une reconstruction exacte de l'image de départ, on dit qu'on a affaire à un algorithme de compression sans perte. Si au contraire l'image reconstruite n'est pas exactement la même que l'image initiale, on dit qu'on fait de la compression avec perte.

I.3. Principe générale de la compression des images

Tout système de compression peut se décomposer en trois modules distincts : la dé-corrélation des données images source, la quantification des valeurs dé-corrélées et l'affectation de codes binaires [2]. Le schéma fonctionnel de la compression est présenté dans la figure1.1 ci-dessous :

Figure I.1 : ***Schéma générale de la compression d'image*** **[23].**

- **Le module de dé-corrélation (Transformation)** de l'image réduit la redondance contenue dans les données ou encore, permet de dissocier la redondance de l'information pertinente. Dans une image naturelle, la valeur d'un pixel est fortement corrélée à celle deses voisins. Cette étape de traitement, parfaitement réversible, produisant des coefficients dé-corrélés et concentrant l'énergie sur peu de coefficients, permet d'optimiser les étapes ultérieures de quantification et de codage.
- **Le quantificateur** est l'organe essentiel du système decompression. Il permet de diminuer effectivement la quantité d'information transmise en éliminant le bruit issu du capteur ettoute l'information non pertinente (dans le meilleur des cas!) vis à vis de l'utilisation qui est faite des images après compression et décompression. Cette étape est la seule qui introduise une non-réversibilité dans l'algorithmede compression. Elle peut jouer le rôle d'organe de commande du système de compression car elle détermine la quantité d'information à transmettre et ainsi le taux de compression.Deux types de quantification peuvent être appliqués: la quantification scalaire et la quantification vectorielle. Cette dernière est beaucoup plus efficace mais plus difficile à mettreen œuvre et de fait beaucoup moins utilisée dans la pratique. En outre la quantification peut être uniforme ou adaptée spécifiquement à la statistique des données à quantifier: on parle alors de quantification non uniforme
- **L'affectation de codes (ou codage)** constitue la dernière étape de la chaîne de compression. Elle a pour rôle de produire le train binaire, représentatif des valeurs quantifiées, qui sera effectivement transmis ou stocké pour une transmission ultérieure. Son rôle est d'affecter à chaque valeur quantifiée un code binaire qui pourra être déchiffré par le décodeur. Cette étape peut être considérée comme une technique de compression à part entière. Les codes les plus efficaces (si une quantification uniforme a été appliquée) sont les codes à longueur variable dont le principe de réalisation est simple : il s'agit d'affecter les codes les plus courts aux valeurs les plus probables. Le défaut des codes à longueur variable réside dans la création en sortie du compresseur d'un débit dynamiquement variable, en fonction du contenu informationnel local de l'image. L'exigence d'un débit fixe peut alors être satisfaite moyennant l'adjonction d'un système d'allocation dynamique de débit, qui vient adapter la quantification à la complexité locale: on quantifie plus sévèrement les zones complexes que les zones quasi-uniformes. Si par contre le quantificateur a été adapté à la statistique des données (quantification non uniforme), un codage à longueur fixe peut être utilisé.

I.4. Mesure les performances en compression

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

I.4.1. La qualité de reconstitution de l'image (débit-distorsion et critères de qualité)

La distorsion est une mesure de l'erreur commise entre l'image originale et l'image reconstruite. Pour les mesures de distorsion, on utilisera l'erreur quadratique moyenne *EQM* [1] entre l'image originale I_0 et l'image compressée $I_c$de taille $M \times N$:

$$EQM(I_0, I_c) = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}(I_0(i,j) - I_c(i,j))^2 \quad \text{(EQ I.1)}$$

Pour les mesures de qualité, on utilisera le PSNR (Peak Signal to Noise Ratio) [1] en décibels défini à partir de l'EQM par :

$$PSNR(I_0, I_c) = 20log_{10}\left(\frac{2^r - 1)}{\sqrt{EQM(I_0, I_c)}}\right) \quad \text{(EQ I.2)}$$

Où r est la résolution numérique de l'image.

I.4.2. Le taux de compression

Le taux de compression Tc est le rapport entre la taille d'image compressée et la taille d'image originale en bits.

I.4.3. La rapidité du codeur et décodeur (complexité)

La complexité calculatoire peut être mesurée par le temps d'exécution du processus de compression ou en nombre d'opérations par pixel : c'est le nombre moyen d'opérations qui sont nécessaires à la compression de l'image.

I.5. Classification des méthodes de compression

La plupart des méthodes de compression visent à enlever la redondance présente dans l'image de manière à diminuer le nombre de bits nécessaires à sa représentation [3].

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

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

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

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

I.5.1. Compression avec perte

La suite de bits obtenue après la compression est différente de l'originale, mais l'information reste sensiblement la même. Elle est réservée aux données «perceptibles», en général sonores ou visuelles, qui peuvent subir une modification, parfois importante, sans que cela ne soit perceptible par un humain [4]. La perte d'information est irréversible, il est impossible de retrouver les données d'origine après une telle compression. Ce type de compression est utilisée en : schémas, les photos, les dessins techniques, les icônes, les bandes dessinées, les fichiers exécutables ou texte, le fax car les images compressées présentent des défauts de compression.

I.5.2. Compression sans perte

La suite de bits obtenue après la compression est strictement identique à l'originale. Il n'y a aucune perte dans l'information d'origine, l'information est seulement récrite d'une manière plus concise [5]. Les algorithmes utilisés ne garantissent pas que tous les ensembles de données en entrée soient réduits : en d'autres termes, il y a des données en entrée qui restent inchangées. Ce type de compression est utilisée en : Images médicales, les archives car il faut préserver une grande précision.

I.6. Transformée en ondelettes

L'intérêt que présentent de nombreuses transformées pour comprimer l'information est de projeter le signal sur une base de fonctions orthogonales, c'est-à- dire de distribuer l'énergie de ce signal sur des composantes de-corrélées entre elles [6]. Il existe de nombreuses transformes orthogonales dont les propriétés diffèrent.

On peut citer la transformée de fourier (DFT), la transformée en Cosinus (DCT), en sinus (DST) et celle de karhunen-Loeve (KL) qui sont les plus utilisées, ainsi que les transformées de Haar et de Hadamard.

La première transformation par ondelettes (TO) est une technique inventée par Y. Meyer [7] est un outil largement utilisé en traitement du signal et d'image. L'une de ses applications principales est la compression d'images du fait sa capacité à compacter l'énergie sur un petit nombre de coefficients permettant un codage efficace de l'image. Elle est bien localise en fréquence et en temps et peut être obtenue par un algorithme rapide.

I.7. Géométrie

La géométrie est une des caractéristiques essentielles des images : elle constitue un élément de régularité qui n'est pas pris en compte par les bases classiques. L'exploitation de cette régularité géométrique est ainsi une direction prometteuse pour la compression d'image et plus généralement pour le traitement des images.

Utilisation la géométrie de l'image permet de construire de bases d'ondelettes adaptées aux contours de l'image (curvelets, ridgelets, bandelettes, contourlets, ondelettes ENO, wedgelets, etc...) pour améliorer encore la compression.

I.8. Curvelet , Notion

Les curvelets ont été proposées par E. Candès et Donoho [8], constituent une nouvelle famille de frames d'ondelette géométrique (des ondelettes de seconde génération) plus efficaces que les transformées traditionnelles, et qui sont conçus pour représenter de façon parcimonieuse les contours. Par exemple, sur la figure I.2 (a), les ondelettes prendrait beaucoup de coefficients pour représente précisément un tel contour. Comparées aux ondelettes, les curvelets peuvent représenter un contour lisse avec moins de coefficients pour la même précision figure I.2 (b) [9].

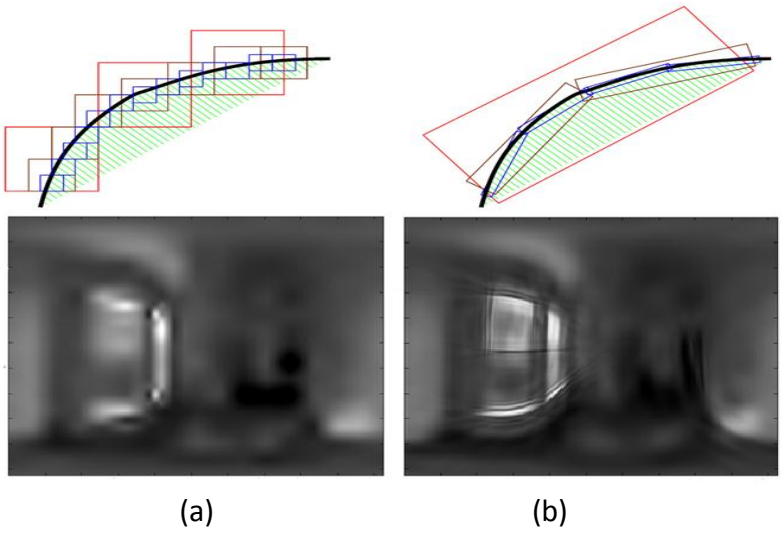


Figure I.2 : ***Comparaison de l'approximation des ondelettes et des curvelets*** **[9].**

La transformation en curvelets est obtenue en deux grandes étapes [10]. Tout d'abord on partitionne l'image en carrés de tailles variables avec recouvrement pour éviter les effets de bord. Ces carrés sont obtenus grâce à une fenêtre de Fourrier à support fini. Au sein de ces carrés on applique une transformée en Ridgelets discrète avec une dilatation de la fonction d'onde de $1/a^2$. Les contours non capturés par l'analyse en ondelettes séparables se retrouvent dans les sous-bandes de détails. Un partitionnement suffisamment fin des sous-bandes permet alors d'obtenir des blocs où ces contours forment des lignes droites et sont donc adaptés à l'analyse en Ridgelets. La transformée en Curvelets est inversible mais redondante car l'analyse en Ridgelets discrète sous-jacente est réalisée au moyen d'une FFT2D du plan polaire, nécessitant plus de points que ceux disponibles dans la grille rectangulaire. Le choix d'utilisation de la FFT provient essentiellement du théorème de la projection de Fourrier (Fourier Slice Theorem). En effet, celui-ci indique que la transformée de Radon peut être obtenue en appliquant une transformée de Fourier inverse 1-D le long des lignes radiales passant par l'origine dans le domaine de Fourier 2-D de l'image. La figure I.3 résume les étapes d'une transformée en curvelets.

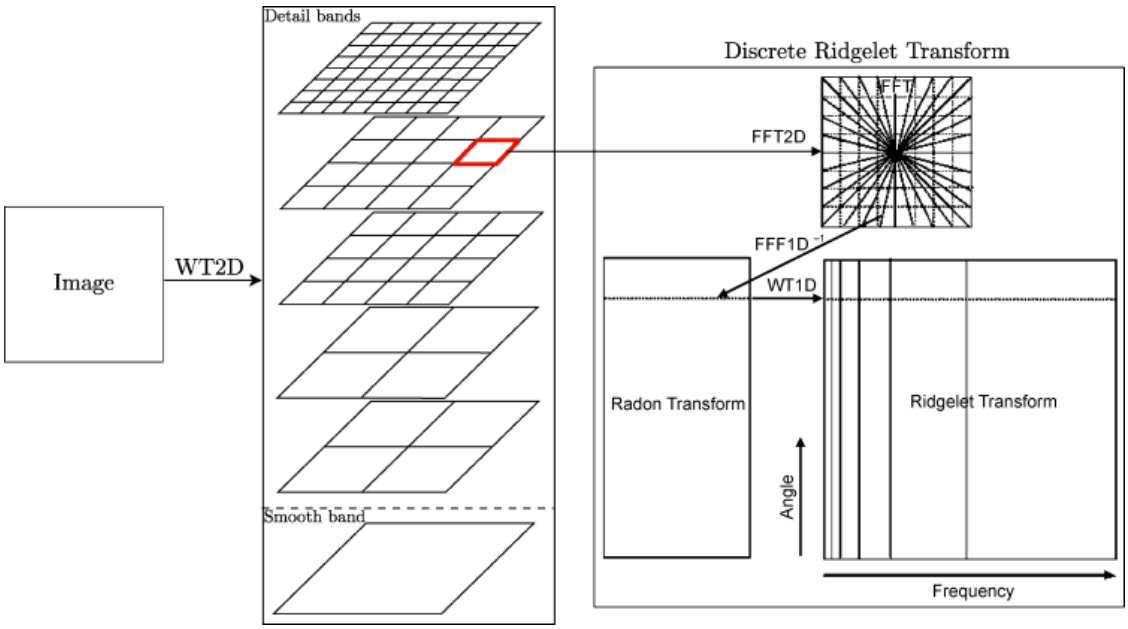


Figure I.3 : ***Schéma de construction de la transformée en Curvelets d'une image*** **[24].**

I.9. Curvelets première génération

La transformée en curvelet – 99 (première génération) [11] se dérive des ridgelets multi-échelles. Une première décomposition permet d'obtenir une analyse multi-échelles en sous-bandes centrées sur les couronnes de fréquences $f \in (0.1\ /2^{2s})/(0.1\ /2^{2s+2})$ ou $s \in n$ ou $s\ \in N$ représente l'échelle. Notons que cette découpe de l'espace fréquentiel n'est pas classique. Ces sous-bandes sont en suites analysées par une transformée en ortho-ridgelets sur des blocs de taille $2^s \times 2^s$. Les blocs d'analyse sont alors des éléments rectilignes de taille $2^s \times 2^{2s}$. Ces éléments suivent donc une loi de changement d'échelle parabolique $l = w2$. L'ondelette utilisée pour construire les ortho-ridgelets est l'ondelette de Meyer tandis que la fonction d'échelle de Lemarié est utilisée pour la représentation des basses fréquences. Une construction différente et plus générale, reposant sur la théorie des frames. La frame d'analyse est construite directement à partir d'une fonction mère C bidimensionnelle de haute fréquence selon l'un des axes et de basse fréquence selon l'autre (typiquement le produit tensoriel d'une fonction d'ondelette et d'une fonction d'échelle). La famille de curvelets $\left(C_{l,n,\theta(t)}\right) I \in n\ , n \in Z, \theta \in 2\pi$ correspondante est alors donnée par :

$$C_{l,n,\theta(t)} = 2^{31/2} C(D_l R_\theta t - n) \qquad \text{(EQ I.3)}$$

$$D_l = \begin{pmatrix} 2^{2l} & 0 \\ 0 & 2^l \end{pmatrix} \qquad \text{(EQ I.4)}$$

Où D_l est une matrice de sous-échantillonnage du changement d'échelle.

