République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Mohammed Seddik BENYAHIA – Jijel

Faculté des Sciences et de la Technologie

Département d'Électronique

MÉMOIRE DE FIN D'ETUDES

Présenté en vue de l'obtention du diplôme

De **MASTER** en **Systèmes des télécommunications**

Étude de l'effet des transformations réversibles sur la performance de compression sans perte des images couleur.

Encadré par :

Mr. Tahar BRAHIMI

Réalisé par :

M^elle^ Rabiaa YAKHLEF

M^elle^ Manel KOUAHI

Année universitaire : 2019-2020

Remerciement

Nous remercions en premier lieu, Allah le tout puissant de nous avoir accordé santé, force et courage pour accomplir ce travail.

Nous tenons à remercier particulièrement M. Brahimi Tahar pour son encadrement, ses conseils, sa disponibilité et son aide

En second lieu, tous les enseignants, qui nous ont honorés lors de notre cursus en nous prodiguant le savoir avec dévouement et particulièrement Mr Boukkeroum.

Ensuite, tout le corps enseignants de la faculté de Technologie en général, et ceux du département d'électronique en particulier.

Enfin Messieurs les membres du jury, qui ont accepté de nous honorer en acceptant d'examiner, de juger et d'évaluer notre mémoire de fin d'études pour l'obtention du Master.

Merci à ceux qu'on oublie toujours : Nos maîtres d'école, de collège, et de lycée. Nous vous somme très reconnaissante.

A toute personne qui a contribué de près ou de loin à l'élaboration de ce travail, MERCI.

Dédicace

Avec tout mon amour éternel et avec l'intensité de mes émotions.

Je dédie ce travail :

A Mes chers parents

pour tous vos sacrifices, tendresse et votre soutien tout au long de mes études. A ceux qui m'ont appris à aimer et à être forte Maman et le meilleur père du monde

A toute ma famille

Chères tantes, oncles, Tous mes cousins et mes cousines surtout Meriem, Nibras, Hadjer, et radja Je vous dédie ce travail en témoignage de mon profond amour A mes sœurs RABAB SARA et ASMA vous êtes ma source de tendresse,à ma belle-sœur Mouna

A Mes chers frères : Ishak et Yacoub et mes beaux-frères Kacem Mouloud et Yazid.

A Mes adorables nièces Ranime Kaoutar Nour cine Sidra et ma jolie Ariame Darine

A mes neveux Anes Ahcen et Ayoub Ihab

A toute ma deuxième famille

Ma moitié Asma Mazoued

Mes partenaires de folie : Djouhaina, Dounia, Nahla et Hamida, et à mon âme sœur Imen merci d'être toujours là à mes côtés,

A mon ange,binômeet meilleure amie

Manel pour tous les moments qu'on a vécu ensemble la joie et la folie.

Je dédie ce travail à mes amies A tous ceux qui me sont chères A tous ceux qui m'aiment A tous ceux que j'aime

A la mémoire de mes grand père Alloua et Salah

».

Dédicace

Avec tout mon amour éternel et avec l'intensité de mes émotions.

Je dédie ce travail :

A Mon très cher père

L'homme qui a tellement sacrifié pour moi et qui mérite

toute ma reconnaissance Aucune dédicace ne saurait

être assez éloquente pour exprimer ce que vous méritez.

Ma très chère mère

Je n'oublie pas ses sacrifices : l'amour qu'elle m'a donné

et Pour ces encouragements, je te souhaite la joie et de

bonne santé.

A mon très cher petit frère Mohamed

Je te souhaite un avenir plein de joie, de bonheur, de réussite

Je t'exprime à travers ce travail mes sentiments de fraternité et d'amour.

A tous mes oncles et tantes

Et spécialement Sita, Fazila et mon oncle Toufik .

A mes très chères cousines

Bouchra, Assia, Safa, Malika, Lina, Dalia, Imene, Zineb,Rym,Ahlem ,Yanis, Amira et la petite sirine

En témoignage de mon amour éternel que Dieu vous garde,

Vous protège et vous offre une vie pleine de joie et de réussite.

A ma chère binôme et meilleure amie

« Rabaa » *et à toute sa famille*
Je te souhaite un avenir plein de joie, de bonheur et de réussite
Merci pour tous les moments qu'on a vécu ensemble, tu resteras toujours ma sœur.

A mes chères amies

Hamida et Asma Mezoued, Asma Alia et Yasmine …..

Que toute personne qui, d'une manière ou d'une autre, m'a aidée et encouragée pour l'aboutissement de ce travail, trouve ici l'expression de notre sincère reconnaissance

A la mémoire de ma grand-mère

».

Table des matières

Chapitre II

Techniques de compression

Chapitre III

Résultats et simulation

Liste des figures

Liste des tableaux

Liste des abréviations

BPP	**B**its **p**ar **p**ixel.
DCT	**D**iscret **C**osinus **T**ransform.
DWT	Discret Wavelet Transform.
DXF	**D**ata e**X**change **F**ormat.
EQM	MSE pour Mean Square Error.
JPEG	**J**oint **P**hotographic **E**xperts **G**roup.
WMF	**W**indows **M**eta **F**ile.
LZW	**L**empel-**Z**iv-**W**elch
LZ77	**L**empel et **Z**iv en 1977
LC	**L**uminance /**C**hrominance
NTSC	**N**ational **T**élévision **S**ystems **C**ommette
MPEG	**M**oving **P**icture **E**xperts **G**roup
MSE	**L**'erreur **M**oyenne **Q**uadratique
PAO	**P**ublication **A**ssistée par **O**rdinateur
PSNR	**p**eak-to-peak **S**ignal-to-**N**oise **R**adio.
RGB	**R**ed, **G**reen, **B**lue.
RLE	**R**un **L**ength Coding
TIFF	**T**ag **I**mage **F**ile **F**ormat.
FFT	(**F**ast **F**ourier **T**ransform)

Introduction générale

Vu l'importance des images et la quantité d'informations qu'elle peut générer, le monde s'intéresse de plus en plus à l'image et tend à mieux extraire ses informations et à les rendre plus faciles à utiliser dans notre vie. En effet, les images sont présentes dans des domaines très variés: les télécommunications, la médecine, la météorologie, la géologie, etc.

Vu le volume important d'information mis en œuvre lors de l'utilisation des images numériques. Il est indispensable de disposer d'outils performants pour la transmission et le stockage d'énormes quantités d'informations.

Pour surmonter ces contraintes, les chercheurs ont développé au cours des dernières décennies de nombreuses méthodes de compression de données dérivées de la théorie de l'information et impliquant de nombreux domaines dont les mathématiques et l'informatique. Ces méthodes de compression peuvent être classées selon le besoin de récupération parfaite ou non de l'information d'origine, il existe deux catégories principales.

La compression sans perte ou réversible de l'information présente l'avantage de conserver la qualité de l'image originale, mais avec un taux de compression relativement faible. La compression d'informations avec perte, qui est généralement basée sur une phase de transformation, est utilisée pour compacter les informations utiles en un nombre réduit de coefficients non nuls [1]. Cette classe de compression est suivie d'une phase de quantification, qui entraîne la perte d'information. Ce type de compression regroupe des algorithmes caractérisés par un taux de compression assez élevé tout en maintenant autant que possible une qualité acceptable de l'image originale [2].

Habituellement, une image en couleur est composée de trois couleurs primaires : le rouge, le vert et le bleu. Chaque pixel de l'image couleur a des proportions différentes de rouge, de vert et de bleu. Dans la plupart des cas, il existe une forte corrélation entre les composantes rouge, verte et bleue de l'image couleur. De nombreuses transformations ont été proposées dans la littérature scientifique pour exploiter au mieux cette corrélation entre les trois composantes. YIQ (luminance et chrominance)est une transformation largement acceptée pour différents standards de couleur. Le modèle YIQ est une réorganisation du

système RGB (rouge, vert, bleu) pour réduire toute corrélation entre les signaux de couleur: rouge, vert et bleu [3].

L'objectif principal de notre projet vise à étudier l'effet conjoint des transformations réversibles sur la performance de compression sans perte des images couleur. Il s'agit de la transformation en ondelettes réversible, d'une part, et de la transformation couleur réversible d'autre part. Pour cela, nous employons deux techniques du codage source (Huffman et Arithmétique),fréquemment utilisées en compression de données, afin d'effectuer une étude comparative entre plusieurs variantes, qui s'appliquent soit au domaine spatial soit au domaine transformé. La première consiste à appliquer les codeurs directement sur l'image couleur, et la seconde à appliquer les mêmes codeurs dans le domaine transformé en combinant deux transformations réversibles, dans ce cas, la transformée en ondelettes réversible et la transformation couleur réversible (Y'I'Q',$O_1O_2O_3$). L'objectif principal vise àréduire la corrélation et trouver l'espace de couleurs le plus efficace et le plus adapté à la compression.Une comparaison des résultats obtenus est ensuite effectuée en termes du débit (exprimé en Bpp). Afin de réaliser ces objectifs, le mémoire contient trois chapitres:

Chapitre I: réservé aux généralités sur les différentes images et leurs définitionsainsi qu'aux différents espaces de couleurs utilisées dans la simulation des résultats.

Chapitre II: consiste en un tour d'horizon portant sur les principales méthodes decompression d'images avec et sans perte ainsi que des notions générales telles que les mesuresde performance et de distorsion. A la fin de ce chapitre,on présente les variantes implémentées avec les notions nécessaires qui permettent une meilleure compréhension.

Chapitre III : débute par récapituler les résultats obtenus, et puis, par le biais de quelques critères objectifs, une analyse comparative est effectuée, et approfondie par une discussion des résultats observés.
Le mémoire s'achève par une conclusion générale.

Chapitre I

Notion générales sur l’image

I.1 Introduction

Ce chapitre sera consacré à aborder les notions de base du traitement de l'image et de l'information. La présentation de ces dernières, constitue une étape fondamentale, aidant à construire un pont logique, qui nous permettra de mieux comprendre ce qui suivra, et de faire un lien fiable et adéquat, entre ce chapitre et celui de la compression. Dans un premier temps, nous explorons les notions portant sur l'image, les espaces couleurs RGB, Y'I'Q' et $O_1O_2O_3$. Nous terminerons par la présentation des concepts et outils d'analyse d'une méthode de compression.

I.2 Multimédia

Le multimédia est une forme de communication qui combine différentes formes de contenu comme le texte, l'audio, les images, les animations ou la vidéo en une seule présentation, contrairement aux médias de masse traditionnels, tels que les documents imprimés ou les enregistrements audio. Les exemples les plus courants de multimédia sont les podcasts vidéo, les diaporamas audio, les spectacles animés et les films [4].

Le multimédia peut être enregistré pour être lu sur des ordinateurs, des portables, des smartphones et d'autres appareils électroniques, soit à la demande, soit en temps réel (streaming). Dans les premières années du multimédia, le terme "riche media" était synonyme de multimédia interactif. Au fil du temps, les extensions hyper médiatiques ont apporté le multimédia sur la toile mondiale [4].

I.3 Notion sur l'image

I.3.1 Définition de l'image

C'est une structure d'information (objet être vivant ou concept) affichée sur un écran et a une signification visuelle ou mentale. On peut citer deux types d'images.

- ✓ Les images fixes telles que les photographies, bandesdessinés, affiches, panneaux publicitaire.
- ✓ Les images animées ou mouvantes comme les films, émissions, reportages, et dessins animés.

I.3.2 Définition d'un pixel

Le pixel est le plus petit point de l'image Chaque pixel a une valeur numérique qui représente le niveau de gris ou de couleur selon la nature de l'image. Quand on parle de pixel on parle de l'image numérique Donc, une image numérique est une collection de pixels qui représentent chacun un point de l'image, codé par des valeurs numériques [5].

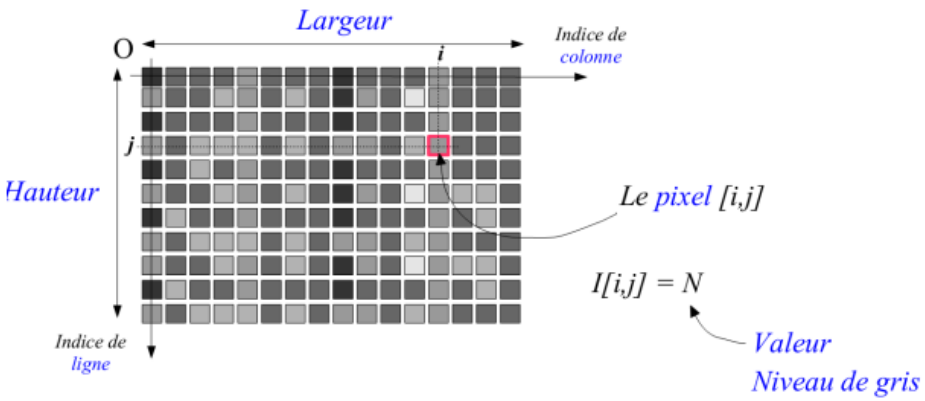


Figure I.1: Elément d'une image (le pixel)

I.3.2.1Dimension (définition) d'une image

Pour le calcul il suffit de multiplier le nombre de pixels sur la hauteur par le nombre de pixels sur la largeur de l'image.

I.4Différents types d'images

Plusieurs catégories d'images sont classées selon le nombre de bits sur le quel est codée la valeur de chaque pixel. On cite :

- ✓ Les images monochromes.
- ✓ Les images en niveau de gris.
- ✓ Les images couleurs.

I.4.1 Image monochrome (binaire)

Chaque pixel est codé sur un seul bit et peux prendre uniquement la valeur noir ou blanc, c'est aussi le plus simple type d'images et c'est celui que l'on utilise pour scanner du texte quand celui-ci est composé d'une seule couleur [6].

I.4.2 Image en niveaux de gris

C'est la valeur numérique qui reflète l'intensité de luminosité d'un point. Chaque pixel est représenté par un octet et non par un bit.

La couleur du pixel permet d'obtenir différentes valeurs de gris, afin d'afficher des images nuancées. Cette valeur peut être comprise par exemple entre 0 et 255 [6].

I.4.3Image en couleur

C'est une image ou chaque pixel est codé dans l'espace de couleur RGB donc c'est une représentation tridimensionnelle elle mesure également l'intensité et la chrominance de la lumière [7].

I.5 Différents formats d'images

Il existe deux grands types de format **:**

I.5.1 Images matricielles

L'image matricielle (ou « Image en mode point», ou en anglais un « Bitmap») est une image numérique dont les données sont stockées dans une matrice de points appelés « pixels ». Les images matricielles sont créées par les imprimantes, scanners, appareils photographiques et certains logiciels d'infographie comme Photoshop [8].

