**REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE**
**MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE**

**UNIVERSITE MOULOUD MAMMERI, TIZI-OUZOU**
**FACULTE DE GENIE ELECTRIQUE ET DE L'INFORMATIQUE**
**DEPARTEMENT D'ELECTRONIQUE**

*MEMOIRE DE MAGISTER EN ELECTRONIQUE*

*OPTION : **Télédétection***

Présenté par :

**M^r^ CHERIFI Mehdi**

THEME :

# Application de la transformation de Radon discrète pour la compression d'images MSG

Devant les membres du jury :

| | | | | |
|---|---|---|---|---|
| Président : | Mr. AMEUR | Soltane | Professeur | UMMTO |
| Rapporteur : | Mr. LAHDIR | Mourad | Maître de conférences A | UMMTO |
| Examinateurs : | Mr. HADDAB | Salah | Maître de conférences A | UMMTO |
| | Mr. LAZRI | Mourad | Maître de conférences B | UMMTO |

Soutenu le: 05/03/2015

# Remerciement

Ce mémoire a été réalisé au sein du laboratoire d'Analyse et Modélisation des Phénomènes Aléatoires (LAMPA) de la Faculté de Génie Electrique et Informatique de l'Université Mouloud MAMMERI de Tizi-Ouzou.

La première personne que je souhaite remercier est mon directeur du mémoire Monsieur LAHDIR Mourad. Il m'a constamment soutenu, encouragé et stimulé le long parcours de ce mémoire. Ses nombreuses remarques ont montré une très vaste connaissance des sujets abordés et m'ont conduit à améliorer mon travail.

Je remercie vivement Monsieur AMEUR Soltane, Professeur à l'UMMTO d'avoir accepté de présider le jury.

J'exprime mes plus sincères remerciements à Monsieur HADDAB Salah, Maitre de Conférences à l'UMMTO d'avoir accepté de faire partie d jury.

Mes remerciements vont aussi à Monsieur LAZRI Mourad, Docteur à l'UMMTO, pour l'intérêt qu'il a porté à ce travail et d'accepter de faire partie du jury.

Je remercie vivement Madame AMEUR Zohra, Professeur à l'UMMTO et Directrice du laboratoire LAMPA d'avoir mis à notre disponibilité les moyens nécessaires pour le développement de nos travaux. Un grand merci aussi, aux membres du laboratoire LAMPA ainsi à ceux qui assurent son bon fonctionnement.

Je remercie particulièrement, Nos enseignants de première année: Mr. ADDANE, Mme. AMEUR, Mr. AIDENE, Mr. HADDAB, Mr. LAGHROUCHE, Mr. LAHDIR, et Mlle. NOUALI pour leur contribution et le temps qu'ils ont bien voulus consacrer pour nous instruire. Qu'ils veuillent bien trouver ici l'expression de ma profonde reconnaissance.

A mes parents bien-aimés, mes frères, sœurs, amis. Ce mémoire est pour vous.

# Résumé

La compression d'images a pour but de réduire le nombre de bits nécessaires pour représenter une image. Le satellite MSG (Meteosat Second Generation) permet l'acquisition de douze fichiers image toutes les quinze minutes ce qui engendre des bases de données de volume important. La transformée sélectionnée dans la compression d'images doit contribuer à la réduction des données représentants l'image. La transformée de Radon permet de récupérer les points Radon qui représentent la somme des pixels dans chaque direction pour une plage d'angle donnée. Le codage prédictif linéaire LPC avec filtrage assure une forte décorrélation des points Radon. L'utilisation du codage RLC nous donne un taux de compression élevé et fixe quel que soit l'image introduite au départ. Nous présentons dans ce mémoire une nouvelle méthode de compression d'images MSG basée sur la transformée de Radon et le codage prédictif linéaire LPC. La comparaison de notre méthode de compression avec d'autres récemment publiées dans la littérature donne avantage à notre méthode qui assure un bon compromis débit-distorsion. L'application de notre méthode sur une base de données constituée de deux types d'images : test et Météosat a permis d'atteindre un taux de compression de 99.90%. Ces résultats montrent clairement que notre méthode est qualifiée pour être un bon choix pour l'archivage et la transmission.



***Mots-clés*** : Compression d'images, Transformée de Radon, Codage Prédictif Linéaire (LPC), Run Length Coding (RLC), Meteosat Second Generation (MSG).