Notons au passage que $23l$ / 2 correspond à la racine carrée de son déterminant, $R\theta$ est la matrice de rotation d'angle q et n indique la position de la curvelet. Cette transformée a été utilisée avec succès dans le cadre du débruitage .

I.10. Curvelets rapides seconde génération

Les curvelets-99 se sont montrées très performantes en débruitage comme dans d'autres applications, mais souffrent d'un certain nombre de défauts. Tout d'abord, leur construction par étapes successives est complexe, et elles possèdent un trop grand nombre de paramètres (un d'angle, deux d'échelle, et trois de position,). De plus, elles sont très redondantes (de l'ordre de 16 fois le nombre d'échelles, un facteur 4 des ridgelets, et un autre à cause du recouvrement des blocs du partitionnement spatial), et l'utilisation de ridgelets à toutes les échelles crée des atomes qui obéissent à toute sorte de loi d'échelle.

Ces revers ont conduit à une construction d'un autre type de curvelets, ayant seulement trois paramètres, et avec une redondance plus faible. Cette nouvelle transformée, introduite par P. Loh et M. E. J. Candes [12] , s'implémente comme un pavage de l'espace de Fourier, considéré comme [-1; 1]^2. On définit tout d'abord la transformée continue [13] à l'aide des fenêtres V et W, respectivement radiale et angulaire, qui agiront dans le domaine de Fourier :

$$W(r), e \in \left[\frac{1}{2,2}\right] etV(t), t \in [-1,1] \quad \text{(EQ I.5)}$$

Ces fenêtres satisfont les conditions d'admissibilité suivantes, et forment ainsi une partition de l'espace de Fourier :

$$\sum_{j\in Z} W^2\left(2^j r\right) = 1\,,\ r \in R_+^*. \quad \text{(EQ I.6)}$$

$$\sum_{l\in Z} V^2\,(t-2l) = 1,\ t \in R. \quad \text{(EQ I.7)}$$

Pour toute échelle $j \geq j_0$, on définit une fenêtre fréquentielle U_j par :

$$U_j(r,\theta) = 2^{-\frac{3}{4}} W(2^{-j} r) V(2^{\frac{j}{2}} \frac{\theta}{2\pi}) \quad \text{(EQ I.8)}$$

On peut ainsi définir la curvelet à l'échelle j dans le domaine de Fourier, $\hat{\psi}_j(w) = U_j(w)$, les autres pouvant en être déduites par rotations et translations. La curvelet à l'échelle 2^{-j} d'orientation $\theta_l = 2\pi 2^{-\lceil \frac{j}{2} \rceil} l, l = 0 \ldots 2^{\lceil \frac{j}{2} \rceil} - 1$ et de position $x_k^{(j,l)} = R_{\theta l}^{-1}(k_2 2^{-\frac{j}{2}})$ est donc definie par :

$$\psi_{j,l,k}(x) = \psi_j(R_{\theta l}\left(x - x_k^{(j,l)}\right)) \quad \text{(EQ I.9)}$$

Avec :

$$R_\theta = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} \quad \text{(EQ I.10)}$$

La décomposition en curvelets est maintenant obtenue par simple produit scalaire sur cette famille de fonctions : $c(j,l,k) = \langle f | \psi_{j,l,k} \rangle$qui peut être défini dans le domaine fréquentiel par le théorème de Parseval :

$$c(j,l,k) = \frac{1}{(2\pi)^2} \int_{[-1,1]^2} \hat{f}\,(w) U_j(R_{\theta l} w) e^{l\langle x_k^{(j,l)}, w\rangle} dw \quad \text{(EQ I.11)}$$

On définit ensuite la curvelet à l'échelle grossière en introduisant un filtre passe-bas W0 vérifiant la partition uniforme de l'unité

$$|W_0(r)|^2 + \sum_{j\geq 0} \left|W_0(2^{-j} r)\right|^2 = 1 \quad \text{(EQ I.12)}$$

On construit donc les curvelets d'échelle à partir de la curvelet père

$$\hat{\phi}_{j0}(w) = 2^{-j0} W_0(2^{-j0}\sqrt{w_1^2 + w_2^2} \quad \text{(EQ I.13)}$$

$$\phi_{j0}, k(x) = \phi_{j0}(x - 2^{-j0}k) \quad \text{(EQ I.14)}$$

Propriétés

- ✓ Les curvelets ainsi définies forment une trame ajustée, c'est à dire que toute fonction $f \epsilon L^2(R^2)$se décompose selon $f = \sum\langle f, \psi_{j,l,k}\rangle \psi_{j,l,k}$ et vérifie la conservation d'énergie (identité de Parseval) lorsque l'on inclue les éléments des ondelettes pères.
- ✓ A l'échelle 2^{-j} , les curvelets $\psi_j$sont à décroissance rapide en dehors d'un rectangle de longueur $2^{-j/2}$ et largeur 2^{-j} .
- ✓ Elles ont un nombre infini de moments nuls, par leur support éloigné de l'origine en Fourier.

La transformée discrète

Pour le passage en discret [11], il faut changer quelque peu la forme des fenêtres sur lesquelles vivent les curvelets pour les adapter à une grille cartésienne. Ainsi, le pavage du plan fréquentiel en anneaux circulaires concentriques (filtres passe-bande) se transforme en couronnes cartésiennes concentriques lorsque l'on applique les filtres sur les deux directions $w_1$et w_2 indépendamment.

Soit ϕ une fonction à support dans [-2,2] et valant 1 sur [-1/2,1/2].

On pose

$$\phi_j(w) = \phi(2^{-j}w_1)\phi(2^{-j}w2). \quad \text{(EQ I.15)}$$

On définit la famille de filtres passe-bande

$$W_j^D(w) = \sqrt{\Phi_{j+1}^2 - \Phi_j^2(w)} \ , j \geq 0 \quad \text{(EQ I.16)}$$

On vérifie

$$\Phi_0^2(w) + \sum_{j\geq 0} {W_j^D}^2(w) = 1 \quad \text{(EQ I.17)}$$

On peut conserver la même fonction V qu'en continu, à support dans [-1; 1], vérifiant la condition d'admissibilité et définir

$$V_j(w) = V(\frac{2^{\lceil \frac{j}{2} \rceil} w_2}{w_1}) \quad \text{(EQ I.18)}$$

Et la fenêtre fréquentielle cartésienne associée

$$U_j^D(w) = W_j^D(w) V_j(w) \quad \text{(EQ I.19)}$$

U_j^DIsole donc les fréquences au voisinage du polygone (portion d'un secteur angulaire)

$$\{(w_1, w_2 : 2^j \leq w_1 \leq 2^{j+1}, -2^{\frac{j}{2}} \leq w_2/w_1 \leq 2^{j/2}\}$$

Pour paver tout l'espace, il reste à décaler cette fonction non pas à angles équi-espacés comme dans le domaine continu, mais à pentes équi-espacées, ce qui revient à remplacer la rotation R_θ par un cisaillement

$$S_\theta = \begin{pmatrix} 1 & 0 \\ -tan\theta & 1 \end{pmatrix} \quad \text{(EQ I.20)}$$

Pour des angles compris entre $-\pi/4$ et $\pi/4$.Les pentes à parcourir sont donc dans

[-1; 1[, à savoir$tan\theta_l = l.2^{-\lceil j/2 \rceil}$, $l \in [-2^{\lceil \frac{j}{2} \rceil}, 2^{\lceil \frac{j}{2} \rceil} - 1]$.

Finalement

$$U_{j,l}^D(w) = W_j^D(w) V_j S_{\theta l}(w)) \quad \text{(EQ I.21)}$$

On peut faire de même pour le reste de l'espace de Fourier, en appliquant des rotations de $\pi/2$ radians.

E. Candès, L. Demanet, D. Donoho, and L. Ying [14]. ont développé deux méthodes pour calculer les coefficients à partir de la grille pseudo-polaire décrite ci-dessus, différant dans la manière dont les curvelets sont transposées à une échelle et un angle donnés. La transformée peut donc se calculer par transformée de Fourier rapide non équi-espacée (USFFT) ou par repliement circulaire (wrapping) dans le domaine fréquentiel.

I.11. Conclusion

Après avoir vue des notions sur les curvelets et certaines de ses propriétés, la raison derrière leur supériorité dans la représentation des régularités géométriques par rapport aux ondelettes classiques. Nous passons au développement d'un de module de compression d'image. Il s'agit de quantification et codage.

CHAPITRE II :

Quantification et codage

II.1. Introduction

Le but de la deuxième étape du schéma bloque de la compression (cf Figure I.1), l'étape de quantification, est de diminuer la précision du stockage des entiers de la matrice $X \in R$ pour diminuer le nombre de bits occupés par chaque entier. C'est la seule partie non-conservative du bloc. Puisque les informations de basses fréquences sont plus pertinentes que les informations de hautes fréquences, la diminution de précision doit être plus forte dans les hautes fréquences. La perte de précision va donc être de plus en plus grande lorsqu'on s'éloigne de la position (0,0). Pour cela on utilise une matrice de quantification contenant des entiers par lesquels seront divisées les valeurs de la matrice X. L'information quantifiée doit alors pouvoir être décrite par un nombre de symboles suffisamment petit pour atteindre de bonnes performances en débit, mais aussi suffisamment grand pour ne pas trop détériorer l'information originale.

II.2. Quantification scalaire

Puisqu'une quantification scalaire de X^N ver $\tilde{X}^N$ peut être vue comme N opérations séparées, étudier un quantificateur scalaire de dimension N revient à étudier N quantificateurs unidimensionnels. Dans la suite de cette sous-section, on considère donc un quantificateur scalaire QS avec $N = 1$ et la variable aléatoire X a réalisations dans X.

Si $X = [a, b]$ (avec éventuellement $= -\infty, \mathrm{b} = +\infty$) l'opération quantification revient partitionner l'intervalle $[a, b]$ en K intervalles , $]P_{k-1}, P_k[$, $1 \leq k \leq K$ puis associer à chaque valeur x un point de quantification x_k tel que

$$\forall\, x \in]P_{k-1}, P_k]\,,\ Q(x) = x_k \qquad \text{(EQ II.1)}$$

La longueur de chaque intervalle de quantification $]P_{k-1}, P_k]$ est appelée pas de quantification.

Alors : $\Delta k = P_k - P_{k-1}$

II.2.1. Quantification scalaire uniforme

La Quantification uniforme est la technique de quantification scalaire la plus simple, elle consiste à diviser la dynamique du signal original en un ensemble d'intervalles identiques $K^k = [P_{k-1}, P_k]$ et des niveaux de reconstruction espacés régulièrement.