- **Principaux formats**

- JPG ou JPEG (Joint Photographique Experts Group) : créé par un consortium industriel, ce format très utilisé sur Internet, permet d'afficher les images en mode 16 millions de couleurs. Il est conçu pour réduire le plus possible la taille des fichiers graphiques en acceptant éventuellement de légères pertes de qualité. Il est destiné à la transmission rapide d'information. Ces résultats de compression sont extraordinaires [9].
- TIFF (Tag Image File Format) : c'est un format d'excellente qualité, mais qui présente des problèmes de compatibilité du fait d'une multiplicité de version. Il existe aussi une version compressée qui fournit des fichiers très compacts sans perte notoire de qualité. Ce format est compatible avec d'autres plates-formes (macintosh). Il est utilisé par les professionnels) [9].

- **Avantages**

- Simplicité de stockage en mémoire, puisqu'il suffit de coder la succession des valeurs de la matrice [8].
- Adaptable aux images complexes [5].

- **Inconvénients**

- La création d'une image à la main est difficile et il est conseillé de passer par un périphérique de numérisation : scanner, appareil photo-numérique... mais les retouches sont délicates [9].
- Les images bitmap sont « lourdes » : les fichiers, lorsque l'on traite des images en haute définition, ont des tailles qui varient entre 10 et 30 Mo par image.... Elles sont donc encombrantes, difficiles à faire passer sur le réseau, etc. [5] [6].

I.5.2Images vectorielles

Les données sont représentées par des formes géométriques simples qui sont décrites d'un point de vue mathématique. Par exemple, un cercle est décrit par une information du type(cercle, position du centre, rayon)[9][10]. Ces images sont essentiellement utilisées pour réaliser des

schémas ou des plans. Les logiciels de dessin industriel fonctionnent suivant ce principe. Les principaux logiciels de traitement de texte ou de PAO (publication assistée par ordinateur) proposent également de tels outils [5].

- **Principaux formats**

➢ WMF (Windows Meta File): c'est le format vectoriel utilisé par Windows, il est reconnu par beaucoup de programmes fonctionnant sous Windows. C'est un format qui permet de sauvegarder tant les imagesmatricielles que les images vectorielles [5].

➢ DXF (Data eXchange Format): format des produits Autocad et Autodesk.

- **Avantages**

➢ Les fichiers sont légers et occupent peu de place en mémoire [5].

➢ Adapté aux objets mathématiques et leur extension (voiture, avion...) [5].

- **Inconvénients**

➢ Inutilisables pour des images complexes, des photographies [5].

➢ Non reconnus par les navigateurs Internet et par certains logiciels multimédias.

➢ Un fichier vectoriel est plus fragile qu'un fichier matriciel dont l'en-tête surtout doit être intacte. La moindre dégradation de l'information est souvent irréparable [9].

I.6Différents systèmes de représentation

I.6.1Systèmes de primaires

I.6.1.1 Espace RGB

L'espace RGB (Red, Green, Blue, pour Rouge Vert Bleu, en français) est sans doute l'espace de couleurs le plus utilisé. Il consiste à représenter l'espace des couleurs à partir de trois rayonnements monochromatiques de couleurs (rouge, vert et bleu) [9][11].Cet espace de couleurs correspond à la façon dont les couleurs sont généralement codées informatiquement.

- **Inconvénients**

✓ Valeurs négative des composantes trichromatiques [12].

✓ Valeurs des composantes trichromatiques liées à luminance et corrélées.

I.6.2Système luminance-chrominance

Il existe plusieurs systèmes luminance-chrominances :

I.6.2.1Système de télévision

- **Origines**

✓ Les téléviseurs noir et blanc doivent pouvoir recevoir en noir et blanc les émissions en couleur [12].

✓ Les téléviseurs couleurs doivent pouvoir recevoir en noir et blancles émissions en noir et blanc[12].

✓ Nécessité de séparer l'information de chrominance del'information de luminance pour la transmission des signaux detélévision [12].

a. Espace YIQ

Cet espace de couleur est utilisé par le système de télévision couleur NTSC (National Television Systems Commette), au japon et aux Etats-Unis [13].

Ce modèle a été développé pour permettre la transmission efficace des couleurs tout en gardant la compatibilité avec la transmission pour les téléviseurs en noir et blanc. Il correspond à un changement de base du RGB [14].

Le principal avantage du modèle YIQ est que l'œil humain est le plus sensible à la composante (luminance), qui est celle représentée à la télévision en noir et blanc [14].

La plupart des techniques de compression des images couleur sont basées sur le modèle Y I Q. Cependant, cette transformation souffre d'un sérieux inconvénient pour les applications de compression sans perte : elle implique des opérations en virgule flottante nécessitant une quantification. Il en résulte une quantification et le système n'est plus sans perte [14].

L'espace couleur RGB peut être converti au format YIQ selon l'équation suivante :

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.275 & -0.321 \\ 0.212 & -0.523 & 0.311 \end{bmatrix} . \begin{bmatrix} R \\ G \\ B \end{bmatrix} \qquad \text{(I.1)}$$

Y : détermine l'intensité du canal de couleur Y. (Y représente l'information relative au composant de luminance/à l'échelle de gris)

I : détermine la phase du canal de couleur I. (I représente l'information relative au composant de teinte/à la colorimétrie)

Q : détermine la phase du canal de couleur Q. (Q représente l'information relative au composant de saturation/à la colorimétrie)

La Transformée inverse est donnée par :

$$\begin{pmatrix} R \\ G \\ B \end{pmatrix} = \begin{pmatrix} 1.000 & 0.956 & 0.621 \\ 1.000 & -0.272 & -0.647 \\ 1.000 & -1.0106 & 1.703 \end{pmatrix} \begin{pmatrix} Y \\ I \\ Q \end{pmatrix} \quad \text{(I.2)}$$

- **Transformation réversible Y'I'Q'**

Pour la compatibilité avec le matériel arithmétique entier, les coefficients Y I Q sont quantifiés (arrondis/tronqués), ce qui entraîne une erreur de quantification. Pour que le système soit sans perte, les coefficients Y I Q entiers et l'erreur de quantification doivent être comprimés. Afin d'éviter ces erreurs de quantification, la transformation de RGB vers Y'I'Q', qui est fondée sur une arithmétique entière, est présentée. Outre l'avantage de ne requérir que des additions d'entiers et des opérations de décalage, la transformation est totalement réversible et sans perte. La transformation directe est donnée par les équations suivantes [12].

$$Y' = \left\lfloor \frac{\left\lfloor \frac{R+B}{2} \right\rfloor + G}{2} \right\rfloor$$

$$I' = R - B \quad \text{(I.3)}$$

$$Q' = \left\lfloor \frac{R+B}{2} \right\rfloor - G$$

La transformée inverse est donnée par:

$$\begin{aligned} R &= Y + \left\lfloor \frac{Q+1}{2} \right\rfloor + \left\lfloor \frac{I+1}{2} \right\rfloor \\ G &= Y - \left\lfloor \frac{Q}{2} \right\rfloor \\ B &= Y + \left\lfloor \frac{Q+1}{2} \right\rfloor - \left\lfloor \frac{I}{2} \right\rfloor \end{aligned} \quad \text{(I.4)}$$

b. Espace $O_1O_2O_3$

L'espace $O_1O_2O_3$ est un espace de couleurs qui décompose la couleur en trois composantes, O_1c'est l'intensité ou la luminance, tandis que O_2et O_3, ensemble, représentent la chrominance à chaque pixel [15].Cette transformée ne provoque pas la perte de toute information, contrairement aux transformations de couleur conventionnelles telles que la conversion R G B. D'autres algorithmes de compression sans perte pour les images en échelle de gris peuvent remplacer le prédicateur. Le codage d'entropie est appliqué aux signaux. Une transformation réversible multidimensionnelle pour la compression d'images en échelle de gris. Dans la méthode proposée, nous appliquons une transformée réversible à la transformation de couleur en formulant 3 entrée 3 sortie réversible transformer. La transformation directe et la transformation inverse on le formulaire spécifique suivant :

$$\begin{aligned} O_1 &= \left\lfloor \frac{R+G+B}{3} + 0.5 \right\rfloor \\ O_2 &= \left\lfloor \frac{R-B}{2} + 0.5 \right\rfloor \\ O_3 &= B - 2G + R \end{aligned} \tag{I.5}$$

- **Transformée inverse**

$$\begin{aligned} R &= O_1 - O_2 + \left\lfloor \frac{O_3}{2} + 0.5 \right\rfloor - \left\lfloor \frac{O_3}{3} + 0.5 \right\rfloor \\ G &= O_1 - \left\lfloor \frac{O_3}{3} + 0.5 \right\rfloor \\ B &= O_1 + O_2 + O_3 - \left\lfloor \frac{O_3}{2} + 0.5 \right\rfloor - \left\lfloor \frac{O_3}{3} + 0.5 \right\rfloor \end{aligned} \tag{I.6}$$

I.6.3 Taux d'information (Entropie)

L'entropie est une grandeur qui caractérise la quantité d'information que contient une image. Par exemple, une image dont tous les pixels ont la même valeur contient très peu d'informations car elle est extrêmement redondante, donc son entropie est faible. En revanche une image dont tous les pixels ont une valeur aléatoire contient beaucoup d'informations, son entropie est forte [16] [5].

L'entropie (en bits) est calculée par la formule suivante [17]:

$$H = -\sum_{i=1}^{N} P_i \log_2 P_i \qquad \text{(I .6)}$$

I.6.4 Mesures de distorsion

La distorsion (D) est l'erreur introduite par l'opération de compression, due au fait qu'éventuellement l'image reconstruite n'est pas exactement identique à l'image originale[5]. La mesure de distorsion utilisée généralement en compression d'image est l'erreur quadratique moyenne EQM (ou MSE pour Mean Square Error).

Si l'on note Y (i, j), l'image originale de taille MxN,

Ỳ (i, j), l'image de même taille obtenue après reconstruction, nous pouvons définir :

L'erreur Moyenne Quadratique (MSE) :

$$\mathrm{MSE}^2 = \frac{\sum_{i=1}^{N}\sum_{j=1}^{M}(I(i,j)-\hat{I}(i,j))^2}{N \times M} \qquad \text{(I .7)}$$

I: L'image originale.

Î: L'image décodée.

M: le nombre de lignes de l'image.

N : le nombre de colonnes de l'image.

(i, j) : positionnement des pixels.

I.6.5 Rapport signal sur bruit (PSNR)

Le PSNR (peak-to-peak Signal-to-Noise Radio ou rapport signal à bruit en pic), qui se fonde sur l'erreur quadratique moyenne, est une mesure objective fréquemment utilisé pour évaluer la distorsion. Il se mesure en décibel (dB) et se définit par la formule:

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

$$\text{PSNR}=10.\log_{10}\frac{255^2\times 3}{MSE(R)+MSE(G)+MSE(B)} \qquad \text{(I .8)}$$

$$\textbf{PSNR=10}\ \log_{10}\left(\frac{(2^R-1)^2}{MSE}\right) dB(Décibels)$$

I.7. Conclusion

Ce chapitre a d'abord été consacré aux généralités sur l'imagerie. Dans un second temps, nous avons présenté des concepts et des outils liés aux domaines de l'image et de la compression de données. Enfin, nous avons défini différents formats des espaces couleur pour les implémenter dans les algorithmes de compression que nous utiliserons dans les prochains chapitres.

Chapitre II

Techniques de compression

Introduction

De nos jours, la représentation numérique des images est devenue un problème majeur, les différentes applications, telles que la télécopie, la vidéoconférence, l'imagerie médicale et satellitaire, la télévision haute définition et la télésurveillance, exigent sans cesse de très grands espaces de stockage de l'information et un débit de transmission très élevé, ces deux contraintes restent toujours en suspens malgré le développement de la technologie. La compression est donc un moyen incontournable pour faire face à ces contraintes. C'est une étape essentielle pour optimiser l'utilisation de ces grands volumes d'information dans les réseaux informatiques. Ce chapitre explore les notions de base de compression d'images (but, mesure de performance, compression sans et avec perte, techniques du codage sans perte, etc.). Dans la dernière section, nous exposons notre méthode de travail qui repose sur des transformations réversibles (en ondelettes et couleurs) qui traitent des entiers et retournent des entiers.

Compression d'image

La compression d'image est une application de la compression des données sur des images numériques. Cette compression a pour utilité de réduire la redondance des données d'une image afin de pouvoir l'emmagasiner sans occuper beaucoup d'espace ou la transmettre rapidement. La compression d'image peut être effectuée avec perte de données ou sans perte [3].

But de la compression d’image

La compression est une réduction du nombre de bits nécessaires pour représenter les images. Compresser les images permet d'optimiser la capacité de stockage et la vitesse de transfert des fichiers [16].

Mesures de performance de la compression d’images

a. Taux de compression

Le taux de compression donne une mesure de performance des méthodes de compression des images. Etant donné que l’objectif d’une compression est de minimiser la quantité d’informations nécessaire à la représentation d’une image, le taux de compression représente le rapport entre le nombre de bits utilisés par l'image originale et le nombre de bits utilisés par l'image compressée [17].

$$\tau = \frac{\text{nombre de bit utilisés pour représenter l'image originale}}{\text{nombre de bit utilisés par l'image compressée}} \quad \text{(II.1)}$$

b. Débit

Le débit binaire, qui est une autre mesure souvent utilisée, est déterminé à partir du nombre moyen de bits par pixel (Bpp) de l'image codée[18], il est défini comme suit:

$$\text{débit} = \frac{\text{nombre de bits par pixel dans l'image originale}}{\text{taux}} \quad \text{(II.2)}$$

c. Temps de calcul

La contrainte du temps est un facteur essentiel dans l'évaluation des performances de toute méthode de compression, elle revient à calculer le temps pris par la compression et la décompression des images. Cette contrainte est plus au moins imposée selon l'application visée par la compression (transmission ou archivage). En effet, il serait dommage, dans une application de transmission, que le temps gagné par une réduction de la taille des données à transmettre soit inférieur au temps passé à la compression / décompression. Cette qualité sera cependant moins cruciale dans des applications visant l'archivage de données [19].

Compression des données

La compression des données peut être définie comme étant un système dont l'entrée est une donnée sans compression et la sortie est un flux de données numériques relativement court représentant la donnée compressée. Le processus inverse est appelé décompression permettant la reconstruction de l'image à partir du flux de données numériques. Parfois, les systèmes de compression et de décompression ensembles sont appelés "Codec" (codage pour compression et décodage pour décompression) [20].

Méthodes de compression

On peut distinguer deux grandes familles d'algorithmes de compression, les méthodes dites sans perte ou réversibles garantissent la restitution parfaite des images, alors que les méthodes dites avec perte ou irréversibles modifient plus ou moins la valeur des pixels [5].