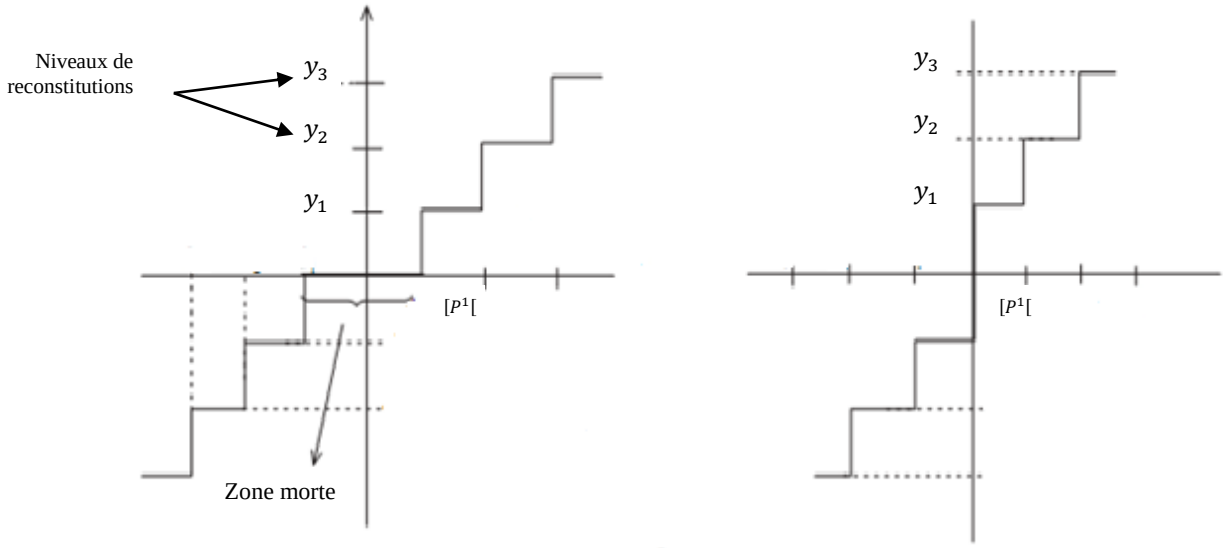
Lorsque le pas de quantification est plus grand, nous obtenons des valeurs beaucoup plus redondantes dans le signal de sorite, et par conséquent une capacité de compression plus importante, mais contre une précision plus faible lors de la reconstitution à cause de la distorsion introduite. En revanche, on peut obtenir moins de distorsions de quantification (meilleure précision) en utilisant un pas de quantification plus petite, mais ça sera au détriment de la compression.

Définir un quantificateur scalaire avec une résolution de n bits par échantillon consiste à réaliser trois opérations :

- une partition de l'intervalle $[-A, +A]$ en $L = 2^k$ intervalles distincts $\{K^1, K^2, \dots, K^L\}$ de longueur Δ.
- une numérotation des intervalles $\{\boldsymbol{I^1}, \boldsymbol{I^2}, \dots, \boldsymbol{I^L}\}$ **.**
- la sélection d'un représentant par intervalle, l'ensemble de ces représentants composant un dictionnaire (code book) $Y = \{y_1, y_{2,} \dots, y_L\}$

Nous citons deux types des quantificateurs en fonction de principe de reconstitution. Le premier est appelé « Midrise » et sera désigné par Q_{MR}. Le second est appelé « Midtread » et sera désigné par Q_{MT}[23] [24]. Cette terminologie est basée sur ce qui se passe au voisinage du zéro, et utilise l'analogie de représenter la fonction IN-Out de quantificateur sous forme d'un escalier. Le quantificateur « Midtread » a Un niveau de reconstruction y_0 nul, cependant, le quantificateur « Midrise » a un Seuil de classification du niveau de quantification k_0 nul.

Ces deux types de quantificateurs.



Quantificateur Midtread *Quantificateur Midrise*

Figure II.1 : *Quantificateur Midtread et Quantificateur Midrise.*

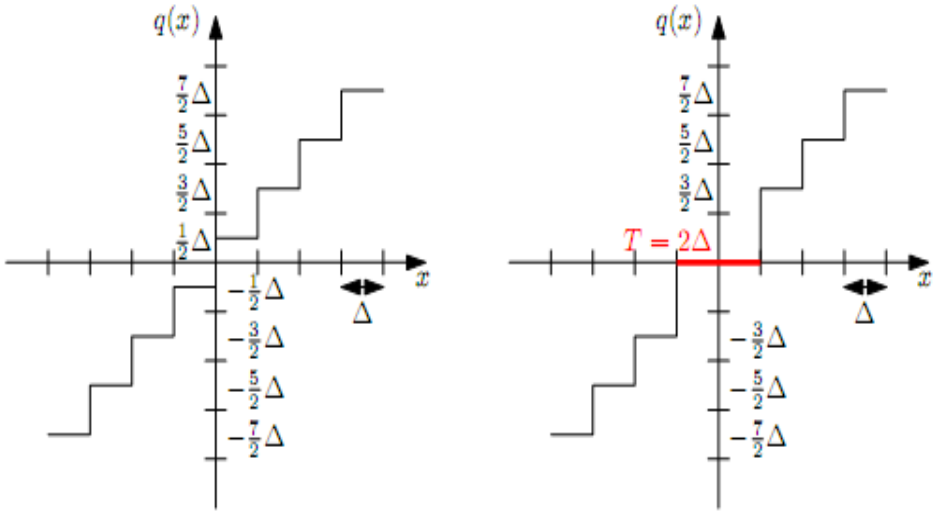
II.2.2. Quantificateur uniforme a zone morte

La quantification scalaire à zone morte (Dead-zone en anglais) est une quantification scalaire où l'intervalle autour de zéro est plus large. La zone morte notée T qualifie cet intervalle qui permet à l'ensemble des valeurs considérées comme petites d'être quantifié en une seule et même valeur égale à zéro.

Au sens strict, ce type de quantification est non uniforme ($T \neq \Delta$). Toutefois, si tous les autres pas de quantification sont égaux, on appelle cette quantification, uniforme à zone morte.

Ce type de quantification est très répandu en compression d'image. Elle peut être vue comme l'association d'une quantification uniforme et une opération de seuillage par rapport à une valeur Q [14][15].

La Figure suivant présente d'un quantificateur scalaire uniforme et un quantificateur scalaire uniforme à zone morte.



Quantificateur uniforme *Quantificateur uniforme à zone morte*

Figure II.2 : ***Quantificateur uniforme à zone morte.***

II.2.3. Quantificateur scalaire non uniforme

La quantification non uniforme est motivée par le constat qu'un quantificateur uniforme n'est pas optimal que dans le cas où le signal d'entrée a une distribution uniforme, ce qui n'est pas le cas pour la plupart des signaux, elles suivent des distributions différentes et des fois aléatoires : Les signaux de parole ont une densité de probabilité proche d'une loi de Laplace, tandis que les signaux de musique ont souvent une distribution gaussienne. La distribution des valeurs des pixels d'une image peut beaucoup varier d'une image à l'autre, et elle est rarement uniforme. Ainsi, certaines valeurs de quantification correspondant à des échantillons peu probables sont rarement utilisées, ce qui représente un gaspillage de bits.

II.2.4. Algorithme de Lloyd-MAX

Les travaux de Lloyd et de Max [16][17] ont permis d'établir les deux conditions nécessaires d'optimalité pour construire un quantificateur adapté à des sources quelconques. Comme nous.

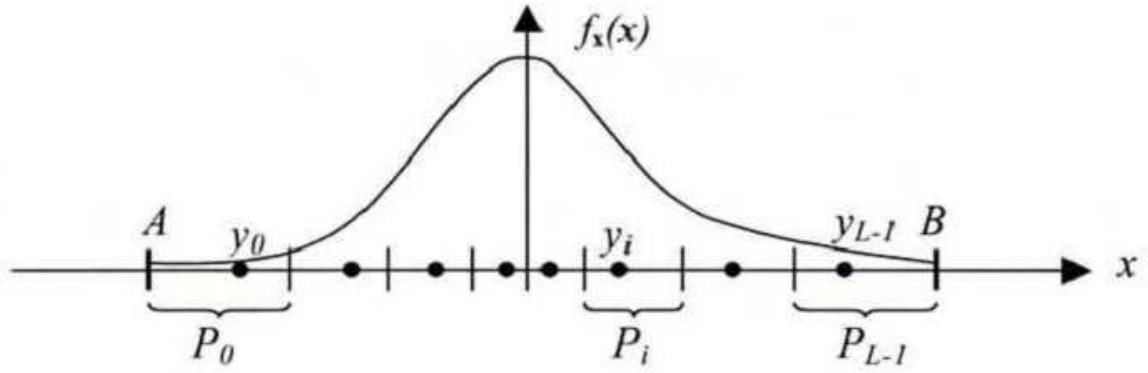


Figure II.3 : ***Quantification scalaire non uniforme de Lloyd-MAX.***

L'avons expliqué pour la QS non uniforme, définir un quantificateur "optimal" consiste à trouver les L meilleures partitions P_i du signal et les meilleurs représentants Y_i associés (les représentants du dictionnaire C) afin de minimiser la distorsion. Cette optimisation conjointe n'admet pas de solution simple. Cependant, il est possible de définir les deux conditions suivantes:

- connaissant le dictionnaire, il est possible de trouver le meilleur partitionnement. En effet, étant donné un dictionnaire $C = \{y_0, y_1, \dots, y_{L-1}\}$, il est possible de classer les valeurs de X dans chaque partition P_i telle que la règle (EQ II.2) dite du plus proche voisin soit vérifiée ;

$$P_i = \left\{ \frac{x}{d(x,y_i)} \leq d(x,y_i) \;\; \forall\, i \neq j,\; j \in \{0,1,\dots,l-1\} \right\} \qquad \text{(EQ II.2)}$$

connaissant le partitionnement, il est possible de trouver le meilleur dictionnaire. En effet, étant donné un partitionnement $P = \{P_0, P_1, \dots, P_{L-1}\}$, il est possible de trouver le meilleur représentant Y_i pour chaque partition P_i en calculant le centre de gravité ou centroïde de la partition P_i , formule (EQ II.3)

$$y_i = E[x|x \in P_i] = \frac{\int_{x \in P_i} x f_x(x) dx}{\int_{x \in P_i} f_x(x) dx} \qquad \text{(EQ II.3)}$$

De manière générale, les algorithmes d'optimisation pour la création du dictionnaire sont basés sur ces deux conditions. L'algorithme de Lloyd-Max utilise ces deux conditions afin de construire itérativement un dictionnaire de façon à minimiser la distorsion DQS.

II.2.5. Quantificateur scalaire non uniforme par compression (companding)

Pour un signal de source non-uniformément réparties, on pourrait utiliser une fonction non linéaire $F(x)$ pour le convertir en un autre signal avec une PDF proche d'une distribution uniforme. Ensuite, le quantificateur uniforme simple et efficace pourrait être utilisé.

Après les indices de quantification sont transmis (codés) et reçus ensuite par les décodeurs, ils sont d'abord quantifié inversement pour reconstituer les valeurs quantifiées de manière uniforme, puis la fonction inverse $F^{-1}(x)$ est appliqué afin de produire les valeurs finales quantifiés [23].

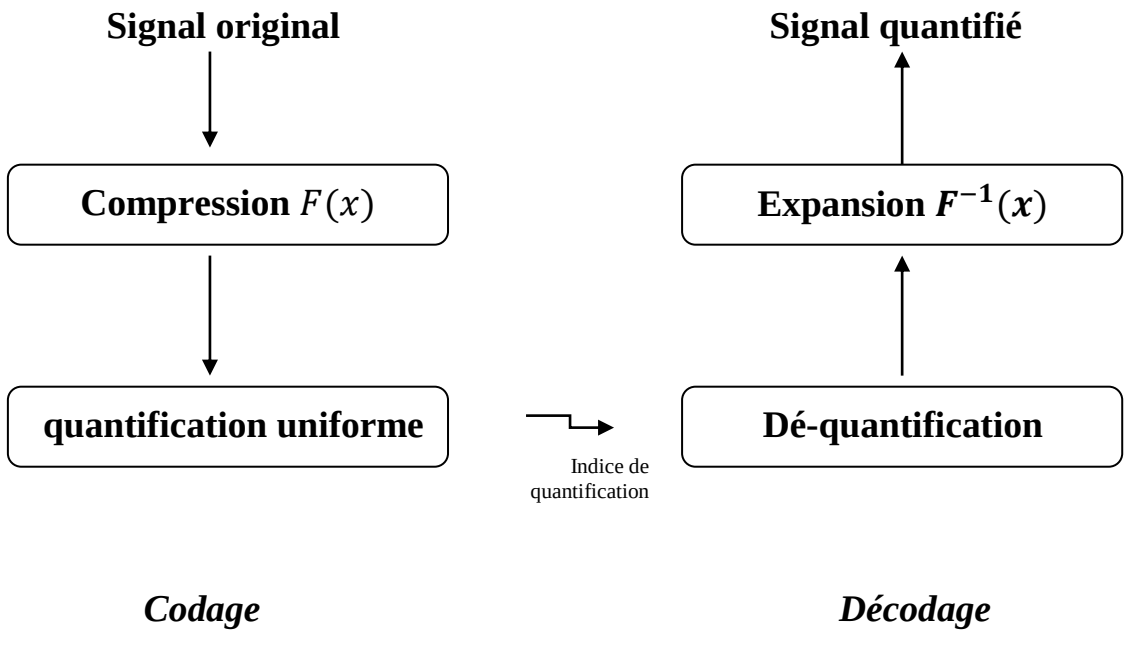
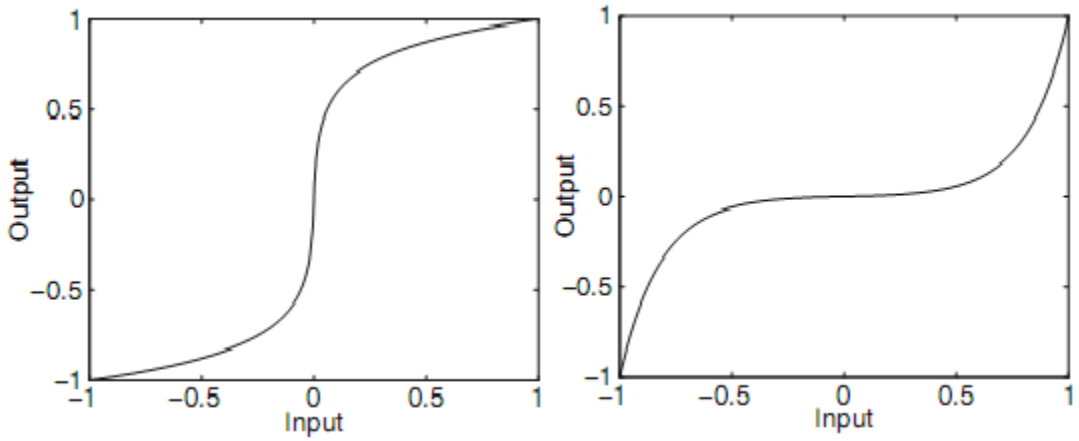


Figure II.4 : ***La quantification scalaire non uniforme par compression (companding).***

La fonction non linéaire $F(x)$ est appelée une fonction de compression, car elle a généralement une forme analogue à celle représentée sur la Figure II.5 qui amplifie les faibles amplitudes de signal source et comprime les échantillons de grandes amplitudes.



fonctions de compression *fonctions de d'expansion*

Figure II.5 : ***Exemples des fonctions de compression et d'expansion*** **[23].**

La fonction inverse $F^{-1}(x)$ est appelée une fonction d'expansion du fait que l'inverse de la compression est l'expansion. Après cette opération de compression-expansion, les bornes de décision efficaces seront non uniformes, lorsqu'elles sont vues à partir de la sortie de bloc d'expansion. Ainsi, l'effet global est une quantification non uniforme. Cette forme de compression est généralement utilisée avec des formes de PDF qui possède une densité de probabilité importante pour les petites valeurs absolues d'échantillons et décroit en direction de grandes valeurs absolues d'échantillons, tel que la distribution gaussienne et l'alsacienne, afin de rendre le PDF du signal converti similaire à une distribution uniforme.

Deux types des fonctions communes sont utilisés dans la conception des quantificateurs non uniforme par Companding: la fonction à loi de puissance (A-law) et la fonction logarithmique. Ces techniques réduisent le nombre de bits nécessaire à coder le signal et maintiennent un rapport signal à bruit acceptable.

Généralement, La fonction à loi de puissance de la forme suivante est utilisée:

$$c_{power}(|x|) = |x|^p \qquad \text{(EQ II.4)}$$

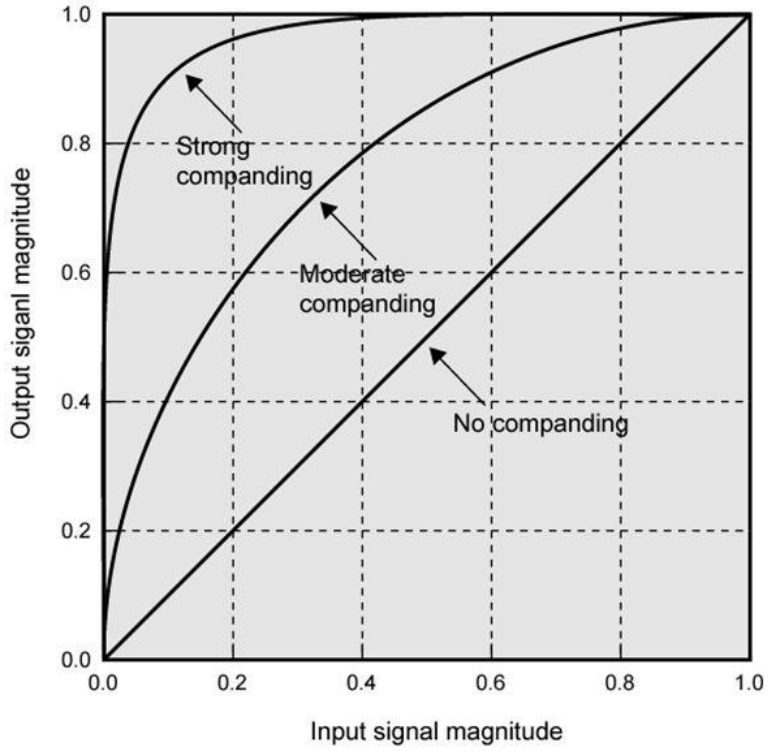


Figure II.6 : ***Fonction companding.***

II.2.6. Quantificateur scalaire non uniforme par Mu-law companding

μ-law companding est une forme de quantification par compression logarithmique, elle été introduit pour la première fois dans l'industrie téléphonique au Japon et les États-Unis (le standard G.711), pour les données audio. Elle exploite le fait que le système de perception auditive humaine est un processus logarithmique, dans laquelle les signaux sonores de hautes amplitudes ne nécessitent pas la même résolution que celles de faibles amplitudes.

L'oreille humaine est plus sensible au bruit de quantification dans les petits signaux que dans les signaux forts. «μ-law » applique une fonction de quantification logarithmique $F(x)$ pour ajuster la résolution de données en proportion du niveau du signal d'entrée. Les petits signaux sont représentés avec plus de précision que les grands signaux.

La fonction $F(x)$ de compression μ-law est définie par l'équation (EQ II.5), où μ est le coefficient de compression et x le vecteur de données à quantifier [18].

$$F(x) = \frac{sign(x)*\ln(1+\mu|x|)}{\ln(1+\mu)} \qquad 0 \leq |x| \leq 1 \qquad \text{(EQ II.5)}$$

La figure suivant présente cette fonction pour les trois valeurs de $\mu = 1$, 10, et 255. Plus la valeur de $\boldsymbol{\mu}$ est grande plus les valeurs de faible amplitudes sont amplifiées

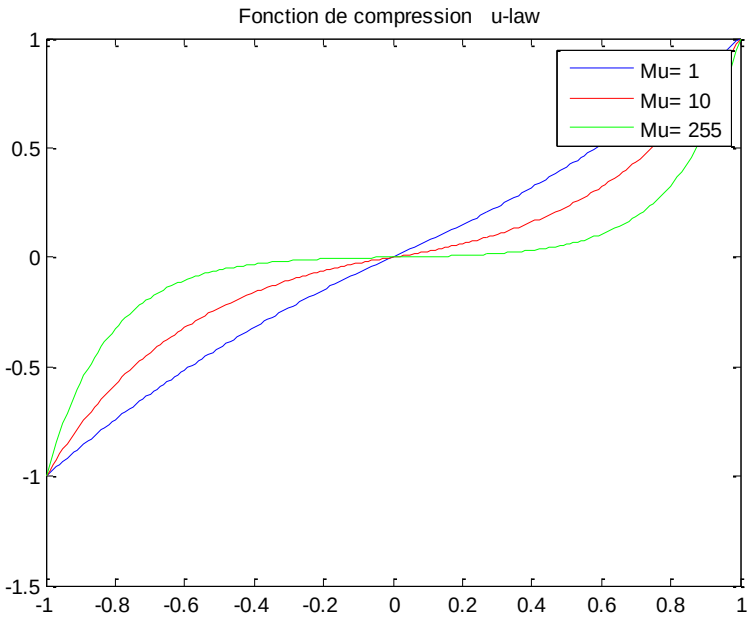


Figure II.7 : ***Fonction de compression Mu-law (μ =1, 10, 255).***

Le décodage des données codées n'est ensuite qu'une question d'inverser les étapes de l'encodage. Les étapes de quantification par compression μ-law peuvent être représentées par le schéma de la Figure II.7.

Il est clair que les grandes valeurs de μ provoquent une quantification grossière pour les grandes amplitudes et plus précis pour les faibles amplitudes.

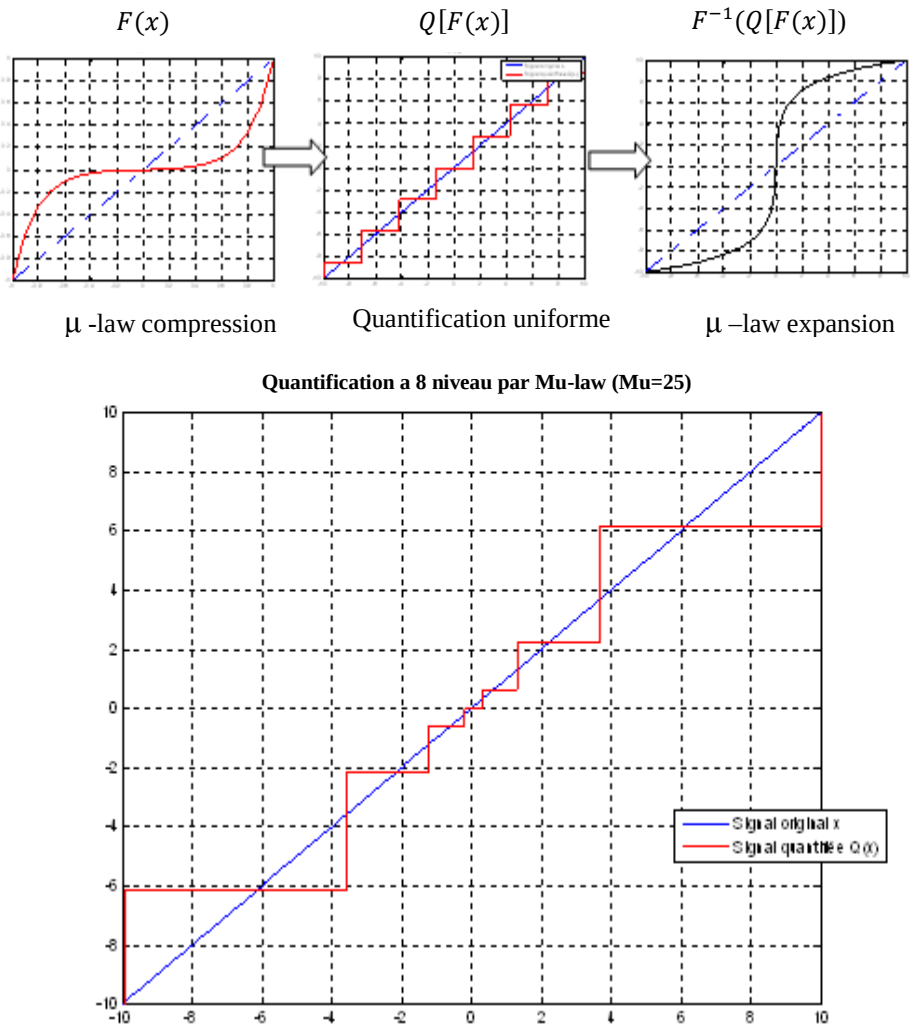


Figure II.8 : ***Etapes de quantification par Companding et quantification μ-law à 8 niveaux. (μ=25)***

II.3. Quantification et l'entropie

Après l'étape de quantification, plusieurs des valeurs de coefficients sont éliminées ou remplacées par d'autres valeurs convenables. Cependant, ces valeurs sont toujours représentées avec des codes $\{I\}$, de longueur fixe (généralement 8 ou 16 bits), alors le nombre de coefficients conservé à la sortie de quantificateur indique directement la taille de données. Si le nombre de coefficients obtenus à la sortie de quantificateur est M , la longueur du code du signal source X en bits sera tout simplement le produit du nombre de coefficients M par la longueur du mot du code en bits :

$$L_c(X) = M.l(bits) \qquad \text{(EQ II.6)}$$

Toutefois, certains coefficients sont plus redondants que les autres, donc il sera plus efficace de les coder avec moins de bits, notons la probabilité d'apparition d'un coefficient y_i par (y_i) , la longueur de code $L_c(X)$ du signal est donnée par :

$$L_c(X) = \sum_{i=1}^{M} l_i \,.\, prob(y_i) \qquad \text{(EQ II.7)}$$

Où l_i est la longueur de code de l'événement y_i.

Pour un signal X caractérisé par la fonction de distribution $f_x(x)$. Les probabilités $prob(y_i)$ dépend des limites de décision des intervalles de quantification $[b_i]$, la probabilité que y_i se produis est donnée par :

$$prob(y_i) = \int_{b_{i-1}}^{b_i} f_x\,(x) \qquad \text{(EQ II.8)}$$

Par conséquent, la taille $L_c(X)$est en fonction des limites de décision et elle est donnée par l'expression :

$$L_c(X) = \sum_{i=1}^{M} l_i . \int_{b_{i-1}}^{b_i} f_x(x).\, dx \qquad \text{(EQ II.9)}$$

À partir de cette expression, on constate que les partitions que nous sélectionnons et les codes binaires représentant ces partitions sont les paramètres qui déterminent la taille finale de données $L_c(X)$. Le problème de trouver les partitions optimales, les codes et les niveaux de représentation sont tous liés. Le but de codage entropique est de minimiser le nombre de bits moyen utilisé pour représenter les réalisations, ce qui revient à approcher le plus possible la borne inférieure de débit définie par le théorème de Shannon qui égal à l'entropie de la source Y.

$$H(Y) = -\sum_{i=1}^{M} prob(y_i) \,.\, log_2(prob(y_i)) \qquad \text{(EQ II.10)}$$

Le codage de Huffman et le codage arithmétique sont les deux codeurs les plus utilisés dans la compression d'image. Dans ce qui suit, nous présentons brièvement ces deux codeurs ainsi que leurs principes de fonctionnement.