Compression sans perte(ou réversible)

La compression est dite sans perte lorsqu'il n'y a aucune perte des données sur l'information

D'origine. Il y a autant d'information après la compression qu'avant. Le but est de réduire la taille des images obtenues après la compression tout en ayant la possibilité de retrouver exactement l'image d'origine[14].Elle est utilisée dans des applications comme l'archivage des images médicales, l'imagerie satellitaire (le coût des images est élevé et les détails sont importants), les textes, les programmes et tout autre type de données nécessitant une conservation intégrale [5].

Il existe de nombreux types d'algorithmes de compression d'images sans perte. Voici les plus répandus.

Méthodes statistique

a. Codage de Shannon

Utilisé dans les années cinquante, le code de Shannon-Fano est le premier code à avoir exploité la redondance d'une source. Tous les symboles à compresser sont triés selon leur probabilité, et l'ensemble trié des symboles est coupé en deux parties de telle façon que les probabilités des deux parties soient le plus proche possible de l'égalité (la probabilité d'une partie étant égale à la somme des probabilités des différents symboles de cette partie)[21].

Tous les symboles de la première partie sont codés par un "0" suivi de leur code de Shannon-Fano en ne prenant en compte que les symboles de la première partie, et tous les symboles de la seconde partie sont codés par un "1" suivi de leur code de Shannon-Fano en ne prenant en compte que les symboles de la seconde partie, récursivement. Lorsqu'une partie ne contient qu'un seul symbole, celui-ci est représenté par un code vide (de longueur nulle).L'approche du codage de Shannon-Fano est descendante: l'algorithme part de l'ensemble des symboles et divise cet ensemble récursivement jusqu'à arriver à des parties ne contenant qu'un seul symbole [22].

L'inconvénient de cette approche est que, lorsqu'il n'est pas possible de séparer un ensemble de symboles et deux sous-ensembles de probabilités à peu près égales (c'est-à-dire lorsque l'un des sous-ensembles est beaucoup plus probable que l'autre), les codes produits ne sont pas optimaux [22].

b. Codage de Huffman

Le codage Huffman a été proposé par David Huffman en 1952.C'est une méthode statistique basée sur l'attribution d'un mot de code binaire pour chaque symbole de la chaîne à compresser.

La longueur des mots de code des symboles est variable. Les symboles ayant la probabilité d'apparition forte sont codés avec des séquences de bits plus courtes, tandis que les symboles dont la probabilité d'apparition est faible sont codés par des séquences plus longues. Le codeur de Huffman est un arbre binaire ordonné par tous les symboles et par leurs fréquences d'apparition. Les deux symboles les moins fréquents de la source sont reliés parleurs 'Parents' en faisant la somme de leurs fréquences. Les symboles prennent alors les valeurs "0" et "1"[3]. Le processus est répété sur l'ensemble des symboles jusqu'à ce qu'il ne reste qu'un seul symbole en formant la racine de l'arbre binaire. L'opération inverse est utilisée pour le décodage[23].

- **Algorithme du codage**

1. Les symboles sont triés et classés en fonction de leur fréquence.
2. A partir des deux symboles présentant la fréquence la plus faible, un nœud est créé. Il lui est affecté un poids égal à la somme des fréquences des deux symboles.
3. Le nœud crée remplace désormais les deux symboles dans la suite du processus. Aces derniers sont affectés respectivement les chiffres binaires "0" à la branche de gauche et "1" à la branche de droite.
4. La même démarche est reprise en considérant les deux symboles ou nœuds de poids le plus faible. Elle est renouvelée tant qu'il reste plus d'un nœud libre.

   Pour obtenir le code binaire de chaque symbole, on remonte l'arbre à partir de la racine jusqu'aux feuilles en rajoutant à chaque fois au code "0" ou un "1" selon la branche suivie. Il est intéressant de noter qu'on changeant la procédure de tri, on peut obtenir différents codes de Huffman pour une même source. En général, on cherche à établir le code de Huffman à variance minimale. Pour obtenir ce dernier, on doit toujours placer la nouvelle combinaison de lettres le plus haut possible dans la liste triée. Par ailleurs, on distingue des variantes de l'algorithme de Huffman telle que: le codage statique le codage semi-adaptatif et le codage adaptatif [24].

Pour une source X d'entropie H(X). La longueur moyenne L d'un mot de code obtenu par le codage de Huffman vérifie la relation suivante [24]**:**

$$\mathbf{H}(\mathbf{X}) \leq \mathbf{L} < H(\mathbf{X}) + \mathbf{1} \qquad (\text{II. } 3)$$

- **Exemple du codage**

Une source émet des lettres d'un alphabet A= {a_1, a2, a_3, a_4} avec les probabilités P (a_1)=0.1, P (a_2)=0.3, P (a_3)=0.25 et P (a_4) =0.3.

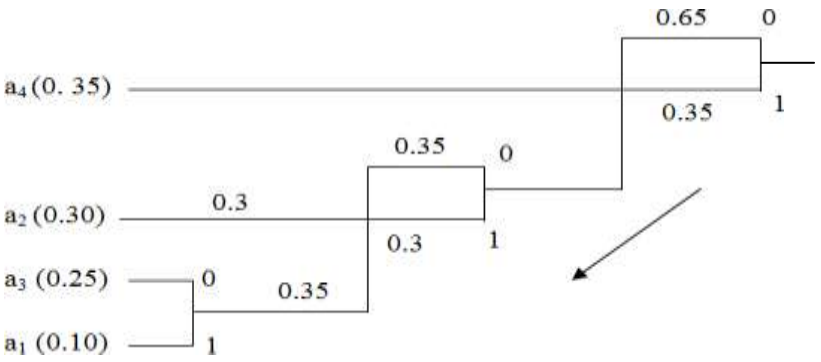


Figure II.1 : Algorithme de Huffman

Le code des symboles est: a_1= 001, a_3= 000, a2= 01, a_4= 1

- **Avantages et inconvénients du codage**

Le codage de Huffman offre une compression caractère par caractère optimale, Il a aussi l'avantage d'être facile à implanter par programmation, et le temps d'exécution est plutôt rapide. Toutefois, certaines méthodes de compression qui encode des mots plutôt que des caractères peuvent offrir un taux de compression plus intéressant. Ainsi, le codage de Huffman est souvent utilisé en conjonction avec d'autres techniques de compression comme par exemple: EZW [3].

c. Codage Arithmétique

Le codage arithmétique est un codage utilisant un modèle statistique, tout comme le codeur de Huffman [5]. Contrairement à ce dernier, il produit un code pour de symboles tout entière, et non pas un code par symbole. Chaque nouveau symbole lu modifie de façon incrémentale le code de sortie. Ce code de sortie est un nombre à virgule flottante compris entre 0 et 1, dont le nombre de chiffres après la virgule correspond au nombre de symboles. Contrairement à Huffman, il n'est pas obligatoire que chaque code ait un nombre entier de bits. Le codeur arithmétique est plus performant que le codeur de Huffman, mais il est plus complexe à implémenter [5].Décrivons brièvement ci-dessous l'algorithme de codage arithmétique dans le but d'en illustrer le principe, sachant que le décodage opère de manière inverse [3]:

- Calculer la probabilité associée à chaque symbole dans la chaîne à coder
- Associer à chaque symbole un sous intervalle proportionnel à sa probabilité, dans l'intervalle [0,1[(l'ordre de rangement des intervalles sera mémorisé car nécessaire au décodeur).
- Initialiser la limite inférieure de l'intervalle de travail à la valeur "0" et la limite supérieure à la valeur "1".

- Tant qu'il reste un symbole dans la chaîne à coder:
  - Largeur = limite supérieure - limite inférieure
  - Limite inférieure = limite inférieure + largeur x (limite basse du sous intervalle du symbole)
  - Limite supérieure = limite inférieure + largeur x (limite haute du sous intervalle du symbole).

A la fin, n'importe quelle valeur de l'intervalle final représente d'une manière unique la séquence d'entrée. Généralement, on choisit comme représentant de la séquence la limite inférieure de l'intervalle, ou bien on peut choisir la valeur moyenne. Il est à noter qu'à la fin du processus de codage, le codeur arithmétique génère un fichier composé d'un en-tête suivi de la représentation binaire du code choisi. De plus, l'en-tête peut contenir soit la table des probabilités, soit la table des fréquences de tous les symboles [1].

Le principal inconvénient de l'algorithme réside dans sa complexité d'implémentation.

Méthode par répétition

a. Codage RLE

Le principe de compression RLE est assez simple à mettre en œuvre. Il repose sur le fait que dans une image, il existe de nombreuses répétitions d'un même pixel, ou d'une même séquence de pixels, tous juxtaposés [25]. Ainsi, au lieu de coder chaque pixel d'une image, le RLE regroupe les 36 valeurs voisines identiques et ne transmet une valeur qu'une seule fois, précédée par le nombre de répétitions.

Il est clair que cette approche fonctionne bien s'il y a beaucoup de répétitions dans l'image. Cet algorithme très simple, peut aboutir à des taux de compression plus élevés [1]. Ce type d'encodage est principalement utilisé par des formats d'images comme BMP ou PCX, ainsi que dans la numérisation d'images en noir et blanc [26]. Un exemple du codage RLE est illustré sur la figure II.2.

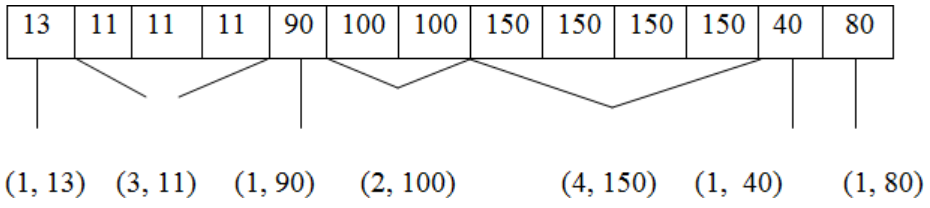


Figure II.2: Un exemple de codage par plage RLE

- **Avantages et inconvénients**

Le principal avantage de l'encodage RLE est son immense simplicité, son algorithme tient sur quelques lignes à peine autant au niveau de la compression que de la décompression. Il peut de plus être utilisé sur n'importe quel type de fichier ou après d'autres algorithmes décompression plus évolués pour gagner quelques octets de plus. En revanche, l'encodage RLE possède un désavantage de taille. Le fichier ou texte à compresser doit contenir plusieurs chaînes de caractères ou bits répétés pour être d'une quelconque utilité [26].

Méthode à base du dictionnaire

a. Codage (LZW)

Ce sont Abraham Lempel et Jakob Ziv qui ont inventé, en 1977, le premier algorithme de compression sous le nom de Lempel-Ziv-77 ou LZ77. LZ77 connut rapidement une nouvelle version :LZ78.

C'est une technique de codage à base d'un dictionnaire où nous mémorisons les chaînes qui se répètent dans ce dictionnaire. Ensuite, on remplace les chaînes mémorisées par leur adresse (ou indice) construite dans le dictionnaire [27]. L'élaboration du dictionnaire ainsi que la recherche de chaîne répétée sont différentes selon la version de l'algorithme.

Le premier algorithme de ce type a été mis au point par Lempel et Ziv en 1977 [28] et a donné lieu au programme LZ77. Celui-ci lit un flot de symboles et cherche des chaînes équivalentes dans une fenêtre de 4Ko précédant le flot d'entrée. Les équivalences sont remplacées par des codes. Les formats ZIP, ARJ et LHA basent leur compression sur cet algorithme [26].

Lempel et Ziv ont développé une nouvelle version de leur algorithme en 1978, LZ78, où le dictionnaire est construit à partir de tous les symboles précédemment rencontrés et non par une fenêtre, il est spécialisé dans la compression d'images et de fichiers binaires [26]. Puis en 1984, Terry Welch a modifié le format LZ78 pour pouvoir l'utiliser dans les contrôleurs de disques durs, le format LZW est né [26]. Le dictionnaire dans ce cas est initialement construit et contient l'ensemble des codes ASCII. Il est élaboré au fur et à mesure. Ce qui permet de changer la taille du dictionnaire au cours du codage [29]. Le codage LZW est une technique de compression réversible qui peut être appliquée à tout type de fichier de données, que ce soit: texte, image, fichier informatique, etc. Elle a été adoptée pour la mise en œuvre du format de compression d'images 'GIF' [30].L'Algorithme suivant montre son mécanisme :

Compression avec pertes (ou irréversible)

Ce type comporte une perte de données pendant le processus de codage/décodage. Le résultat qu'on peut en obtenir est une version altérée de l'image originale. Le but de ce type de compression est d'éliminer le plus d'information possible sans atténuer la qualité de l'image perçue par le système visuel humain [30].La compression avec pertes ne s'applique qu'aux données « perceptuelles », en général sonores ou visuelles il est impossible de retrouver les données d'origine après une telle compression. La compression avec perte est pour cela parfois appelée compression irréversible ou non conservatrice. La quantification est un des mécanismes utilisé dans les algorithmes de compression, qui produit des pertes d'information [31].

Méthodes spatiales et méthodes par transformation

Cette méthode s'intéresse au domaine dans lequel s'effectuent les opérations de base de la compression. Une image peut être représentée de deux façons strictement équivalentes:

- **Le domaine spatial**

Dans lequel l'image est représentée sous forme de pixels. C'est le domaine accessible visuellement à l'observateur [1].

- **Le domaine fréquentiel**

Dans lequel l'image est représentée sous forme de coefficients de fréquences. Le passage d'un domaine à l'autre se fait par des transformations mathématiques(en général linéaire) [1]

Dans le domaine spatial, l'information contenue dans l'image est distribuée sur toute la matrice image. Mais dans le domaine de fréquences, l'information est généralement plus concentrée.

a. Principe général d'un système de compression par transformée

La compression d'images par la méthode des transformées nécessite trois étapes de base, comme l'illustre la figure II.3. La première étape est la transformation des données de l'image pour obtenir des données moins corrélées. La deuxième étape est la quantification ou le seuillage, c'est dans cette étape que la perte d'information se produit.

Finalement le codage, les données quantifiées sont réduites par codage pour fin de transmission ou d'archivage. Dans la phase de décompression ou de reconstruction de l'image, trois étapes sont nécessaires. Les données transmises ou archivées sont décodées. La quantification inverse est appliquée au résultat du décodage. Puis, la transformation inverse est appliquée aux données du bloc [32].