II.4. Codage de l'information

II.4.1. Codeur de Huffman

L'imminent mathématicien, David Huffman, a proposé en 1952 une méthode statistique qui permet d'attribuer un mot de code binaire aux différents symboles à compresser (pixels ou caractères par exemple) [18]. La longueur de chaque mot de code n'est pas identique pour tous les symboles: les symboles les plus fréquents (qui apparaissent le plus souvent) sont codés avec de petits mots de code [18], tandis que les symboles les plus rares reçoivent de plus longs codes binaires. Le codeur de Huffman est très couramment employé en compression d'image. Il constitue très souvent l'étape finale produisant le flot binaire dans les méthodes par transformations.

II.4.2. Codage arithmétique

Le codage arithmétique est un codage statistique, c'est-à-dire que plus un caractère est représenté, moins il faudra de bits pour le coder. Il s'agit d'un cousin du codage de Huffman qui cependant reste toujours plus efficace que ce dernier (sauf dans le cas particulier où tous les poids des feuilles/nœuds/racines de l'arbre de Huffman sont des puissances de 2). Il est aussi plus simple à implémenter.

L'avantage que possède le codage arithmétique sur le codage de Huffman est que ce dernier va coder un caractère sur un nombre entier de bits (il ne peut coder sur 1.5 bits) là où le codage arithmétique le peut. Par exemple, si un caractère est représenté à 90%, la taille optimale du code du caractère serait de 0.15 bit, alors que Huffman coderait sûrement ce symbole sur 1 bit, soit 6 fois trop [19] [20].

Ce codage n'est que très peu utilisé en pratique mais elle reste présente, notamment dans le format JPEG2000 [21].L'association de codage arithmétique à un RLE pour la compression d'image sans perte dans [22] a porté une amélioration nette et considérable.

Pour présenter la compression, nous allons utiliser un exemple et nous décrirons chaque étape de compression. Codons le mot "ESIPE" à l'aide du codage arithmétique.

La première étape consiste à décompter chaque lettre du mot. Nous avons donc 2 'E' ,'S', 1 'I' et 1 'P'. Nous en générons alors une probabilité de présence dans le mot soit 40% de chance de trouver un E et 20% de chance pour les autres lettres. Dernière actions à effectuer pour cette première partie, nous affectons à chaque lettre un intervalle entre 0 et 1 de la manière suivante :

- La lettre 'E' à une probabilité de 40% (soit 0.4). Son intervalle est donc [0, 0.4[
- La lettre 'P'a une probabilité de 20% (soit 0.2). Son intervalle est donc [0.4, 0.6[
- Etc...

On obtient dès lors le tableau suivant :

Lattre	probabilité	intervalle
E	$4/10$	$[0, 0.4[$
S	$2/10$	$[0.4, 0.6[$
I	$2/10$	$[0.6, 0.8[$
P	$2/10$	$[0.8, 1[$

Tableau II.1 : ***Résultats du probabilité obtenus par utilisant codage arithmétique.***

Le codage va maintenant consister à remplacer le mot ESIPE par un nombre flottant lui correspondant. Pour cela, le mot va se voir affecter un intervalle compris entre 0 et 1 où chaque nombre compris entre les deux intervalles permettra de retrouver le mot ESIPE.

L'algorithme appliqué est le suivant : le mot commence avec un intervalle de [0,1[. Puis pour chaque lettre croisée, nous appliquons la formule suivante :

- La borne inférieure (BI) du mot est modifiée avec le résultat du calcul "BI + (BS – BI) * Borne_Inférieure_Lettre"
- La borne supérieure (BS) du mot est modifiée avec le résultat du calcul "BI + (BS – BI) * Borne_Supérieure_Lettre"

Le tableau suivant montre les étapes du calcul:

Lettre	Borne Inférieure	Borne Supérieure
	0.0	1
E	0.0	0.1
S	0.16	0.24
I	0.208	0.224
P	0.2208	0.224
E	0.2208	0.22208

Tableau II.2 : ***Résultats du nombre nécessaire pour chaque caractère.***

Dès lors, tous nombre flottant entre 0.2208 et 0.22208 est le format compressé du mot "ESIPE"

II.5. Conclusion

Selon le type des données qu'on veut quantifier et coder on doit choisir le quantificateur et le codeur adéquat, dans notre travail (compression d'image) nous avons choisi d'utiliser le quantificateur uniforme, le quantificateur non uniforme Mu-law, et nous avons applique l'algorithme de Lloyd max afin de voir son effet quant à la qualité d'image. Nous avons en outre utilisé le codeur arithmétique pour les quatre cas précédents de quantificateurs.

Le chapitre suivant expose et analyse les différents résultats.

CHEAPITR III:

Résultats et discussions

III.1. Introduction

Dans le présent chapitre une étude comparative est menée afin d'arriver un taux de compression assez élevé avec moins des pertes en terme de qualité.

Ce chapitre présente différents résultats de compression pour les quatre cas suivants :

1- utilisation du quantificateur scalaire uniforme.
2- utilisation du quantificateur scalaire non-uniforme par application de Mu-law companding.
3- utilisation du quantificateur scalaire uniforme avec optimisation des partitions par l'algorithme de Lloyd-Max.
4- utilisation du quantificateur scalaire non-uniforme par application de Mu-law companding et optimisation des partitions par l'algorithme de Lloyd-Max.

Le codeur utilisé est le codeur arithmétique.

III.2. Algorithme de compression

Nous avons développé l'algorithme de compression proposé par KADRI Oussama [23] sera donc décrits par la figure suivante :

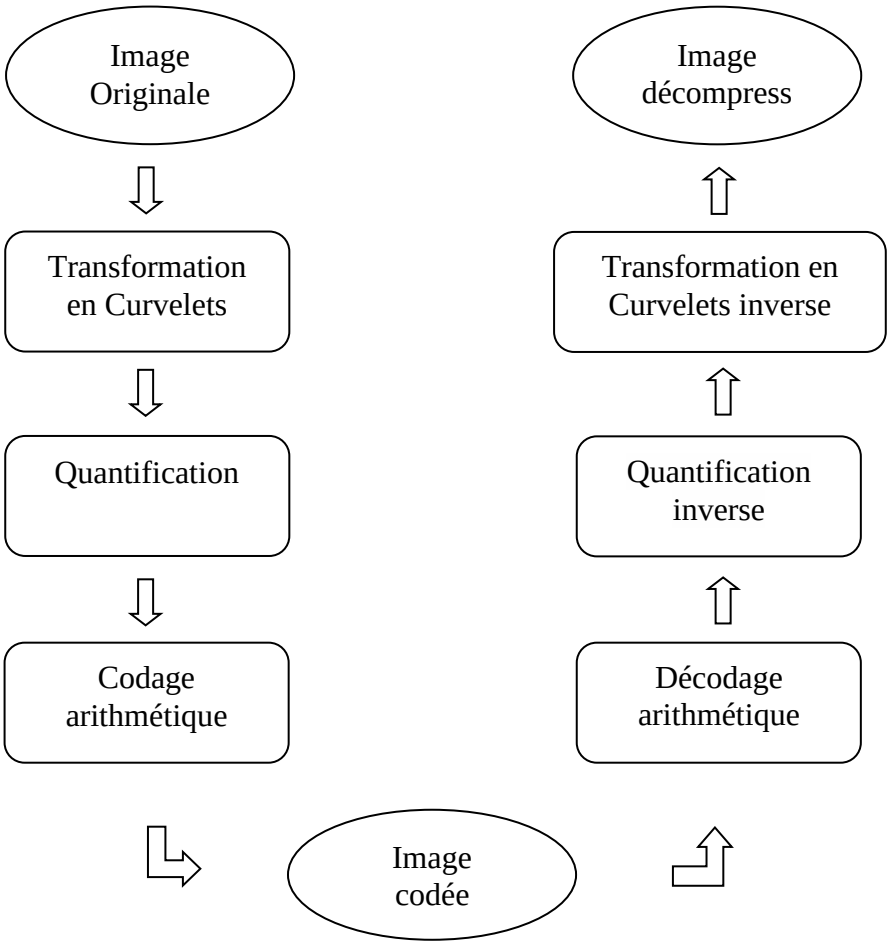


Figure III.1 : ***Algorithme de compression et décompression*** **[23].**

III.3. Présentation et discussions

Nous avons appliqué l'algorithme de la figure III.1 en utilisant différents types de quantificateur scalaire à suivant :

- ✓ quantification scalaire uniforme
- ✓ quantification scalaire non-uniforme avec Mu-law
- ✓ quantification scalaire non-uniforme avec Lloyd-Max
- ✓ quantification scalaire non-uniforme avec Mu-law et Lloyd-Max

et ce en variant à chaque fois les paramètres tels que le pas de partition et/ou le facteur d'expansion

Nous avons obtenu les résultats suivants :

III.3.1. Compression en utilisation un quantificateur scalaire uniforme

Le tableau III.1 présente et compare les résultats du $PSNR$ et les taux de compression obtenus en fonction du facteur de partition P pour trois images de tests.

Image / Facteur de partition	Lena		Girl		Peppers	
	$PSNR\ (db)$	$T\ COM\ \%$	$PSNR\ (db)$	$T\ COM\ \%$	$PSNR\ (db)$	$T\ COM\ \%$
50	31.4876	33.0654	31.8941	33.0860	32.4203	33.6181
100	32.5063	36.3666	32.7902	36.3807	33.5512	36.9316
150	32.7174	38.3720	32.9752	38.3659	33.7990	38.9060

Tableau III.1 : ***Résultats obtenus par utilisant un quantificateur scalaire uniforme.***

Les figures III.2 et III.3 présentent les $PSNR$ et les taux de compression respectivement en fonction du facteur de partition P pour chaque image de test.

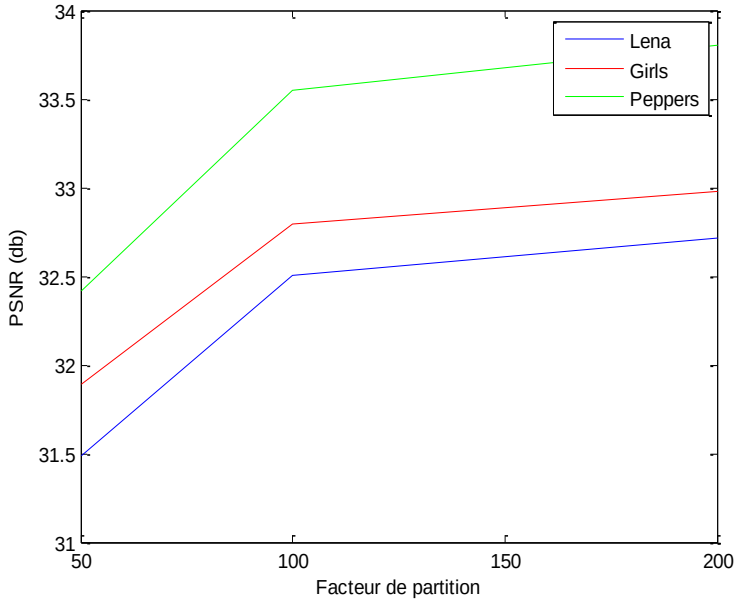


Figure III.2 : ***Evolution du PSNR en utilisation un quantificateur scalaire uniforme pour différentes images de test.***

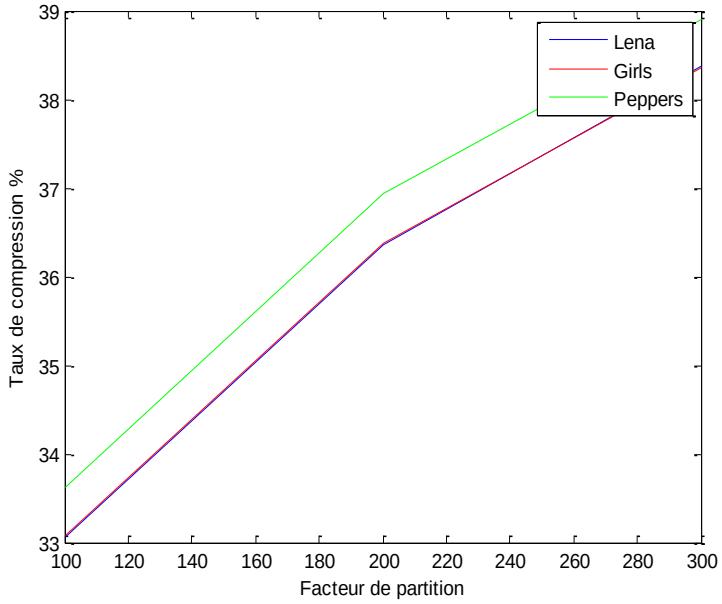


Figure III.3 : ***Evolution du Taux de compression en utilisation un quantificateur scalaire uniforme pour différentes images de test.***

On remarque que le taux de compression augmente en fonction du facteur de partition P ainsi que le $PSNR$, donc la meilleure compression avec maintien le $PSNR \geq 30\ db$ est celle correspondant au facteur de partition $(P = 50\ pas)$.