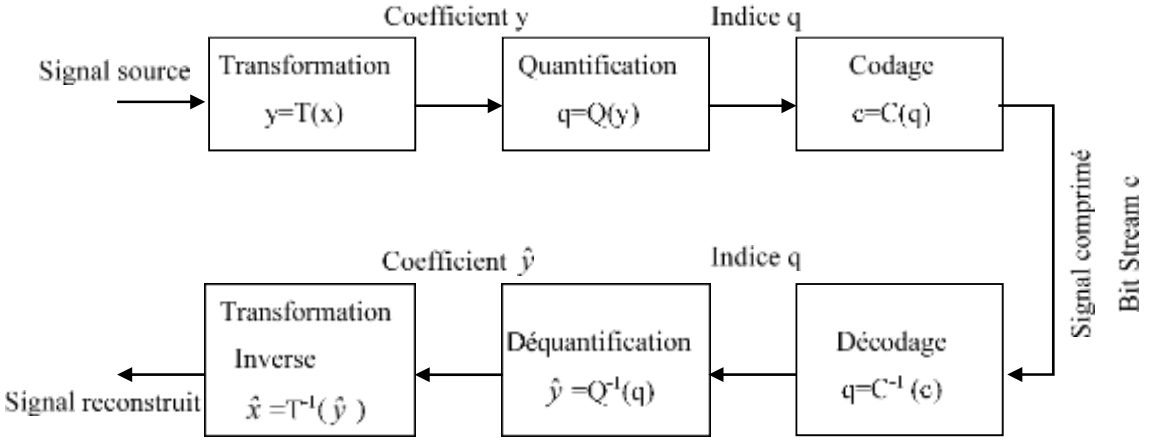


Figure II.3 Schéma typique de la compression/décompression par transformation (codec)

- **Transformation**

Dans ces méthodes, l'image de dimension NxN est subdivisée en sous images ou blocs détaille réduite. Chaque bloc subit une transformation mathématique orthogonale inversible linéaire du domaine spatial vers le domaine fréquentiel, indépendamment des autres blocs (transformée en un ensemble de coefficients plus ou moins indépendants).Les coefficients obtenus sont alors quantifiés et codés en vue de leur transmission ou de leur stockage. Pour retrouver l'intensité des pixels initiaux, on applique sur ces coefficients la transformation inverse [5]. L'objectif de ces transformations est double [5]: il s'agit de:

- ✓ Décorréler les données, c'est-à-dire d'obtenir des coefficients transformés moins corrélés que les pixels de l'image [5].
- ✓ Concentrer l'énergie sur un nombre réduit de coefficient, les coefficients ayant une valeur plus importante aux basses fréquences qu'aux hautes fréquences [5].

Dans ce but, plusieurs transformations linéaires ont été proposées dans la littérature scientifique [6]:

- Transformation de Fourier discrète (TFD).
- Transformation en cosinus discrète (TCD).

- **Quantification**

En général, la quantification apparaît en deuxième lieu dans un processus de compression, pour réduire la quantité d'information, et dégrade le signal de manière souvent irréversible [33]. Ainsi, une quantification est utilisée pour simplifier la représentation, tout en préservant l'information la plus pertinente. On remplace ainsi les valeurs initiales par un ensemble fini d'éléments qui donneront des résultats acceptables lors de la phase de décompression [5].

- **Codage**

Le codage est utilisé dans une chaîne de compression sans en général après l'étape de transformation. Cette étape cherche à éliminer toute redondance de telle façon que la quantité d'information à transmettre sois minimale, et permet de générer un message binaire (Bit Stream ou flot binaire) à partir duquel le décodeur sera en mesure de reconstruire soit une version approchée de l'image originale s'il s'agit d'une compression avec perte ou l'image originale s'il s'agit d'une compression sans perte.

b. Les transformées discrètes

Les méthodes par transformées figurent parmi les techniques de compression les plus employées. Elles n'agissent pas directement sur l'image numérique dans sa représentation canonique, mais sur le domaine de sa transformée. Elles permettent d'obtenir des taux de compression élevés tout en conservant une bonne qualité d'image.

Malgré qu'il existe un nombre considérable de transformées discrètes dans la littérature, on trouve que les transformées les plus utilisées dans le domaine de compression d'images sont la DCT et les DWT (Discret Wavelet Transform). Les ondelettes sont considérées dans le standard de compression JPEG 2000 [34], et la DCT dans les standards JPEG [35] et MPEG [36] (Moving Picture Experts Group) pour la compression des images fixes et animées, respectivement.

La popularité remarquable de la DCT est due essentiellement à sa capacité de compactage d'énergie (presque optimale) et à l'existence des algorithmes pour son calcul rapide et efficace.

a. Transformée de Fourier

Elle permet simplement de passer du domaine spatial au domaine fréquentiel [5]. Cette transformée rend donc visible les composantes en fréquence de l'image Il est intéressant de noter que nous pouvons revenir au domaine spatial via la transformée de fourrier inverse [5]. Une autre façon d'effectuer ce calcul permet de limiter considérablement la durée de cette transformation. C'est ce que l'on appelle la FFT (Fast Fourier Transform) [5].

b. Transformation en cosinus discrète DCT

La transformée en cosinus discrète est une transformation mathématique complexe qui a pour but de transformer le domaine de représentation de nos données, c'est-à-dire de passer d'un domaine spatial à un domaine fréquentiel. Notre œil est moins sensible à certaines données de l'image, le but de la transformée en cosinus discrète est de passer dans un domaine fréquentiel, ce qui nous permettra de trier efficacement l'ensemble des données de l'image et ainsi supprimer certaines données où l'œil humain ne verra que très peu de différences, ce qui revient à supprimer les hautes fréquences de l'image tout en gardant les données majeures qui sont représentées par les basses fréquences [7].

Durant la dernière décennie, la DCT (Discrète Cosine Transform) est émergée comme transformation d'image dans la plupart des systèmes visuels. La DCT a été largement déployé par les normes visuelles modernes de codage, par exemple, le MPEG [7].

Comme d'autre transformation, la DCT tente de dé corréler les données de l'image. Après la décorrélation chaque coefficient peut être codé indépendamment, sans perdre l'efficacité de la compression [7].

L'équation de la DCT est donnée comme suit :

$$DCT(i,j) = \frac{1}{2\sqrt{2N}} C(i)\, C(j) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} pixel(x,y) \cos\left(\frac{(2x+1)\, i\pi}{2N}\right) \cos\left(\frac{(2y+1)\, j\pi}{2N}\right) \qquad (II.3)$$

- N : la largeur d'un bloc, ici N=8.
- i,j : les indices d'un coefficient de la DCT dans un bloc.
- x, y : les indices d'un pixel de l'image dans un bloc.
- DCT(i,j) : la valeur d'un coefficient dans un bloc.
- $C(x) = \frac{1}{\sqrt{2N}}$ si $x = 0$; $C(x) = 1$ si $x \neq 0$.

DCT (i, j) le coefficient repéré par la ligne i et la colonne j dans la matrice DCT [5].

La DCT est effectuée sur une matrice carrée NxN de valeurs de pixels en donnant une matrice carrée NxN de coefficients de fréquence. Le temps de calcul requis pour chaque élément dans la DCT dépend de la taille de la matrice [5].

❖ **Différence entre la FFT et la DCT**

- La DCT est beaucoup plus simple en termes de coup de programmation [5].
- La DCT est efficace dans la compression de multimédia (cas de JPEG) [5].
- Beaucoup plus utilisée [5].

c. La compression par ondelettes

Les ondelettes c'est d'abord une théorie mathématique d'analyse du signal, développée dans les années 90. On peut considérer qu'il s'agit d'une extension de l'analyse de Fourier [37]. La technologie de compression à base d'ondelettes offre une plus grande finesse au niveau de l'analyse du signal, et permet de mieux s'adapter aux propriétés locales de l'image. La transformation par ondelettes est une technique de compression d'image fixe très performante [38].

Passons maintenant à l'algorithme pyramidal utilisé. La décomposition en coefficients d'ondelettes n'utilise pas une fonction de moyenne, mais s'appuie sur deux filtres. Un filtre passe bas (H) et un filtre passe haut (L). La combinaison de ces filtres permet d'obtenir quatre sous images HH, HL, LH et LL, comme le montre le figure II.4. Ces filtres sont nommés miroirs en quadratures [5].

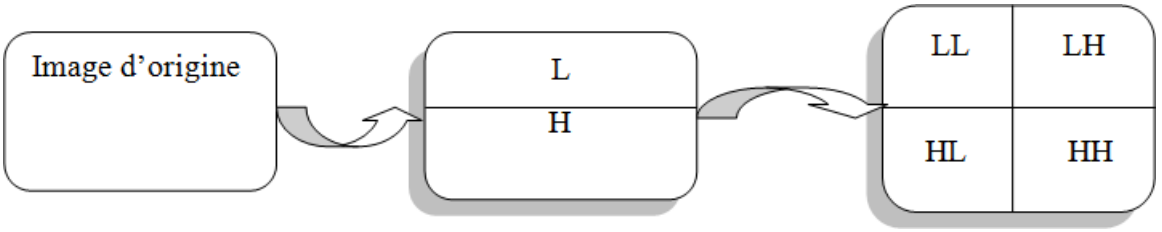


Figure II.4 : Transformation en colonnes et en lignes.

Chacune des quatre images obtenues par la transformation représente des informations bien distinctes.

Les étapes de compression par ondelettes sont usuellement :

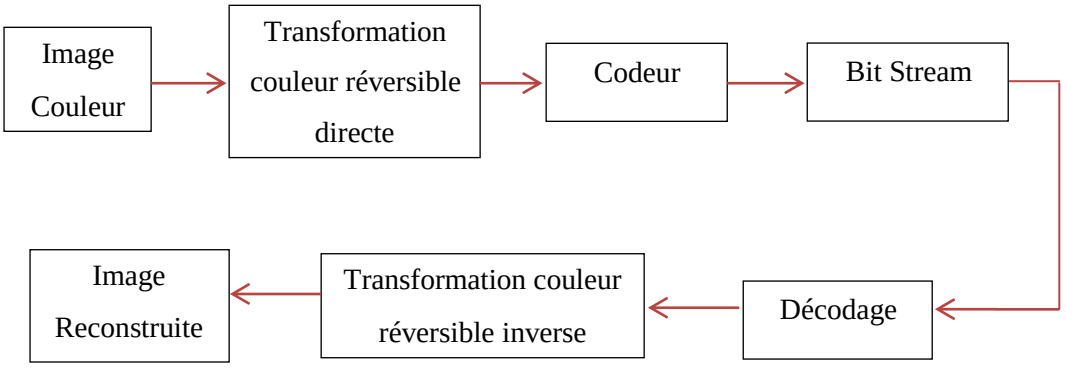
- Transformation par ondelettes.
- Quantification : les valeurs des coefficients de détails inférieurs à un certain niveau sont éliminées, en fonction de l'efficacité recherchée. C'est cette étape qui introduit des pertes.
- Codage des restantes : les données restantes sont transmises à un encodeur entropie, c'est à dire à un algorithme de compression de données (LZW, HUFFMAN, RLE, …) [1].

Présentation et application de notre méthode

Dans cette section, nous présentons notre approche de compression et décompression d'images. Les figures si dessous résument la structure de ces algorithmes ainsi que les étapes à suivre pour l'implémentation de nos codeurs (Huffman et Arithmétique) dans deux domaines différents: spatial et transformé [39], en appliquant un changement d'espace de couleurs RGB vers d'autres espaces, tels que Y'I'Q' et $O_1O_2O_3$. Ceci est dans l'optique de trouver l'algorithme de compression le plus performant et l'espace de couleurs le plus adapté à la compression des images couleur.

Codage appliqué dans le domaine spatial

Le schéma de compression/décompression sans perte des images couleur dans le domaine spatial est présenté dans la figure suivante :



FigureII.5 : Schéma de compression/décompression sans perte dans le domaine spatial.

a. Codage de Huffman

L'implémentation de notre algorithme de compression et décompression pour le codage de Huffman s'énonce comme suit :

Algorithme de compression:

Etape 1: lire l'image à compresser.

Etape 2: appliquer la transformation couleur réversible directe.

Etape 3: transformer les composantes de l'image couleur en vecteurs.

Etape 4: calculer l'Alphabet, les fréquences et les probabilités.

Etape 5: appliquer le codage de Huffman à chaque composante de l'image couleur.

Algorithme de décompression:

Etape 1: appliquer le décodage de Huffman à chaque composante de l'image couleur.

Etape 2: transformer les vecteurs en matrices.

Etape 3: appliquer la transformé couleur réversible inverse.

Etape 4: calculer l'entropie et le débit en Bpp (Bits par pixel).

Etape 5: afficher la taille de l'image originale et de l'image compressée en Bits.

Etape 6: afficher l'entropie et le débit.

b. Codage Arithmétique

L'algorithme faisant appel au codage Arithmétique se résume comme suit :

Algorithme de compression

Etape 1: lire l'image à compresser.

Etape 2: appliquer la transformation couleur réversible directe.

Etape 3: transformer les composantes de l'image couleur en vecteurs.

Etape 4: calculer l'Alphabet, les fréquences et les probabilités.

Etape 5: appliquer le codage Arithmétique à chaque composante de l'image couleur.

Algorithme de décompression

Etape 1: appliquer le décodage Arithmétique à chaque composante de l'image couleur.

Etape 2: transformer les vecteurs en matrices.

Etape 3: appliquer la transformé couleur réversible inverse.

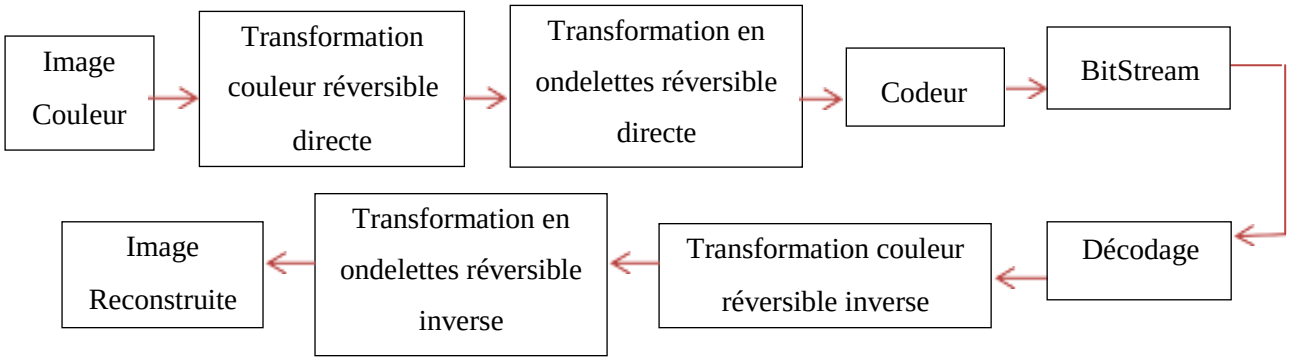
Etape 4: calculer l'entropie et le débit en Bpp (Bits par pixel).

Etape 5: afficher la taille de l'image originale et de l'image compressée en Bits.

Etape 6: afficher l'entropie et le débit.

Codage appliqué dans le domaine transformé

Le schéma de compression/décompression sans perte des images couleur dans le domaine transformé est illustré dans la figure suivante :



FigureII.6 : Schéma de compression/décompression sans perte dans le domaine transformé.

a. Codage de Huffman avec la transformée en ondelettes réversible (TOR)

L'algorithme du codage de Huffman basée sur une étape de transformation réversible s'établit comme suit :

Algorithme de compression

Etape 1: lire l'image à compresser.

Etape 2: appliquer la transformation couleur réversible directe.

Etape 3: appliquer la transformation en ondelettes réversible directe.

Etape 4: transformer les composantes de l'image couleur en vecteurs.

Etape 5: calculer l'Alphabet, les fréquences et les probabilités.

Etape 6: appliquer le codage de Huffman à chaque composante de l'image couleur.

Algorithme de décompression

Etape 1: appliquer le décodage de Huffman à chaque composante de l'image couleur.

Etape 2: transformer les vecteurs en matrices.

Etape 3: appliquer la transformé en ondelettes réversible inverse.

Etape 4: appliquer la transformé couleur réversible inverse.

Etape 5: calculer l'entropie et le débit en Bpp (Bits par pixel).

Etape 6: afficher la taille de l'image originale et de l'image compressée en Bits.

Etape 7: afficher l'entropie et le débit.

b. Codage Arithmétique avec la transformée en ondelettes réversible(TOR)

L'algorithme du codage Arithmétique basée sur une étape de transformation réversible s'établit comme suit :

Algorithme de compression

Etape 1: lire l'image à compresser.

Etape 2: appliquer la transformation couleur réversible directe.

Etape 3: appliquer la transformation en ondelettes réversible directe.

Etape 4: transformer les composantes de l'image couleur en vecteurs.

Etape 5: calculer l'Alphabet, les fréquences et les probabilités.

Etape 6: appliquer le codage Arithmétique à chaque composante de l'image couleur.

Algorithme de décompression

Etape 1: appliquer le décodage Arithmétique à chaque composante de l'image couleur.

Etape 2: transformer les vecteurs en matrices.

Etape 3: appliquer la transformé en ondelettes réversible inverse.

Etape 4: appliquer la transformé couleur réversible inverse.

Etape 5: calculer l'entropie et le débit en Bpp (Bits par pixel).

Etape 6: afficher la taille de l'image originale et de l'image compressée en Bits.

Etape 7: afficher l'entropie et le débit.

Conclusion

La compression des données est appelée à prendre un rôle encore plus important en raison du développement des réseaux et du multimédia. Nous avons abordé dans ce chapitre les concepts généraux de compression d'images fixes, les étapes principaux des algorithmes de compression avec et sans perte avec les différents méthodes existantes (Huffman, LZW, Arithmétique, DCT, etc.). Le chapitre s'est terminé par la présentation de notre méthode de travail qui est basée sur l'implémentation des deux algorithmes (Huffman et Arithmétique) dans le domaine spatial et transformé en se basant sur des transformations réversibles à coefficients entiers (en ondelettes et couleur).

Chapitre III

Résultats et simulation

III.1 Introduction

L'objectif de ce dernier chapitre est d'évaluer les performances de compression sans perte de deux techniques de codage de source, citées déjà dessus, dans le chapitre précédent. En utilisant trois espaces de couleurs différents afin d'effectuer une étude comparative pour trouver l'espace de couleurs le plus adapté à la compression des images A ce propos, la démarche suivie consiste à comparer les résultats des quatre algorithmes obtenus au précédent chapitre, avec un ensemble d'image de test en termes de débits et de l'entropie.

Dans un premier lieu, nous appliquons les deux premiers algorithmes Huff et Arith, pour RGB, Y'I'Q' et $O_1O_2O_3$ directement sur l'image à compresser, Alors que dans le second, l'application des deux autres algorithmes Huff_TOR et Arith_TOR se fait particulièrement sur l'image transformée, par le biais d'une transformation en ondelette réversible a coefficient entier, afin d'assurer la compression sans perte.

III.2 Environnement de travail

- Edition windows: Windows 7 ÉditionIntégrale.
- Type de système: Système d'exploitation 32 bits.
- Processeur: Intel (R) Core (TM) i5-4300M CPU @ 2.60GHz 2.59GHz.
- Mémoire installée (RAM): 4.00 Go.

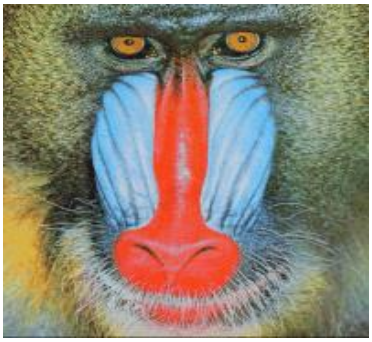
Expérience de travail dans Matlab R2015b.

III.3 Base d'image utilisée

Le choix des images de test est un problème fondamental lors de l'évaluation des systèmes de Compression, pour notre cas on a choisi des images couleurs à trois composantes :

1. **4.1.06 (**Tree) : dimension 256 x 256, taille 192Ko.
2. **4.2.03** (Mandrill) : dimension512 x 512, taille 768Ko.
3. **4.2.05** (Airplane) :dimension512 x 512, taille 768Ko.
4. **4.2.06 (**Sailboat on lake) :dimension512 x 512, taille 768Ko.
5. **4.2.07** (Peppers) :dimension512 x 512, taille 768Ko.

4.1.06 (Tree)



4.2.03 (Mandrill)

4.2.05 (Airplane)

4.2.06 (Sailboat on lake)

4.2.07 (Peppers)

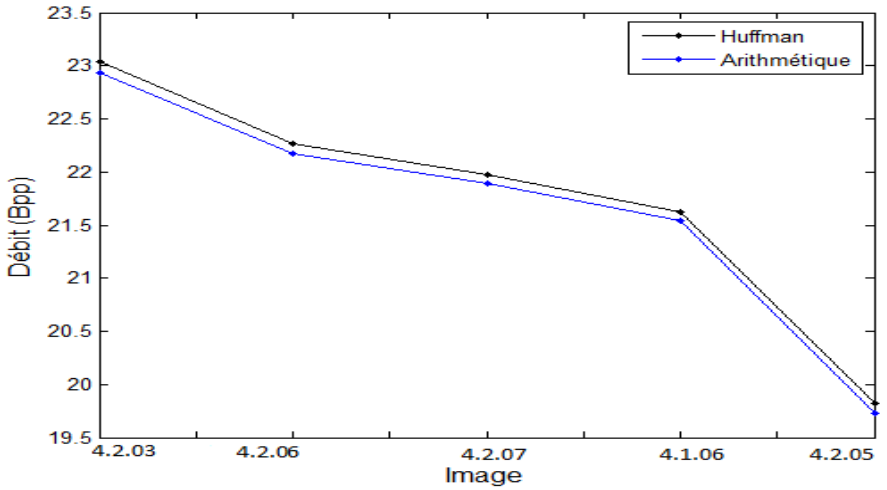
FigureIII.1 : Les différentes images originales.

III.4 Comparaison des performances des deux codeurs Huffman et Arithmétique pour les espaces de couleurs (RGB, Y'I'Q', $O_1O_2O_3$)

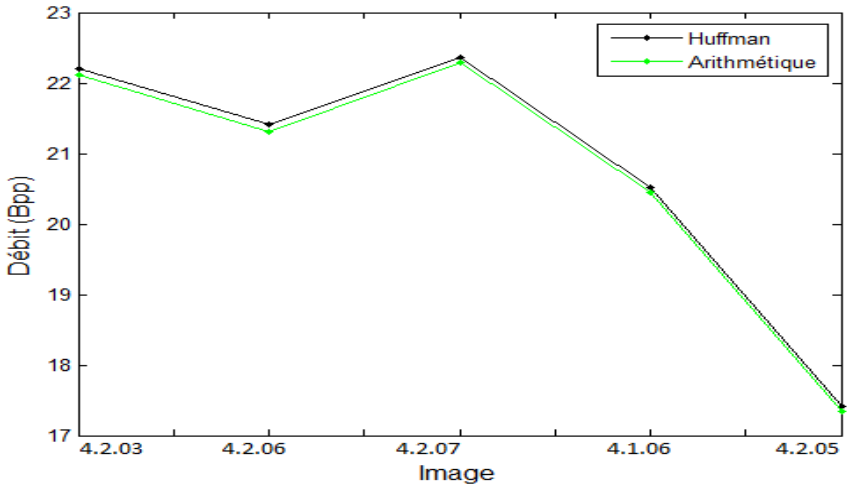
Dans le cadre d'évaluation de performances de compression sans pertes des techniques de codages décrites dans le chapitre précédant Huffman et Arithmétique en se basant sur Les méthodes de réduction de redondances qui peuvent être classées d'après leur domaine de réalisation (spatial ou transformé).

Pour évaluer les performances de notre méthode, les algorithmes ont été appliqué sur 5 images de tests (4.1.06, 4.2.03, 4.2.05,4.2.06.4.2.07) Les images que nous avons choisies sont des images standards utilisées par toute la communauté scientifique travaillant dans ce domaine. De même, ce sont des images avec des caractéristiques relativement différentes et de différentes tailles : 256 x 256 et 512 x 512 (8 Bits par pixels). Dans cette étape, nous choisissons deux autres espaces de couleurs réversible (Y'I'Q'et $O_1O_2O_3$) or que RGB dans le but de trouver l'espace de couleurs le plus adéquat a la compression.

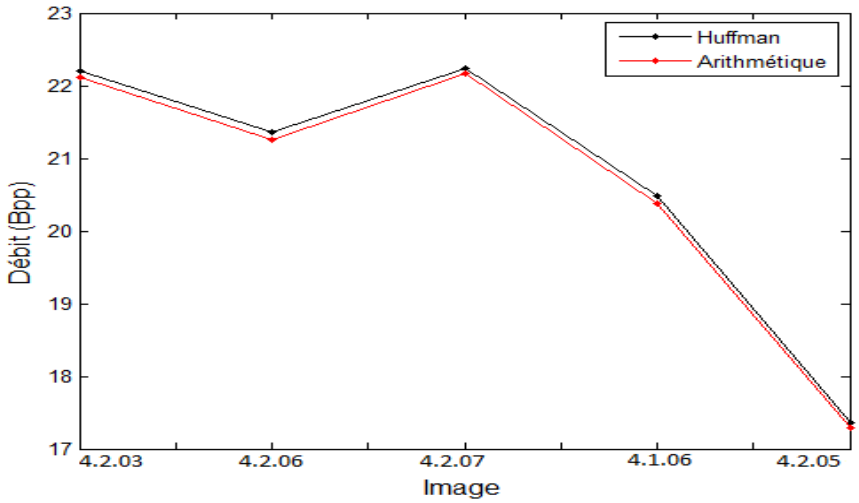
Le Tableau [III.1] ci-dessous résume les résultats obtenus à partir des algorithmes mis en œuvre et compare l'effet du changement de l'espace de couleurs dans le domaine spatial.



(a)



(b)



(c)

Figure III.2 : Evaluation de la performance de compression sans perte des codeurs Huffman et Arithmétique en utilisant les formats : (a) RGB. (b) Y'I'Q'. (c) $O_1O_2O_3$.

	Format RGB			Format Y'I'Q'			Format $O_1O_2O_3$		
	/	Codage de Huffman	Codage Arithmétique	/	Codage de Huffman	Codage Arithmétique	/	Codage de Huffman	Codage Arithmétique
Image	**Entropie**	**Débit**	**Débit**	**Entropie**	**Débit**	**Débit**	**Entropie**	**Débit**	**Débit**
4.1.06	21.5448	21.6251	21.5456	20.4388	20.5261	20.4397	20.3881	20.4795	20.3889
4.2.03	22.9333	23.0374	22.9335	22.1184	22.2087	22.1186	22.1148	22.2066	22.1150
4.2.05	19.7305	19.8220	19.7307	17.3410	17.4194	17.3412	17.2859	17.3521	17.2861
4.2.06	22.1689	22.2642	22.1691	21.3008	21.4066	21.3010	21.2477	21.3529	21.2479
4.2.07	21.8934	21.9683	21.8936	22.2854	22.3525	22.2856	22.1577	22.2294	22.1579
Moyenne	21.6542	21.7434	21.6545	20.6969	20.7827	20.6972	20.6389	20.7241	20.6391

Tableau III.1 : Comparaison des performances de compression sans perte des images couleur pour les codeurs Huffman et Arithmétique en utilisant 3 espaces couleur.

		Format					
		RGB		Y'I'Q'		$O_1O_2O_3$	
		Codage de Huffman	Codage Arithmétique	Codage de Huffman	Codage Arithmétique	Codage de Huffman	Codage Arithmétique
Image	**Taille de l'image originale**	**Taille de l'image compressée**	**Taille de l'image compressée**	**Taille de l'image compressée**	**Taille de l'image compressée**	**Taille de l'image compressée**	**Taille de l'image Compressée**
4.1.06	1572864 Bits	1417224 Bits	1412012 Bits	1435201 Bits	1339533 Bits	1342146 Bits	1336206 Bits
4.2.03	6291456 Bits	6039113 Bits	6011890 Bits	5821878 Bits	5798253 Bits	5821330 Bits	5797318 Bits
4.2.05	6291456 Bits	5196229 Bits	5172296 Bits	456387 Bits	4545897 Bits	4548757 Bits	4531451 Bits
4.2.06	6291456 Bits	5836421 Bits	5811496 Bits	5611604 Bits	5583938 Bits	5597542 Bits	5570011 Bits
4.2.07	6291456 Bits	5758867 Bits	5739275 Bits	5859575Bits	5842027 Bits	5827292 Bits	5808569 Bits

Tableau III.2 : Les tailles des images originales et compressées par les codeurs (Huffman et Arithmétique) en utilisant 3 espaces couleur.

	Le gain entre Huffman et Arithmétique			Le gain entre Huffman+TOR et Arithmétique+TOR		
	Format			Format		
Image	**RGB**	**Y'I'Q'**	**$O_1O_2O_3$**	**RGB**	**Y'I'Q'**	**$O_1O_2O_3$**
4.1.06	0.0695	0.0864	0.0906	0.0937	0.0984	0.0802
4.2.03	0.1039	0.0901	0.0916	0.0757	0.0943	0.0929
4.2.05	0.0913	0.0782	0.0660	0.0676	0.0953	0.1041
4.2.06	0.0951	0.1056	0.1050	0.0807	0.0886	0.0958
4.2.07	0.0747	0.0669	0.0715	0.0738	0.0983	0.0991

Tableau III.3 : Valeur du gain pour les différentes variantes (Huff et Arith, Huff_TOR et Arith_TOR).

Les résultats reportés dans la Figure III.2 montrent clairement que le codage Arithmétique est plus performant que celui de Huffman pour les 3 espaces couleurs. Nous observons également une diminution du débit et de la taille de l'image compressée pour toutes les images couleurs. De plus, la meilleure performance est obtenue avec l'espace couleur $O_1O_2O_3$ comparée aux deux autres espaces couleurs RGB et Y'I'Q'. Cette dernière transformation, qui offre une performance plus proche de l'espace $O_1O_2O_3$, permet une compression plus efficace par rapport à l'espace RGB car l'information est moins corrélée.