Pour l'appréciation visuelle des images compressées par la méthode proposée, nous présentons les trois images de tests : Lena, Girls et Peppers à meilleur taux de compression avec un $PSNR \geq 30\ db$ sur la figure III.4.

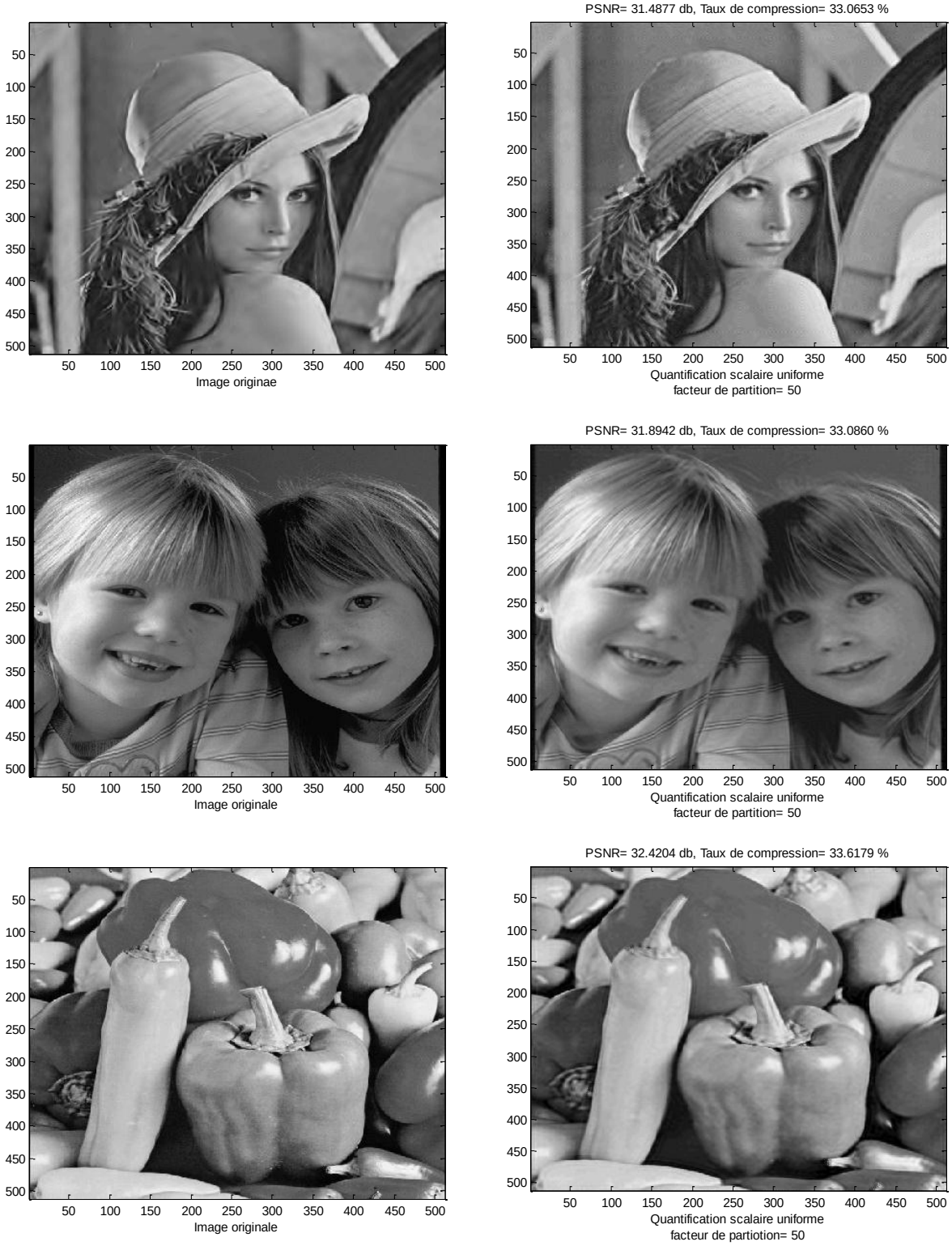


Figure III.4 : ***Résultats obtenus par utilisant un quantificateur scalaire uniforme.***

III.3.2. Compression en utilisation un quantificateur scalaire non-uniforme avec Mu-law companding

Le tableau III.2 présente et compare les résultats du $PSNR$ et du taux de compression obtenus en fonction du facteur de partition P et du facteur d'expansion Mu pour les trois images de tests.

Image →		Lena		Girls		Peppers	
Facteur de partition	Facteur d'expansion	$PSNR(db)$	$T\ COM\%$	$PSNR(db)$	$T\ COM\%$	$PSNR(db)$	$T\ COM\%$
100	255	29.1913	31.0409	30.4294	32.0487	31.9880	31.7824
200	255	32.1380	34.4224	32.4996	35.4130	33.4667	35.1700
300	255	32.9211	36.4202	32.9391	37.4115	33.7800	37.1705
100	150	29.4087	31.5475	30.7162	32.5232	29.8689	32.2433
200	150	32.2037	34.9186	32.501	35.9064	32.643	35.6502
300	150	32.9082	36.9084	32.8977	37.920	33.378	37.6609
100	50	32.8324	32.6603	31.6139	33.5978	31.2151	33.3631
200	50	33.3472	36.0519	32.799	37.0253	33.2061	36.7698
300	50	33.4560	38.0455	33.030	39.0296	33.6789	38.7716

Tableau III.2 : ***Résultats obtenus par utilisant un quantificateur scalaire non-uniforme avec Mu-law.***

Les figures III.5 et III.6 présentent les $PSNR$ et les taux de compression respectivement en fonction du facteur de partition P avec un facteur d'expansion ($Mu = 255$) pour chacune image de test.

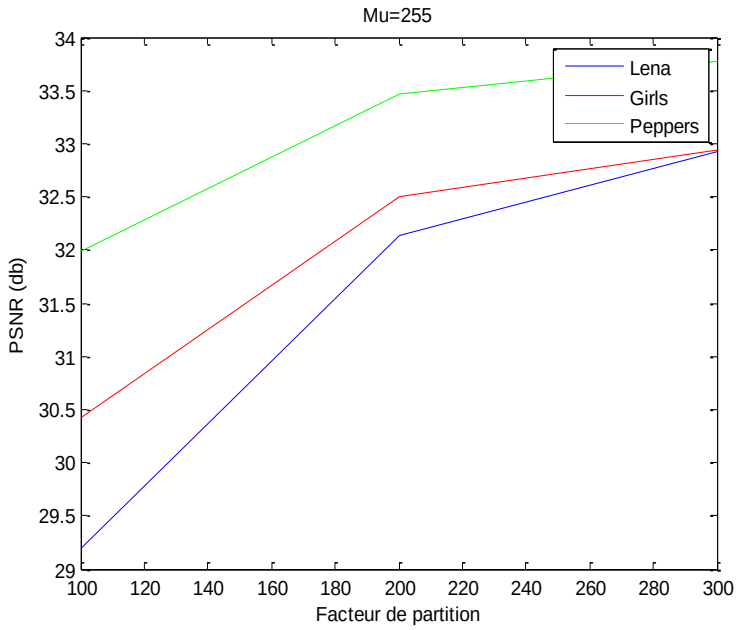


Figure III.5 : ***Evolution du PSNR en utilisation un quantificateur scalaire non-uniforme avec Mu-law pour différentes images de test.***

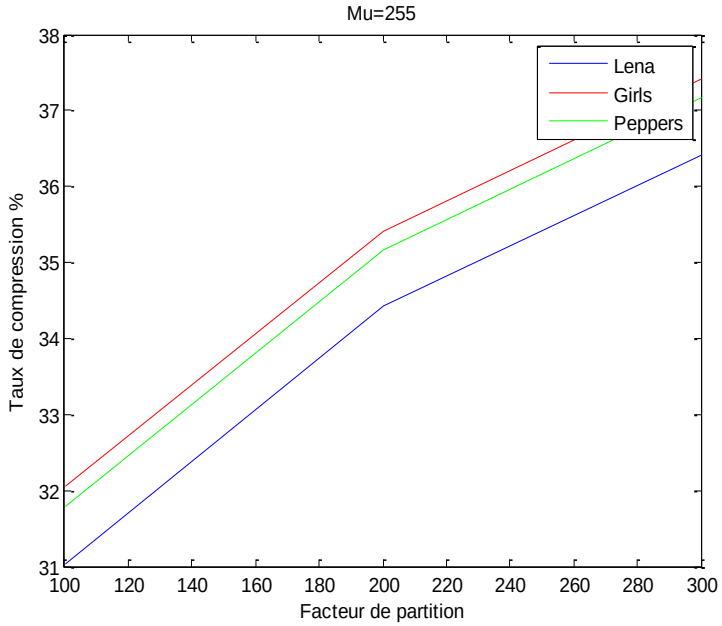


Figure III.6 : ***Evolution du taux de compression en utilisation un quantificateur scalaire non-uniforme avec Mu-law pour différentes images de test.***

On remarque selon les résultats obtenus dans le tableau III.2 et les figures III.5 et III.6 que le taux de compression et le $PSNR$ augmentent respectivement en fonction du facteur de partition P, tandis que les deux diminuent en fonction du facteur d'expansion Mu, donc on peut montrer que la meilleure compression avec maintien le $PSNR \approx 30\ db$ correspond à un facteur de partition ($P = 100\ pas$) et un facteur d'expansion ($Mu = 255$).

Pour l'appréciation visuelle des images compressées par la méthode proposée, nous représentons les trois images de tests : Lena, Girls et peppers qui ont le meilleur taux de compression avec $PSNR \approx 30\ db$ dans la figure III.7.

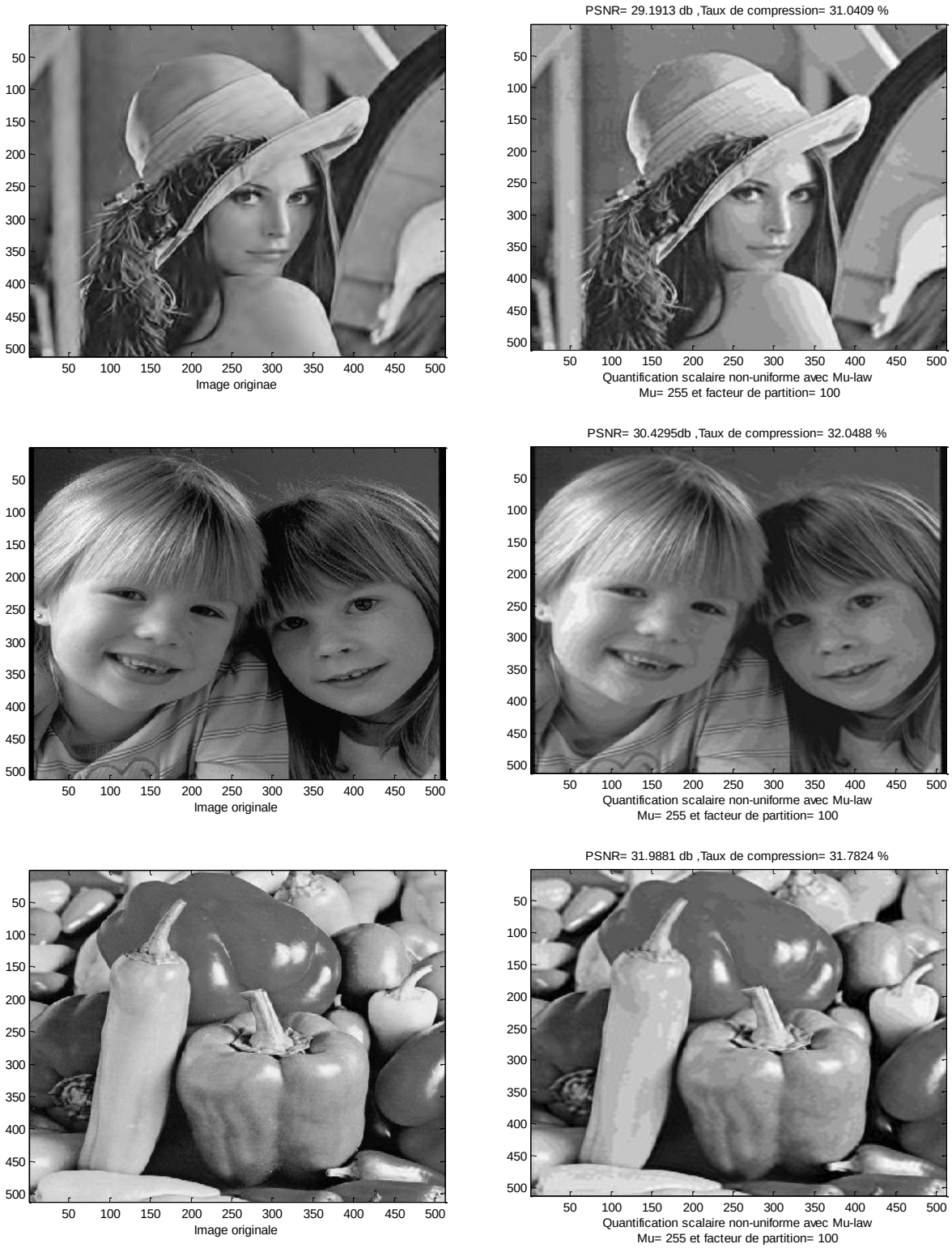


Figure III.7 : ***Résultats obtenu par utilisant un quantificateur scalaire non-uniforme Mu law.***

III.3.3. Compression en utilisation un quantificateur scalaire uniforme avec optimisation des partitions par Lloyd-Max

Le tableau III.3 présente et compare les résultats du $PSNR$ et les taux de compression obtenus en fonction du facteur de partition P pour les trois images de tests.

Image / Facteur de partition	Lena		Girl		Peppers	
	$PSNR\ (db)$	$T\ COM\ \%$	$PSNR\ (db)$	$T\ COM\ \%$	$PSNR\ (db)$	$T\ COM\ \%$
50	33.4110	33.8603	33.1243	33.8409	33.8907	34.2330
100	33.4879	36.6565	33.1790	36.7321	33.9810	37.3397
150	33.5079	38.5353	33.1895	38.5437	33.9978	39.0708

Tableau III.3 : ***Résultats obtenus par utilisant un quantificateur scalaire uniforme optimisé par Lloyd-Max.***

Les figures III.8 et III.9 présentent les $PSNR$ et les taux de compression respectivement en fonction du facteur de partition P pour chaque image de test.

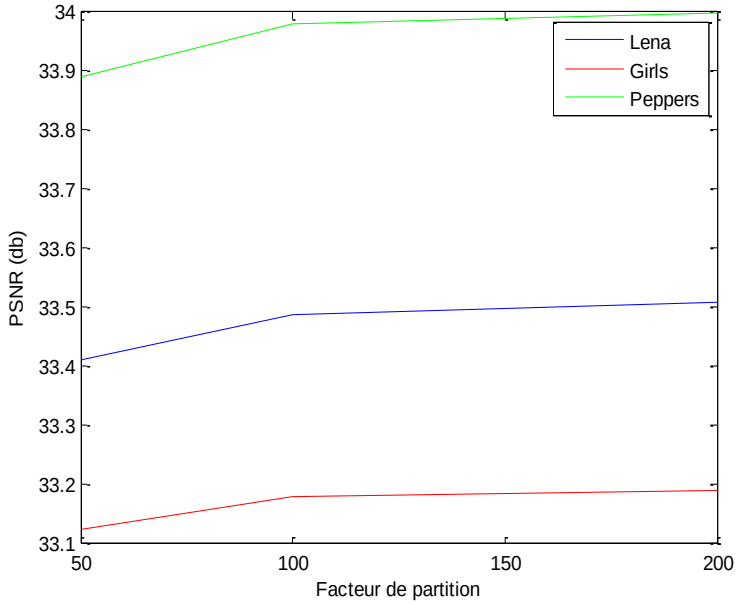


Figure III.8 : ***Evolution du PSNR en utilisation un quantificateur scalaire uniforme optimisé par Lloyd-Max pour différentes images de test.***

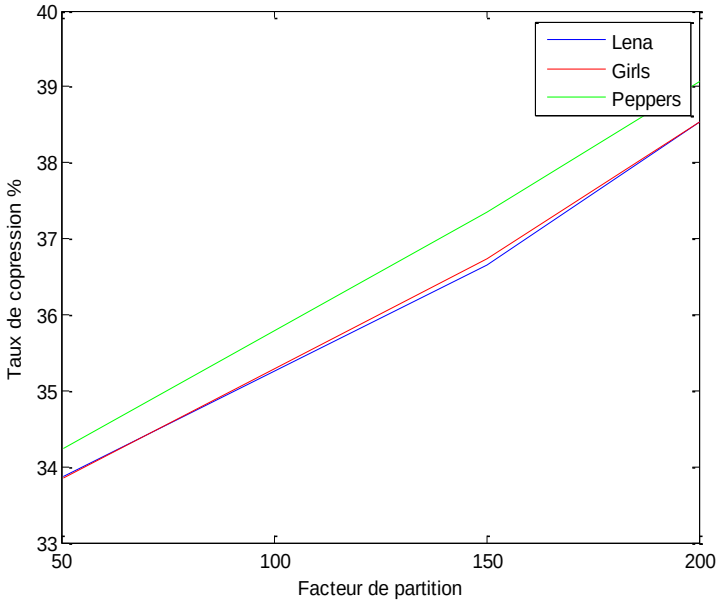


Figure III.9 : ***Evolution du Taux de compression en utilisation un quantificateur scalaire uniforme optimisé par Lloyd-Max pour différentes images de test.***

On remarque que le taux de compression augmente en fonction du facteur de partition P ainsi que du $PSNR$, donc la meilleure compression en maintenant un $PSNR \geq 30\ db$ est celle correspondant au facteur de partition ($P = 50\ pas$).

Pour l'appréciation visuelle des images compressées par la méthode proposée, nous représentons les trois images de tests : Lena, Girls et peppers qui ont le meilleur taux de compression pour un $PSNR \geq 30\ db$ dans la figure III.10.

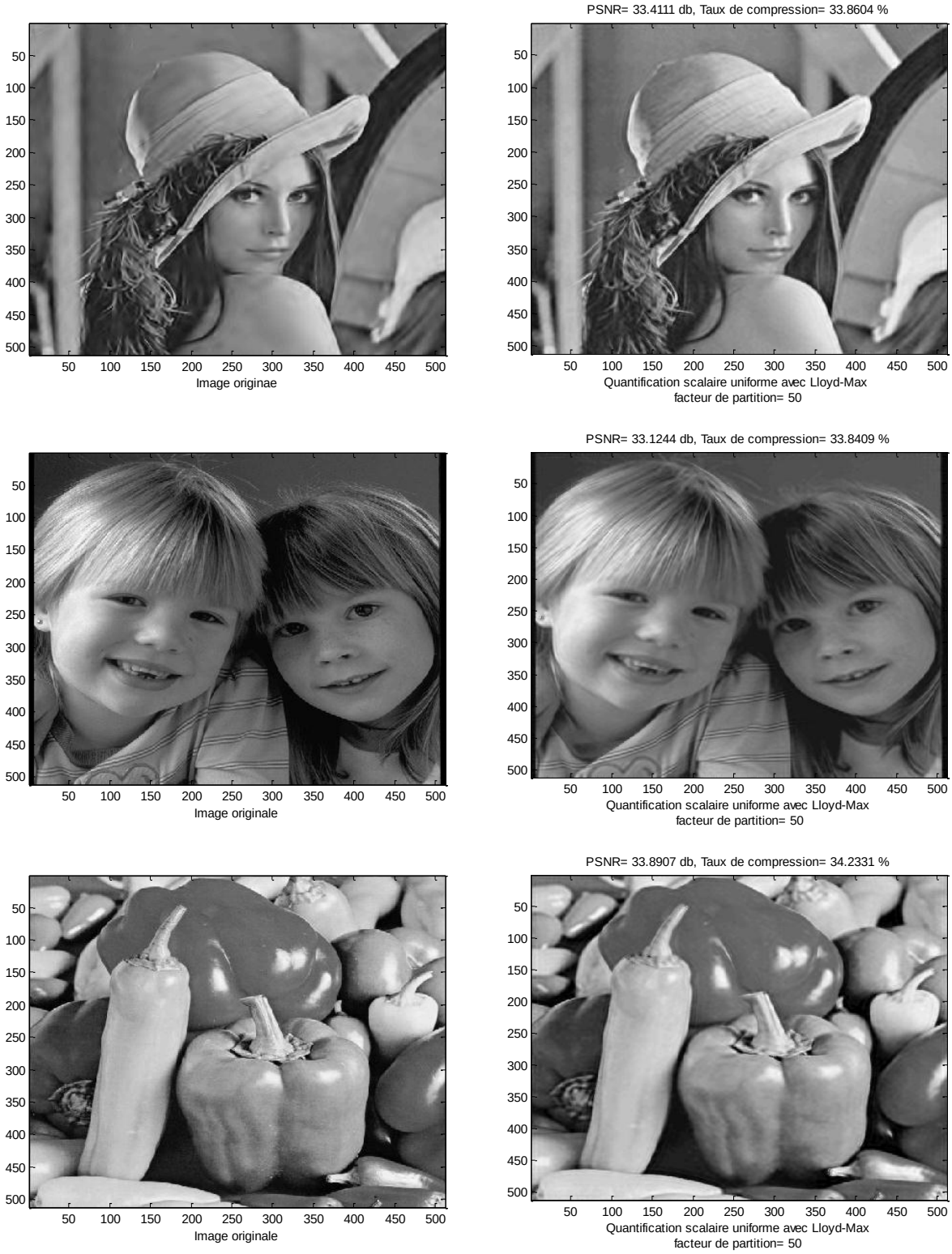


Figure III.10 : ***Résultats obtenu par utilisant un quantificateur scalaire uniforme optimisé par Lloyd-Max.***

III.3.4. Compression en utilisation un quantificateur scalaire non-uniforme avec Mu-law et optimisation des partitions par l'algorithme de Lloyd-Max

Le tableau III.4 présente et compare les résultats des $PSNR$ et des taux de compression obtenus en fonction du facteur de partition P et du facteur d'expansion Mu pour les trois images de tests.

Image →		Lena		Girls		Peppers	
Facteur de partition	Facteur d'expansion	$PSNR(db)$	$T\ COM\%$	$PSNR(db)$	$T\ COM\%$	$PSNR(db)$	$T\ COM\%$
100	255	23.9862	32.4444	28.0366	32.8697	26.8631	32.8720
200	255	27.2741	35.2917	29.9603	35.9199	29.1917	35.8463
300	255	29.9868	36.9289	30.9634	37.7120	30.0911	37.6312
100	150	25.4678	32.7282	28.6279	33.4030	27.2651	33.2443
200	150	28.6326	35.5976	30.2052	36.4997	29.4923	36.2861
300	150	31.1566	37.3184	31.9791	38.1866	30.8987	38.1160
100	50	20.6811	33.7437	29.6512	34.3357	28.6797	34.2285
200	50	30.5313	36.5993	31.0217	37.3348	30.9566	37.2795
300	50	32.3176	38.4007	32.9940	39.1960	32.4839	39.1082

Tableau III.4 : ***Résultats obtenus par utilisant un quantificateur scalaire non-uniforme avec Mu-law et optimisé par Lloyd-Max.***

Les figures III.11 et III.12 présentent les $PSNR$ et les taux de compression respectivement en fonction du facteur de partition P sachant que le facteur d'expansion choisit est $(Mu = 255)$ pour chaque image de test.

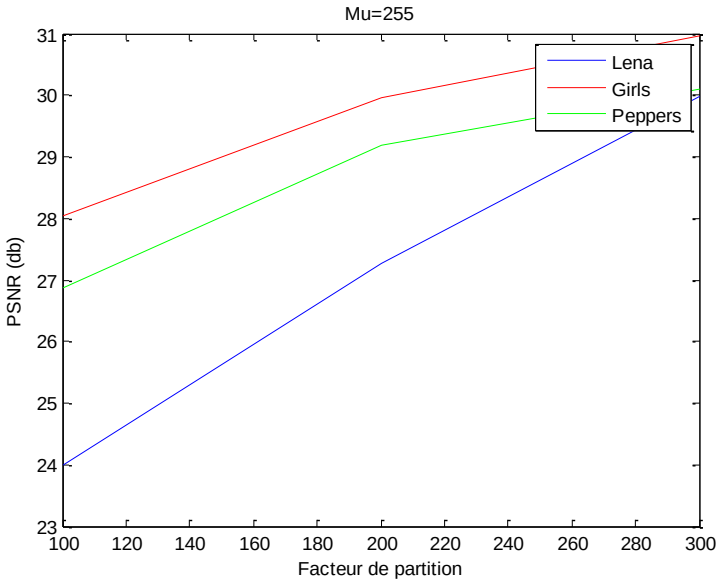


Figure III.11 : ***Evolution du PSNR en utilisation un quantificateur scalaire non-uniforme avec Mu-law et optimisé par Lloyd-Max pour différentes images de test.***

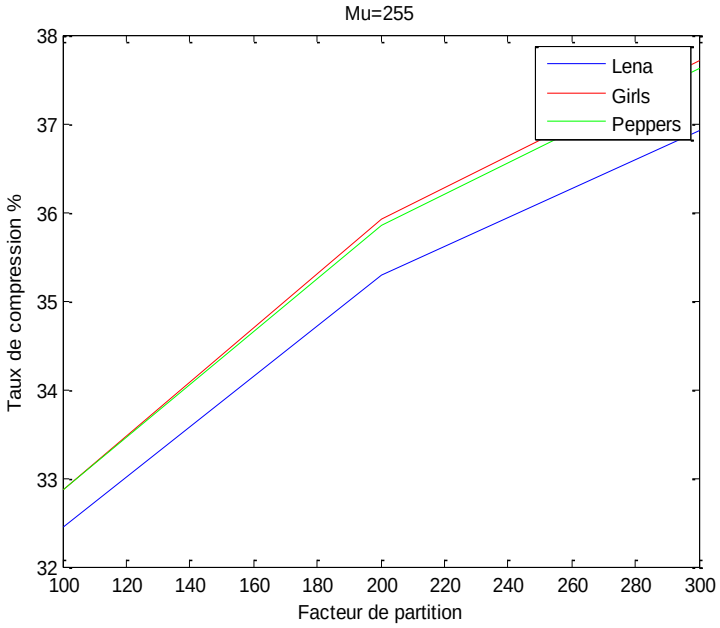


Figure III.12 : ***Evolution du Taux de compression en utilisation un quantification scalaire non-uniforme avec Mu-law et optimisé par Lloyd-Max pour différentes images de test.***

On remarque selon les résultats obtenus dans le tableau III.4 et les figures III.11 et III.12 que le taux de compression et le $PSNR$ augmentent respectivement en fonction du facteur de partition P, tandis que les deux diminuent en fonction du facteur d'expansion Mu, donc on peut montrer que la meilleure compression en maintenant un $PSNR \approx 30\,db$ est celle correspondant à un facteur de partition $(P = 50\,pas)$ avec un facteur d'expansion $(Mu = 255)$.