Comparé au codage de Huffman, le codage Arithmétique réalise des gains évidents qui varient entre 0.07 Bpp et 0.10 Bpp pour les formats RGB et $O_1O_2O_3$, et entre 0.07 Bpp et 0.11 Bpp pour le format Y'I'Q'comme illustré au Tableau III.3

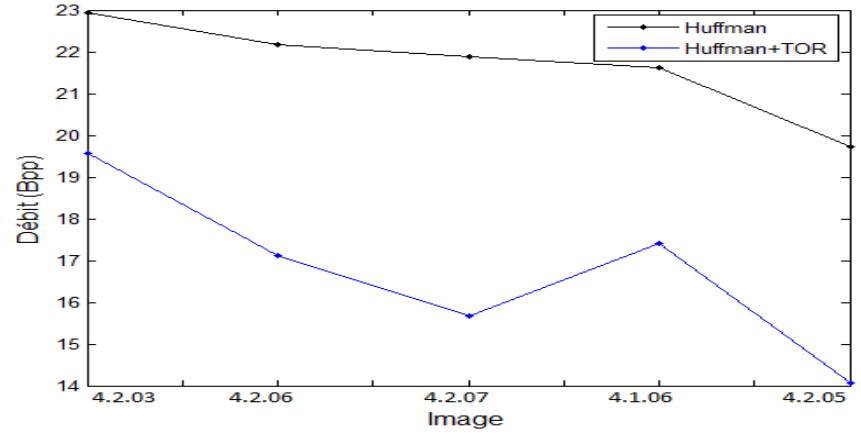
Le tableau III.2 résume les résultats obtenus en termes des tailles des images compressées, exprimées en bits, pour les codeurs de Huffman et Arithmétique. Il est bien clair, que les images compressées par le codeur Arithmétique occupent moins de bits. En autre, l'emploi des espaces couleurs du type Luminance/Chrominances : $O_1O_2O_3$ et Y'I'Q'améliore la performance de compression sans perte des images couleurs.

III.5Comparaison entre les deux variantes (Huff et Huff_TOR)

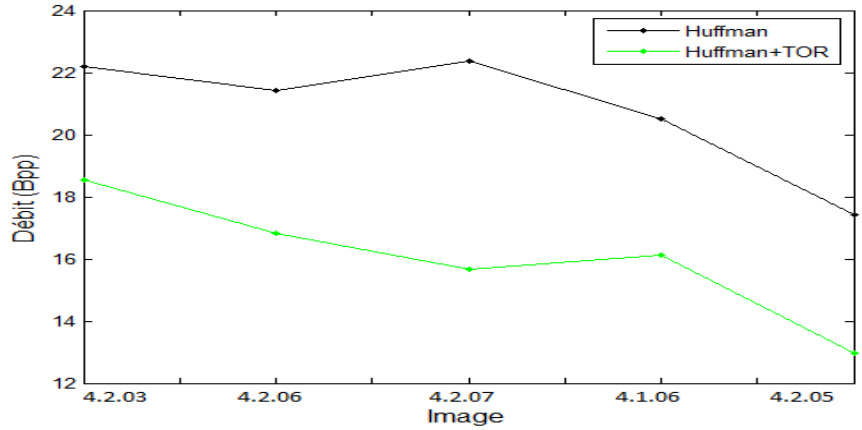
Cette partie aborde la comparaison des performances des deux algorithmes mis en œuvre dans les deux domaines, spatial et transformé, et ce pour les mêmes espaces de couleurs.

Nous tenons, tout de même, à signaler qu'à propos de la transformée en ondelettes discrètes, et dans le but de mener une évaluation objective, la comparaison des résultats est

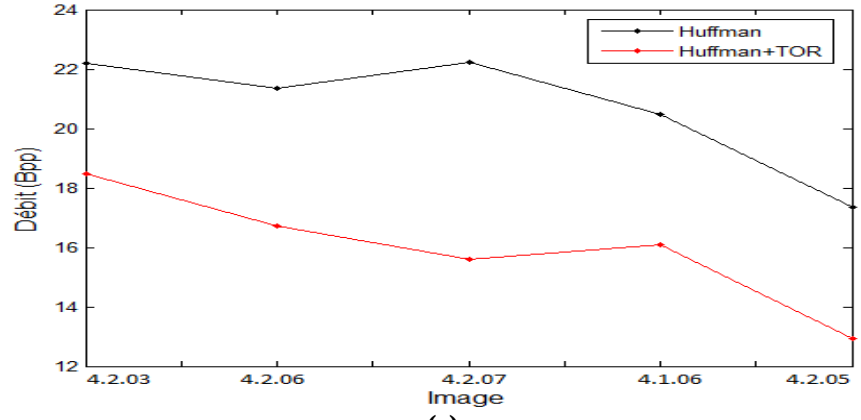
établie en utilisant la même ondelette (db1) et le même niveau de décomposition (6), qui reste suffisant pour l'amélioration des performances de compression. A noter que le but de cette partie est, d'une part, de montrer l'effet du domaine transformé, et d'autre part, de mettre en œuvre l'impact d'utiliser l'espace couleur réversible du type L/C sur la performance de compression sans perte en gardant le même codeur. Le Tableau III.4 ainsi que la Figure III.3 récapitulent les résultats obtenus.



(a)



(b)



(c)

Figure III.3: Evaluation de la performance de compression sans perte des variantes (Huff et Huff_TOR) en utilisant les formats : (a) RGB. (b) Y'I'Q'. (c) $O_1O_2O_3$.

	Format											
	RGB				Y'I'Q'				$O_1O_2O_3$			
	Codage de Huffman		Codage de Huffman +TOR		Codage de Huffman		Codage de Huffman +TOR		Codage de Huffman		Codage de Huffman +TOR	
Image	**Entropie**	**Débit**	**Entropie**	**Débit**	**Entropie**	**Débit**	**Entropie**	**Débit**	**Entropie**	**Débit**	**Entropie**	**Débit**
4.1.06	21.5448	21.6251	17.3109	**17.4053**	20.4388	20.5261	16.0107	**16.1101**	20.3881	20.4795	15.9893	**16.0703**
4.2.03	22.9333	23.0374	19.4777	**19.5537**	22.1184	22.2087	18.4422	**18.5368**	22.1148	22.2066	18.3934	**18.4865**
4.2.05	19.7305	19.8220	14.0102	**14.0781**	17.3410	17.4194	12.8550	**12.9505**	17.2859	17.3521	12.8113	**12.9157**
4.2.06	22.1689	22.2642	17.0388	**17.1197**	21.3008	21.4066	16.7269	**16.8157**	21.2477	21.3529	16.6337	**16.7298**
4.2.07	21.8934	21.9683	15.5882	**15.6622**	22.2854	22.3525	15.5506	**15.6492**	22.1577	22.2294	15.5094	**15.6088**
Moyenne	21.6542	21.7434	16.6852	**16.7638**	20.6988	20.7826	15.9171	**15.9171**	20.6388	20.7241	15.8674	**15.9622**

Tableau III.4 : Comparaison des performances de compression sans perte des images couleur pour les variantes (Huffman et Huffman+TOR) en utilisant 3 espaces couleur: RGB, Y'I'Q' et $O_1O_2O_3$.

	Formats					
	RGB	**Y'I'Q'**	**$O_1O_2O_3$**	**RGB**	**Y'I'Q'**	**$O_1O_2O_3$**
Image	**Le gain entre Huffman et Huffman TOR**			**Le gain entre Arithmétique et Arithmétique TOR**		
4.1.06	4.2198	4.4160	4.4092	4.2340	4.4280	4.3988
4.2.03	3.4837	3.6719	3.7201	3.4555	3.6761	3.7214
4.2.05	5.7439	4.4689	4.4364	5.7202	4.4860	4.4745
4.2.06	5.1445	4.5909	4.6231	5.1301	4.5739	4.6139
4.2.07	6.3061	6.7033	6.6206	6.3052	6.7347	6.6482

Tableau III.5 : Comparaison des gains des espaces couleur RGB Y'I'Q' $O_1O_2O_3$entre les variantes (Huffman et Huffman TOR) et (Arithmétique et Arithmétique TOR)

	Spatial				**Transformé**			
	Codage de Huffman		**Codage Arithmétique**		**Codage de Huffman**		**Codage Arithmétique**	
	Les Formats				**Les Formats**			
Image	**Le gain entre RGB Et YIQ**	**Le gain entre RGB et $O_1O_2O_3$**	**Le gain entre RGB et YIQ**	**Le gain entre RGB et $O_1O_2O_3$**	**Le gain entre RGB et Y'I'Q'**	**Le gain entre RGB et $O_1O_2O_3$**	**Le gain entre RGB et Y'I'Q'**	**Le gain entre RGB et $O_1O_2O_3$**
4.1.06	1.0990	1.1456	1.1059	1.1567	1.2952	1.3350	1.2999	1.3215
4.2.03	0.8287	0.8308	0.8149	0.8185	1.0169	1.0672	1.0355	1.0844
4.2.05	2.4026	2.4699	2.3895	2.4446	1.1276	1.1624	1.1553	1.1989
4.2.06	0.8576	0.9113	0.8681	0.9212	0.3040	0.3899	0.3119	0.4050

Tableau III.6: Les valeurs du gain pour les deux codeurs Huffman et Arithmétique dans les différents domaines spatial et transformé.

A partir du Tableau III.4, ainsi que la figure III.3 contenant les résultats obtenus, nous pouvons en tirer l'intérêt de la transformé réversible pour la compression sans pertes d'image.

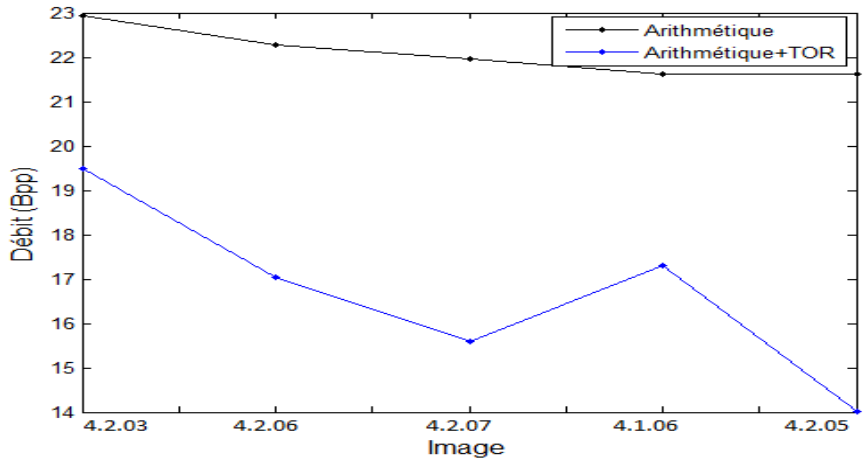
Sur la base de ces résultats affichés, nous pouvons en tirer l'intérêt de la transformée en ondelettes réversible pour la compression sans perte d'images. Nous illustrons ceci avec l'exemple de l'image 4.2.06. Pour les formats (RGB, Y'I'Q', $O_1O_2O_3$), les débits observés par l'emploi du codage de Huffman, dans le domaine spatial sont respectivement {22.2642, 21.4066, 21.3529} Bpp. En utilisant le domaine transformé, les valeurs du débit ont subi une baisse remarquable, prenant les nouvelles valeurs de {17.1197 ,16.8157 ,16.7298} Bpp, en enregistrant d'importants gains {5.1445, 4.5909, 4.6231}Bpp.

En suivant la même démarche, les mêmes résultats sont obtenus, et les mêmes remarques sont à faire, pour les autres images, à titre indicatif, pour ne prendre qu'une, pour l'image 4.2.05, les valeurs du débit obtenues pour (Huffman et Huffman_TOR) passent respectivement de {19.8220, 17.4194, 17.3521} Bpp à {14.0781, 12.9505, 12.9157} Bpp. Les gains significatifs obtenus sont respectivement {5.7439, 4.4689, 4.4364} Bpp. d'une manière générale, le gain moyen du codage de Huffman provenant de l'utilisation de la transformée en ondelettes réversible dépasse {4.97, 4.77, 4.76} Bpp, respectivement pour les trois formats (RGB, Y'I'Q', $O_1O_2O_3$).

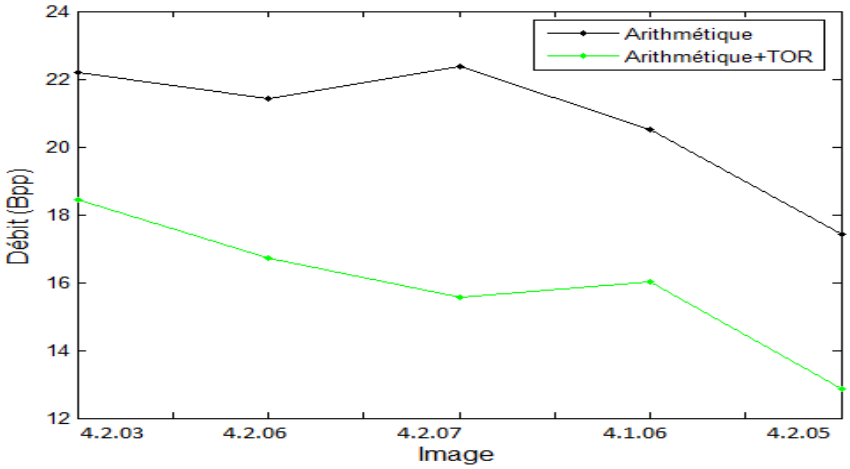
A partir des résultats obtenus, nous constatons aussi que la transformée réversible couleur présente un potentiel intéressant, contribuant aussi à l'amélioration des performances de compression sans perte. En gardant le même codeur (Huffman ou Huffman +TOR dans ce cas) et en changeant juste l'espace couleur de RGB vers Y'I'Q', ou de RGB vers $O_1O_2O_3$, le gain s'élève à 2.4 Bpp pour l'image 4.205. En général, le gain résultant varie entre {0.83 et 2.4} Bpp pour Huffman, et entre {0.30 et 1.30} Bpp pour Huffman +TOR, dépendant en fait de l'image utilisée.