Pour l'appréciation visuelle des images compressées par la méthode proposée, nous représentons les trois images de tests : Lena, Girls et peppers utilisant le meilleur taux de compression pour un $PSNR \approx 30\ db$ sur la figure III.13.

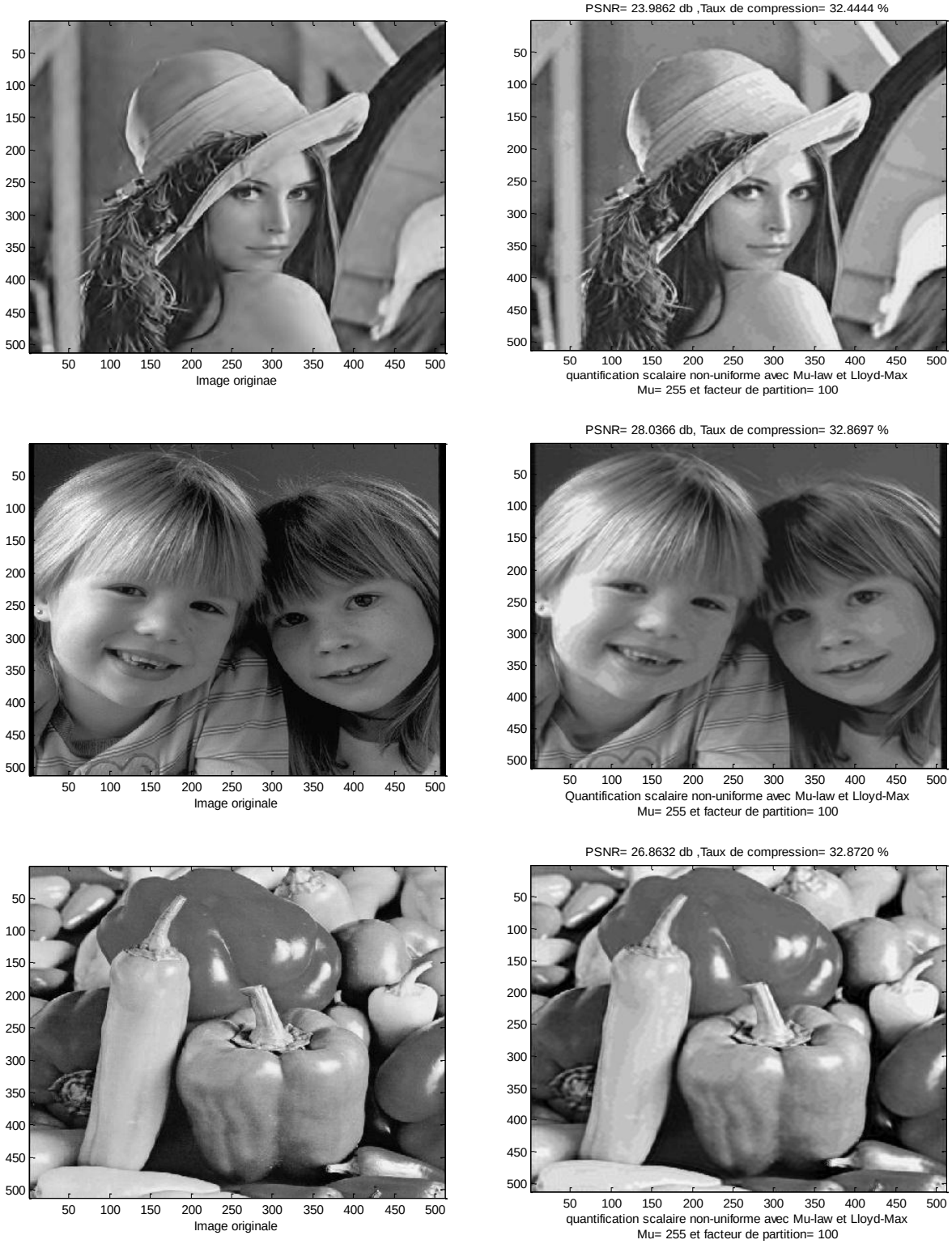


Figure III.13 : ***Résultats obtenu par utilisant un quantificateur scalaire non-uniforme avec Mu-law et optimisation par Lloyd max.***

Finalement on remarque que le meilleur taux de compression obtenu est égal 31.0409 % et qui correspond à un quantificateur non-uniforme avec Mu Law avec ($Mu = 255$) et ($P = 100$).

III.5. Conclusion

Dans ce chapitre nous avons implémenté un algorithme de compression d'images fixes basée sur une transformation en Curvelets et différents types de quantificateurs et un codeur arithmétique. Nous avons remarqué que l'utilisation des Curvelets avec l'algorithme de compression proposé donne de bons résultats en utilisant un quantificateur scalaire non-uniforme avec Mu-law, donne de meilleurs résultats par rapport aux autres types de quantificateurs.

Conclusion générale

Comme de nombreuses méthodes de compression, la compression d'image en curvelets est basé sur des principes mathématiques très compréhensibles. Mais la difficulté intervient lorsque l'on entre en détail dans les démonstrations de l'algorithme.

À l'heure actuelle la méthode de compression d'image en curvelets est parmi les plus utilisées parce qu'elle atteint des taux de compression très élevés sans que les modifications de l'image ne puissent être décelées par l'oeil humain.

De plus, beaucoup d'implémentations permettent de choisir la qualité de l'image comprimée grâce à l'utilisation les matrices de quantification paramétrables.

La réputation et donc le nombre d'utilisateurs d'un algorithme de compression dépendent de différents facteurs : le rapport taille/qualité, taux de compression et la vitesse de compression et de décompression. Il est donc intéressant de comparer les différentes méthodes, pour pouvoir choisir la plus adaptée à nos besoins et à son utilisation.

Nous avons remarqué que l'utilisation d'un quantificateur scalaire non uniforme Mu law donne de meilleurs résultats quant à l'utilisation d'autres quantificateurs tels que : le quantificateur uniforme et le quantificateur uniforme optimisé par Lloyd-Max.

Notre travail nécessite une suite en appliquant d'autres types de quantificateurs tels que le quantificateur vectoriel, ou l'application d'autres algorithmes tels que le SPIHT ou le SPECK.

Les algorithmes développés nécessitent des investissements au niveau du temps de calcul.

Bibliographie

[1] X. Delaunay, “Compression d’images satellite par post-transformées dans le domaine ondelettes,” Doctorat, L’UNIVERSITÉ DE TOULOUSE, 2008.

[2] C. Lambert-nebout, C. Latry, and G. Moury, “La compression embarquée d ’ images pour les systèmes optiques d ’ observation spatiale,” *GRETSI’01 18th Symp. signal image Process.*, 2001.

[3] Z. Athmane, “Ondelettes et techniques de compression d ’ images numérique” , Doctorat, UNIVERSITE MOHAMED KHIDER BISKRA, 2013.

[4] J. Reichel, G. Menegaz, M. J. Nadenau, and M. Kunt, “Integer wavelet transform for embedded lossy to lossless image compression,” *IEEE Trans. Image Process.*, vol. 10, no. 3, pp. 383–392, 2001.

[5] M. J. Weinberger, G. Seroussi, and G. Sapiro, “The LOCO-I lossless image compression algorithm: Principles and standardization into JPEG-LS,” *IEEE Trans. Image Process.*, vol. 9, no. 8, pp. 1309–1324, 2000.

[6] A. Fallis, “COMPRESSION D’IMAGES PAR TRANSFORMEE EN ONDELETTE,” *J. Chem. Inf. Model.*, vol. 53, no. 9, pp. 1689–1699, 2013.

[7] Y. Meyer, “Signal processing and compression with walvelet packets,” *Proc. Conf. Wavelets Appl.*, 1990.

[8] J.-L. Starck, E. J. Candès, and D. L. Donoho, “The curvelet transform for image denoising.,” *IEEE Trans. Image Process.*, vol. 11, no. 6, pp. 670–84, 2002.

[9] L. Boubchir, “Modélisation statistique multivariée des images dans le domaine des transformées multi-échelles orientées et non-orientées,” *Proc. Int. Conf. Image Signal Process. their Appl.*, 2009.

[10] D. L. Donoho, “Digital curvelet transform: strategy, implementation, and experiments,” *Proc. SPIE*, pp. 12–30, 2000.

[11] F. M. Kazemi, S. Samadi, H. R. Poorreza, and M. R. Akbarzadeh-T, “Vehicle recognition using curvelet transform and SVM,” *Proc. - Int. Conf. Inf. Technol. Gener. ITNG 2007*, pp. 516–521, 2007.

[12] P. Loh and M. E. J. Candes, “Image Reconstruction With Ridgelets,” pp. 1–8, 2003.

[13] E. J. Candès and D. L. Donoho, “Continuous curvelet transform: II. Discretization and frames,” *Appl. Comput. Harmon. Anal.*, vol. 19, no. 2, pp. 198–222, 2005.

[14] E. Candès, L. Demanet, D. Donoho, and L. Ying, “Fast Discrete Curvelet Transforms.,” *Multiscale Model. Simul.*, vol. 5, no. 3, pp. 861–899, 2006.

++

[14] M.S. Joshi, R.R.Manthalkar, and Y.V. Joshi. Image Compression Using Curvelet, Ridgelet and Wavelet Transform, A Comparative Study. ICGST International Journal on Graphics, Vision and Image Processing (GVIP). October 2008, Vol. 8, pp 25-34..

[15] Kamlesh Gupta, Sanjay Silakari. Performance Analysis Of Image Compression Using Curvelets Transform. Journal of Signal and Image Processing. July 2011, Vol. 2, pp 13-.

[16] S.P. Lloyd, "Least Squares Quantization in PCM", IEEE Transactions on Information Theory, vol. IT-28, p. 129-137, Mars 1982.

[17] J. Max, "Quantizing for minimum distortion Il, IEEE Transactions on Information Theory, p.7-12, Mars 1960.

[18] Pu, Ida Mengyi. Fundamental Data Compression. Oxford : Elsevier's Science and Technology, 2006. pp 180-181. ISBN-10:0-7506-6310-3.

[19] David A, Huffman. A Method for the Construction of Minimum-Redundancy Codes. s.l. : Proceedings of the I.R.E., September 1952. pp 1098-1101.

[20] S.Medouakh, Z-E. Baarir. Study of the Standard JPEG2000 in Image Compression. International Journal of Computer Applications. March 2011, Vol. 18, pp 27-33

[21] Said, Amir. Introduction to Arithmetic Coding - Theory and Practice. Palo Alto -USA- : Imaging Systems Laboratory -HP Laboratories -, April 21, 2004. Rapport technique. HPL-2004-76.

[22] Asadollah Shahbahrami, Ramin Bahrampour, Mobin Sabbaghi Rostami, Mostafa Ayoubi Mobarhan. Evaluation of Huffman and Arithmetic Algorithms for Multimedia Compression Standards. International Journal of Computer Science, Engineering and Applications. N°4, August 2011, Vol. 1, pp 34-47

[23] KADRI Oussama, " Compression d'images fixes par Ondelettes géométriques par utilisation des Curvelets et différents types d'interpolation dans la quantificat", Thèse de Magister en Electronique, UNIVERSITE MOHAMED KHIDER BISKRA, 2014.

[24] ZEGHIB Maymouna and SEROUTI Abla, "Compression d'images avec curvelets", Master en Electronique, UNIVERSITE Echahid Hamma Lakhdar El Oued, 2015.