III.6 Comparaison entre les deux variantes (Arith et Arith_TOR)

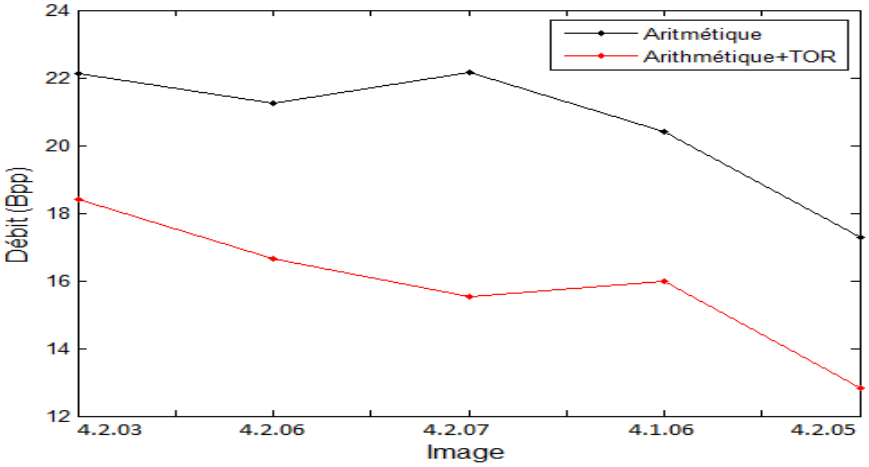
La démarche à mettre en œuvre dans cette section est similaire à la section précédente à l'exception du changement du codeur. Nous utilisons le codeur Arithmétique.au lieu de Huffman. Le tableau III.7et la Figure III.4 présentent les résultats obtenus. L'observation tirée de l'examen des résultats, aide clairement à apercevoir des différences significatives, entre la méthode de codage appliquée au domaine spatial, et celle appliquée au domaine transformé, et ce, pour toutes les données de test.



(a)



(b)



(c)

Figure III.4: Evaluation de la performance de compression sans perte des variantes (Codage Arithmétique et Codage Arithmétique+TOR) en utilisant le format (a) RGB (b) Y'I'Q'. (c) O_1O_2O

	Formats											
	RGB				Y'I'Q'				$O_1O_2O_3$			
	Codage Arithmétique		Codage Arithmétique+TOR		Codage Arithmétique		Codage de Arithmétique+TOR		Codage Arithmétique		Codage Arithmétique+TOR	
Image	**Entropie**	**Débit**	**Entropie**	**Débit**	**Entropie**	**Débit**	**Entropie**	**Débit**	**Entropie**	**Débit**	**Entropie**	**Débit**
4.1.06	21.5448	21.5456	17.3109	**17.3116**	20.4388	20.4397	16.0107	**16.0117**	20.3881	20.3889	15.9893	**15.9901**
4.2.03	22.9333	22.9335	19.4777	**19.4780**	22.1184	22.1186	18.4422	**18.4425**	22.1148	22.1150	18.3934	**18.3936**
4.2.05	19.7305	19.7307	14.0102	**14.0105**	17.3410	17.3412	12.8550	**12.8552**	17.2859	17.2861	12.8113	**12.8116**
4.2.06	22.1689	22.1691	17.0388	**17.0390**	21.3008	21.3010	16.7269	**16.7271**	21.2477	21.2479	16.6337	**16.6340**
4.2.07	21.8934	21.8936	15.5882	**15.5884**	22.2854	22.2856	15.5506	**15.5509**	22.1577	22.1579	15.5094	**15.5097**
Moyenne	21.6542	21.6545	16.6852	**16.6855**	20.6969	20.6972	15.9171	**15.9175**	20.6388	20.6392	15.8674	**15.8678**

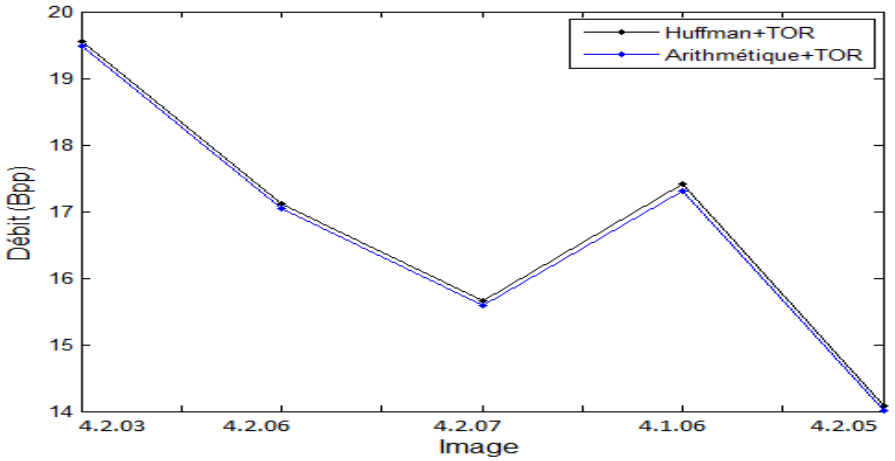
Tableau III.7 : Comparaison des performances de compression sans perte des images couleur pour les codeurs (Arithmétique et Arithmétique+TOR en utilisant 3 espaces couleur RGB, Y'I'Q' et $O_1O_2O_3$.

		Formats					
		RGB		Y'I'Q'		$O_1O_2O_3$	
		Codage de Huffman+TOR	Codage Arithmétique+TOR	Codage de Huffman+TOR	Codage Arithmétique+TOR	Codage de Huffman+TOR	Codage Arithmétique+TOR
Image	**Taille de l'image originale**	**Taille de l'image compressée**	**Taille de l'image compressée**	**Taille de l'image compressée**	**Taille de l'image compressée**	**Taille de l'image compressée**	**Taille de l'image compressée**
4.1.06	1572864 Bits	1140677 Bits	1134536 Bits	1055794 Bits	1572864 Bits	1053183 Bits	1053183 Bits
4.2.03	6291456 Bits	5125873 Bits	5106032 Bits	4859322 Bits	4834585 Bits	4846124 Bits	4846124 Bits
4.2.05	6291456 Bits	3690478 Bits	4466682 Bits	3394896 Bits	4384915 Bits	3385786 Bits	3385786 Bits
4.2.06	6291456 Bits	6291456 Bits	3672763 Bits	4408128 Bits	3369917 Bits	4385605 Bits	4385605 Bits
4.2.07	6291456 Bits	4105752 Bits	4086413 Bits	4102340 Bits	4076569 Bits	4091753 Bits	4091753 Bits

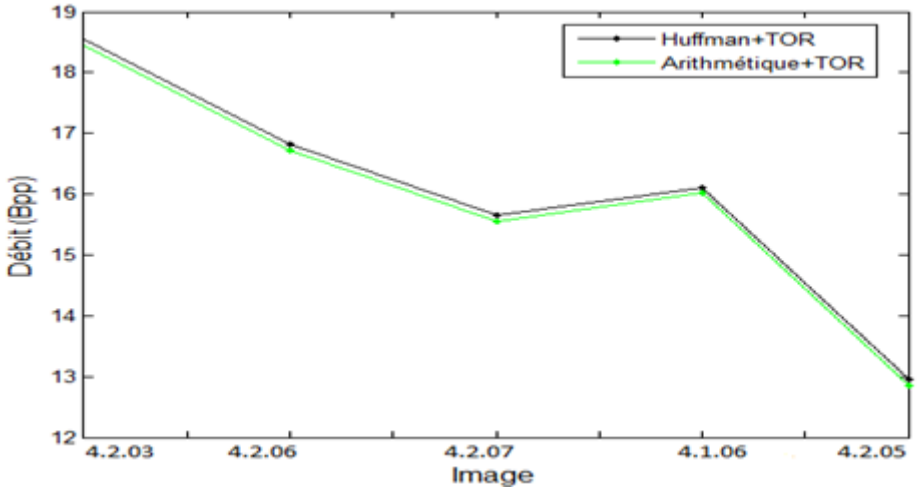
Tableau III.8 : Les tailles des images originales et compressées par les codeurs (Huffman+TOR et Arithmétique+TOR) en utilisant 3 espaces couleur.

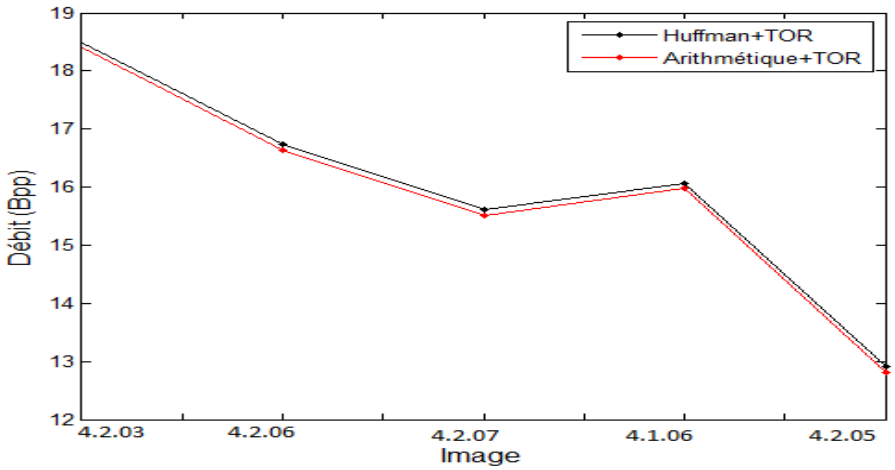
Nous étudions d'abord l'impact de la transformée en ondelettes réversible. Prenant l'exemple de l'image 4.2.06, l'entropie et le débit résultants pour le codage Arithmétique dans le domaine spatial par l'usage du format RGB sont respectivement {22.1689, 22.1691} Bpp. Cependant, après l'utilisation de la transformée, les débits sont diminués jusqu'à {17.0388, 17.0390} Bpp.

Pour le format Y'I'Q', en gardant la même image, nous obtenons les valeurs suivantes de l'entropie et du débit: {17.3410, 17.3412} Bpp. Toutefois, après application de la transformée, on aura les résultats suivants: {16.7269, 16.7271} Bpp. On termine par le format $O_1O_2O_3$. Les résultats de l'entropie et des débits pour le domaine spatial sont comme suit: {17.2861, 17.2859} Bpp, et {12.8116 ,12.8113} Bpp après avoir effectué la transformée en ondelettes réversible.



(a)





(c)

Figure III.5 : Evaluation de la performance de compression sans perte des codeurs Huffman et Arithmétique dans le domaine transformé, en utilisant le format : (a) RGB (b) Y'I'Q'(c) $O_1O_2O_3$.

		Format RGB			Format Y'I'Q'			Format $0_1 0_2 0_3$	
		Codage Huff+TOR	Codage Arith+TOR		Codage Huff+TOR	Codage Arith+TOR		Codage Huff+TOR	Codage Arith+TOR
Image	Entropie	Débit	Débit	Entropie	Débit	Débit	Entropie	Débit	Débit
4.1.06	17.3109	17.4053	17.3116	16.0107	16.1101	16.0117	15.9893	16.0703	15.9901
4.2.03	19.4777	19.5537	19.4780	18.4422	18.5368	18.4425	18.3934	184865	18.3936
4.2.05	14.0102	14.0781	14.0105	12.8550	12.9505	12.8552	12.8113	12.9157	12.8116
4.2.06	17.0388	17.1197	17.0390	16.7269	16.8157	16.7271	16.6337	16.7298	16.6340
4.2.07	15.5882	15.6622	15.5884	15.5506	15.6492	15.5509	15.5094	15.6088	15.5097
Moyenne	16.6851	16.7638	16.6855	15.9170	16.0124	15.9172	15.8674	15.9622	15.8677

Tableau III.9 : Comparaison des performances de compression sans perte des codeurs Huffman et Arithmétique dans le domaine transformé

Comme le montrent le TableauxIII.9 et les Figures III.5, la variante (Arith_TOR) fournit un faible débit pour toutes les images comparées à la variante (Huff_TOR), et que ce dernier a la particularité d'assurer que le débit est plus proche de l'entropie. Notons que la longueur moyenne d'un codeur (dans notre cas, le débit) a tendance à être plus similaire à l'entropie. En général, les méthodes de codage Arithmétique se sont avérées efficaces.

Evidemment, la comparaison avec le codage de Huffman confirme que le codage Arithmétique permet d'obtenir plus rapidement des résultats plus élevés.

L'espace couleur	Codage de Huffman	Codage de Huffman+TOR	Codage Arithmétique	Codage Arithmétique+TOR
	Débit moyen	**Débit moyen**	**Débit moyen**	**Débit moyen**
RGB	21.7434	16.7638	21.6545	16.8924
Y'I'Q'	20.7826	15.9947	20.6972	15.9172
$O_1O_2O_3$	**20.7241**	**15.9622**	**20.6391**	**15.8677**

Tableau III.10 : Les valeurs du débit moyen des deux codeurs (Huffman et Arithmétique) dans les deux domaines spatial et transformé.

Il est à noter, que nous avons testés dans ce tableauIII.10 plusieurs espaces de couleurs à savoir : RGB, Y'I'Q'et $O_1O_2O_3$ pour calculer les débits moyens des deux codages pour les méthodes spatiales et méthodes par transformation. Il est a noté que les valeurs du débit moyen et du gain mettent en évidence les performances des codages de Huffman et Arithmétique dans le domaine transformé en vue du processus de compression / décompression qui se fait sans pertes. Contrairement aux codages de Huffman et Arithmétique dans le domaine spatial qui présente un taux d'erreur. Dans un second lieu les valeurs du débit montrent que les espaces couleur $O_1O_2O_3$ et Y'I'Q' sont plus efficaces que l'espace RGB en raison du taux d'erreurs quasi nul lors du processus de compression / décompression.

III.7Comparaison entre les variantes (Huff_RGB et Huff_TOR_YIQ), Huff_RGB et Huff_TOR_$O_1O_2O_3$), (Arith_RGB et Arith_TOR_YIQ), Arith_RGB etArith_TOR_$O_1O_2O_3$)

Dans cette section nous mettons en évidence l'effet combiné des deux transformations réversibles utilisées, en l'occurrence, la transformée en ondelettes réversible et la transformation couleur réversible du type LC sur la performance décompression d'images couleurs sans perte. Pour cela, une comparaison des résultats obtenus est effectuée entre, d'une part, l'un des codeurs (Huffman ou Arithmétique) appliqué directement sur l'image couleur au format RGB, et

D'autre part, le même codeur est appliqué dans le domaine transformé aux formats (Y'I'Q'et $O_1O_2O_3$).

On désigne par:

Huff_RGB : le codage de Huffman appliqué directement sur l'image couleur RGB.

Arith_RGB : le codage Arithmétique appliqué directement sur l'image couleur RGB.

Huff_YIQ_TOR : le codage de Huffman appliqué au domaine transformé sur l'image couleur à travers l'utilisation du format Y'I'Q'.

Huff_TOR_$O_1O_2O_3$: le codage de Huffman appliqué au domaine transformé sur L'image couleur à travers l';utilisation du format $O_1O_2O_3$.

Arith_YIQ_TOR : le codage Arithmétique appliqué au domaine transformé sur l'image couleur à travers l'utilisation du format Y'I'Q'.

Arith_TOR_$O_1O_2O_3$: le codage Arithmétique appliqué au domaine transformé sur L'image couleur à travers l'utilisation du format $O_1O_2O_3$.

Image	Huff_RGB et Huff_TOR Y'I'Q'	Huff_RGB et Huff_TOR $O_1O_2O_3$	Huff_RGB et Arith_TOR Y'I'Q'	Huff_RGB et Arith_TOR $O_1O_2O_3$
4.1.06	5.5150	5.5548	5.6134	5.6350
4.2.03	4.5006	4.5509	4.5949	4.6438
4.2.05	6.8715	6.9063	6.9668	7.0104
4.2.06	5.4485	5.5344	5.5371	5.6302
4.2.07	6.3191	6.3595	6.4174	6.4586

Tableau III.11 : Comparaison entre les gains des variantes (Huff_RGB et Huff_TOR_YIQ), Huff_RGB et Huff_TOR_$O_1O_2O_3$), (Arith_RGB et Arith_TOR_YIQ), Arith_RGB et Arith_TOR_$O_1O_2O_3$).

Comme on peut le constater au Tableau III.11 les gains apportés par les méthodes du codage appliqués au domaine transformée avec l'utilisation d'un espace couleur du type L/C sont les plus importants et les plus significatifs. Pour Huffman, le gain obtenu varie entre {4.50 et 6.87} Bpp, et entre {4.55 et 6.91} Bpp, respectivement pour les formats Y'I'Q' et $O_1O_2O_3$. En effet le gain moyen s'élève à 5.69 Bpp et 5.73 Bpp, respectivement pour Y'I'Q' et $O_1O_2O_3$. Pour le codage Arithmétique le gain réalisé se situe entre {4.59 et 6.97} Bpp, et entre {4.64 et 7.01} Bpp, respectivement pour les formats Y'I'Q' et $O_1O_2O_3$. Effectivement,

le gain moyen totalise 5.78 Bpp et 5.83 Bpp, respectivement pour les espaces Y'I'Q' et $O_1O_2O_3$.

Au vu des résultats obtenus, le codage Arithmétique réalise la meilleure performance en comparaison avec le codage de Huffman, que ce soit l'application au domaine spatial ou transformé. De plus, les performances du codage Arithmétique sont plus proches de l'entropie. Notons que la longueur moyenne d'un codeur efficace (le débit dans notre cas) a tendance à être plus proche à l'entropie [24].L'utilisation d'une étape de transformation en ondelettes améliore la performance des codeurs. Ainsi, les codeurs appliqués au domaine transformé donnent de meilleurs résultats comparativement à ceux qui s'appliquent directement sur l'image. En outre, la performance de compression sans perte s'améliore encore en utilisant un espace couleur du type L/C, à la place de l'espace RGB. En conséquence notre étude souligne que bien meilleurs résultats sont offerts par la variante du codage Arithmétique qui est basée sur une étape de transformation et qui utilise un espace couleur du type L/C (Y'I'Q' ou $O_1O_2O_3$)[24].

III.8. Conclusion

L'évaluation et la comparaison des performances de compression sans perte, des deux techniques de codage : Huffman, et Arithmétique, sur des images couleur, ont constitué l'objet du présent chapitre. Nous portons connaissance, qu'avec les mêmes données de test, les résultats de quatre variantes obtenus, ont été analysées en termes de débit exprimé en Bpp
Au regard des résultats obtenus, le codage Arithmétique est le plus performant par rapport au codage de Huffman. Les codeurs appliqués au domaine transformé donnent de meilleurs résultats que ceux appliqués directement sur l'image, grâce à L'étape de transformation en odelettes réversible qui conduit à avoir un ensemble de coefficients entiers, moins corrélés. A l'aide d'une deuxième étape de transformation couleur réversible du type L/C, dont le but est de réduire la corrélation existante au format RGB, les résultats obtenus montrent encore une amélioration, en enregistrant des gains appréciables par comparaison aux variantes ne comportant aucune étape de transformation.

Conclusion Générale

Le travail présenté dans ce mémoire a été consacré à l'étude de l'effet conjoint des transformations réversibles sur la performance de compression sans perte des images couleur. Ceci, est dans le but de trouver l'espacede couleurs le plus adapté à la compression. Afin d'atteindre cet objectif, nous avons employé deux techniques de codages entropiques,à savoir: le codage de Huffman et le codage Arithmétique, avec applications au domaine spatial et transformé.

En raison de la forte corrélation existante entre les trois couleurs primaires (R, G, B), nous avons modifié l'espace couleur RGB en faveur d'autres espaces couleur réversibles, moins corrélés, à savoir les espaces: Y'I'Q' et $O_1O_2O_3$. Au cours d'une première étape, l'application directe des techniques de Huffman et Arithmétique a été conduite sur l'image couleur originale au domaine spatial.Dans la seconde étape, une combinaison de deuxtransformations réversibles est formée: la transformée en ondelettesréversible, d'aune part, et la transformation couleur réversible, d'autre part. Les deux techniques du codage (Huffman et Arithmétique) sont, par la suite, appliquées à ce domaine transformé.

Compte tenu des résultats présentés, provenant de la compression sans perte des images couleur, le codage Arithmétique se révèle efficace et offre la meilleure performance par rapport au codage de Huffman. La performance observée apparaît nettement supérieure à celle du codage de Huffman tant dans le domaine spatial que dans le domaine transformé, en termes du débit, exprimé en Bpp. De plus, les résultats de notre étude montrent clairement que l'application d'une étape de transformation en ondelettes réversible conduit à établir un système de compression plus performant, permettant des gains moyens, plus importants, supérieurs à 4.7 Bpp pour les deux variantes du codage Huffman et Arithmétique. Ainsi, les codeurs qui s'appliquent au domaine transformé affichent des résultats plus appréciables que ceux qui s'appliquent directement sur l'image. En outre, les résultats obtenus ont mis en évidence le rôle primordial des transformations couleur réversibles employées (Y'I'Q' et $O_1O_2O_3$).

En effet, l'application de ces dernières, qui traitent ces entiers et retournent des entiers, a pour effet d'améliorer encore la performance de compression sans perte. En conclusion, notre étude a mis en lumière l'effet conjoint des deux transformations réversibles à coefficients entiers pour mettre au point un système de compression sans perte, en réalisant des gains moyens très significatifs qui s'élèvent à 5.7 Bpp pour les deux techniques de codage utilisées et appliquées.

Bibliographie

[1] A. Boucetta, Etude de l'effet des transformations de corrélation en compression des images couleurs RGB, Thèse de Doctorat, Université de Batna 2, 2010.

[2] W. Boureba et W. Bendjerid, Compression d'image couleur par ondelettes à base de la structure Lifting: Application aux images satellitaires, Mémoire de Master, Université Belhadj Bouchaib d'Ain temouchent, Année 2015-2016.

[3] S. Benlahcene, Compression des images médicales par ondelettes, Thèse de Doctorat, Université Abou Bakr Belkaid– Tlemcen, 2018.

[4] Color image, Wikipedia.org>WIKI>Color_umage.

[5] D. Zeroual, implémentation d'un environnement parallèle pour la compression d`image à l'aide des fractales, Thèse de Doctorat, Batna, 2006.

[6] J. Luc, Les images numériques, Généralités, Cours TERI, Diplôme d'Université Assistant à l'Usage des Technologies de l'Information et de la Communication (DU AUTIC) DEUST, Université de Bretagne Occidentale, 2002/2003.

[7] M. Hazem, R. Nabi, Les méthodes utilisées pour la reconnaissance de visage, Mémoire de Master, Universités d'Avignon et du pays du Vaucluse IUP GMI, 2006-2007.

[8] S. Katzenbeisser, F. Petitcolasn, Information Hiding Techniques for steganography and Digital Watermarking, Artech House, London, 2000.

[9] L. Bouanzi, Les formats de numération des images fixes, DESS en informatique Documentaire, Rapport de stage, Ecole Nationale Supérieure des Sciences de l'Information et des Bibliothèques (ENSSIB), Université Claude Bernard Lyon 1, 1999.

[10] J. Fruitet, Outils et méthodes pour le traitement des images par ordinateur, Cours Université de Marne-la-vallée, France, 2000.

[11] S. Bensiali, R. Ajhoun, and M. Abik, ''Novel approach for accessible visual resources in a Web based learning environment'', In Proceedings of the 2012 IEEE Global Engineering Education Conference, IEEE, pp. 1-7, 2012.

[12] R .C. Gonzalez, R. E. Woods, Digital Image Processing, 4th Edition, Pearson, 2018

[13] I. Singh, I. Agathoklis and P. Antoniou, ''Lossless compression of color images using an improved integer-based nonlinear wavelet transform'', International symposium on Circuits and Systems. IEEE, pp. 2609-2610, Hong Kong, 1997.

[14] B. C. Dhara, B. Chanda, '' Color image compression based on block truncation coding using pattern fitting principle'', Pattern Recognition, Vol. 40, No 9, pp. 2408-2417,

2007.

[15] P. Beaurepaire, E. Beretta, Compression d'Images appliquée aux Angiographies Cardiaques: Aspects algorithmiques, Evaluation de la qualité diagnostique, Thèse de Doctorat, Ecole doctorale des sciences pour l'ingénieur, Vol. 156, pp. 157-158, Lyon, 1997.

[16] T. Nakachi, T. Fujii, and J.Suzuki, ''Lossless and near-lossless compression of still color images '' Proceedings 1999 International Conference on Image Processing, Vol. 1. IEEE, pp .453-454, 1999.

[17] A. Czihó, Quantification et Compression d'image. Application à l'imagerie Médicale, Thèse de Doctorat, Université de Rennes 1, France, Mai 1999.

[18] Publié par : Margaret Rouse sur https://www.lemagit.fr/.

[19] I. Bouklihacene, Compression d'images médicales par ondelettes de seconde Génération, Thèse de Doctorat, UABB-Tlemcen, 2014.

[20] M. Rabbani, ''JPEG2000: Image compression fundamentals, standards and practice. Journal of Electronic Imaging'', Vol. 11, No 2, pp. 286, 2002.

[21] H. Karima, Z. Malha, Etude comparative de méthodes de compression d'images basées sur les codeurs par plans de bits, Mémoire de Master, Université Mouloud Mammeri, 2016.

[22] R. J. Clarke, Digital compression of still images and video, Academic Press, Inc, 1995.

[23] S. Brahmi, et Z. Benziane, La compression des images médicales sous un Smartphone Android. Thèse de Doctorat, Université Abou Bakr Belkaïd de Tlemcen, 2016-2017.

[24] K. Sayood, Introduction to Data Compression, Third Edition , Morgan Kaufmann Series in MultiMedia Information and Systems, 2017.

[25] B. Tounsi et H. Zidoune, Compression de données, Mémoire de Master, Université Abderrahmane Mira, Bejaïa, 2015-2016.

[26] A. M. M. S. BOUHLEL, Un nouvel algorithme de compression exploitant le codage arithmétique en lui introduisant de nouveaux paramètres de codage, Institut Supérieur de Biotechnologie de Sfax (ISBS)-TUNIS, 2007.

[27] J. J. Brault, D. Dougherty, Les formats de compression d'image, Rapport de Projet, Institut Universitaire de Technologie de Tours, Département, 2004.

[28] F. Douak, Reconstruction des images compressées en utilisant les réseaux de neurones artificiels et la DCT, Thèse de Doctorat, Université de Batna 2, 2008.

[29] L. Belhumeur, La compression d'images fixe infrarouges contenant des cibles tenues, Thèse de Doctorat, Université Laval, juillet 1997.

[30] B. Rémi, N. François, La compression JPEG, Rapport de Projet, Université de Caen Basse-Normandie, Année 2005-2006.

[31] M. Mekouar, Compression d'images médicales par ondelettes et régions d'intérêt, Thèse de Doctorat, École de technologie supérieure, 2001.

[32] A. K. Jain, ''Image data compression'', review, Proceedings of the IEEE, Vol. 69, No 3, pp. 349-389,1981.

[33] G. K. Wallace, ''Overview of the JPEG (ISO/CCITT) still image compression standard'', in Image Processing Algorithms and Techniques, International Society for Optics and Photonics, pp. 220-233,1990.

[34] G. Mercier, C. Roux, et G. Martineau, ''Technologies du Multimédia'', ENST Bretagne, Vol. 832, pp.29280, France, 15 janvier 2003.

[35] T. Acharya, P .S . Tsai, JPEG2000 standard for image compression: concepts, algorithms and VLSI architectures, John Wiley & Sons, 2005.

[36] G. K. Wallace, ''The JPEG still picture compression standard'', IEEE Transactions on Consumer Electronics, Vol. 38, No 1, pp. xviii-xxxiv, 1992.

[37] R. Khemiri, F.E. Sayadi, H .Bahri, and al. ''Implementation and Comparison of the Lifting 5/3 and 9/7 Algorithms in MatLab on GPU'', Journal of Electrical Systems, vol. 12, no 3, pp. 490-499, 2016.

[38] S. Saha, '' Image compression, from DCT to wavelets, XRDS: Crossroads, The ACM Magazine for Students'', Vol. 6, No 3, pp. 12-21, 2000.

[39] Y. Boudjit, Compression d'images sans perte par des techniques du codage source Mémoire de Master, Université Mohammed Seddik Benyahia-Jijel, Algérie, 2019.

Résumé

L'objectif de ce projet est d'étudier l'effet conjoint des transformations réversibles sur les performances de compression sans perte des images couleur. Il s'agit de la transformation en ondelettes réversible d'une part, et la transformation réversible couleur d'autre part. Pour ce faire, nous utilisons deux techniques de codage source (Huffman et Arithmétique), couramment utilisées en compression de données, pour mener une étude comparative entre plusieurs variantes, qui s'appliquent soit au domaine spatial, soit au domaine transformé. La première consiste à appliquer les codeurs directement sur l'image couleur, et la seconde consiste à appliquer les mêmes codeurs dans le domaine transformé en combinant deux transformations réversibles, dans ce cas la transformée en ondelettes réversible et la transformée couleur réversible (Y'I'Q', O1O2O3). Le but est de réduire la corrélation et de trouver l'espace de couleurs le plus efficace et le plus approprié pour la compression. Une comparaison des résultats obtenus est par la suite établie en termes du débit exprimé en bits par pixel.

Abstract

The aim of this project is to study the joint effect of reversible transformations on the lossless compression performance of colour images. It concerns the reversible wavelet transform on the one hand, and the reversible colour transformation on the other hand. To do this, we use two source coding techniques (Huffman and Arithmetic), commonly used in data compression, to carry out a comparative study between several variants, which apply either to the spatial domain or to the transformed domain. The first consists of applying the encoders directly on the colour image, and the second one consists of applying the same encoders in the transformed domain by combining two reversible transformations, in this case the reversible wavelet transform and the reversible colour transform (Y'I'Q', O1O2O3). The goal is to reduce the correlation and find the most efficient and suitable colour space for compression. A comparison of the results obtained is then established in terms of the bit rate expressed in bits per pixel.