# Abstract

The image compression consists to reduce the number of bits required to represent an image. The MSG (Meteosat Second Generation) satellite allows the acquisition of twelve image files every fifteen minutes which causes a large databases. The selected transform in the image compression must contribute to the reduction of the image data. The Radon transform retrieves Radon points that represent the sum of the pixels in each direction for a given angle range. Linear predictive coding LPC with filtering ensures a high decorrelation of Radon points. The use of the RLC encoding we gives a high compression ratio and fixed regardless of the image introduced initially. We present in this Thesis a new MSG image compression method based on the Radon transform and the Linear Predictive Coding LPC. The comparison of our compression method with other recently published in the literature gives advantage to our method which provides a good compromise rate-distortion. The application of our method on a database consisting of two types of images: test and Meteosat allowed to achieve a compression ratio of 99.90%. These results clearly show that our method is qualified to be a good choice for archiving and transmission.



***Keywords***: Image compressions, Radon transform, Linear Predictive Coding (LPC), Run Length Coding (RLC) and Meteosat Second Generation (MSG).

# *Table des matières*

# *Liste des tableaux*

## *Chapitre I Etat de l'art sur la compression d'images*



## *Chapitre II La transformée de Radon*



## *Chapitre IV Résultats et discussions*

# *Liste des figures*

## *Chapitre I Etat de l'art sur la compression d'images*



## *Chapitre II La transformée de Radon*

# Liste des sigles et abréviations

**ADPCM**: Adaptive Differential Pulse Code Modulation

**ART**: Algebric Reconstruction Technique

**CCITT**: International Telegraph and Telephone Consultative Committee

**CFD**: Cumulative Distribution Function

**Codec**: Compression/ Decompression

**DCT**: Discrete Cosine Transform

**DPCM**: Differential Pulse Code Modulation

**DPCM-1D**: One-Dimensional DPCM

**DPCM-2D**: Two-Dimensional DPCM

**DWT**: Discrete Wavelet Transform

**FBP**: Filter-BackProject

**FFT-2D**: Two-Dimensional Fast Fourier Transform

**FIR**: Finite Impulse Response filter

**GR**: Golomb-Rice coding

**HDTV**: High Definition Television

**IR**: InfraRed channel

**IRCCyN** : Institut de Recherche en Communications et Cybernétique de Nantes

**ISO**: International Standards Organization

**ITU-T**: Telecommunication Standardization Sector of the International Telegraph Union

**IVC**: Images et Video Communications

**JND**: just noticeable difference

**JPEG**: Joint Photographic Experts Group

**KLT**: Karhunen-Loève Transform

**LPC**: Linear Predictive Coding

**MAE**: Mean Absolute Error

**MPEG**: Moving Picture Experts Group

**MSE**: Mean Square Error

**MSG**: Meteosat Second Generation

**NIR**: Near InfraRed channel

**PSNR**: Peak Signal to Noise Ratio

**RAD-DCT**: Image compression technique based on Radon transform and DCT proposed by S. A. Pradeep and R. Manavalan

**RAD-LPC**: Image compression technique based on Radon transform and Linear Predictive Coding LPC

**Ram-Lak**: Ramachandran and Lakshminarayanan filter

**RLC**: Run Length Coding

**RMSE**: Root Mean Square Error

**SDTV**: Standard Definition Television

**SEVIRI**: Spinning Enhanced Visible & InfraRed Imager

**SNN**: Symmetric Nearest Neighbor

**VIS**: Visible channel

**WV**: Water Vapour channel

**1D-DWT**: One-Dimensional DWT

**2D-DWT**: Two-Dimensional DWT

# Introduction

Les images numériques nécessitent de grandes capacités pour la mémoire de stockage et de large bande passante pour la transmission. Ces images qui proviennent de différentes sources (satellites, radars, photographie, télévision, imageurs médicaux) ont non seulement besoin d'être codées de manière efficace, mais ont aussi le besoin d'être compressées pour diminuer de leurs tailles et rendre leur transmission possible.

Dans notre travail, nous nous s'intéressons aux images fournies par le satellite géostationnaire Météosat Seconde Génération « MSG », qui nous fournit des images de l'observation de la Terre au niveau de l'équateur toutes les quinze minutes. Ce satellite apporte une contribution essentielle à la prévision météorologique ainsi qu'à l'étude des phénomènes environnementaux et climatiques à l'échelle mondiale. Il nous permet de suivre l'évolution détaillée des phénomènes atmosphériques grâce aux douze canaux embarqués**.** Le volume de ces images du satellite MSG accumulé aux files du temps, pour étudier l'évolution du climat sur une longue période, nous confronte aux problèmes de stockage et de transmission. L'échange de ces données entre les utilisateurs pour leurs différents traitements et applications exige des réseaux à haut débit, donc d'énormes changements dans les infrastructures de télécommunications. Le stockage de ces images Météorologiques exige un espace mémoire très important. La réduction de ce volume de données par des méthodes de compression des images est alors devenue plus que nécessaire **[Lahdir et al 2007]**.

Nous distinguons deux grandes classes de méthodes de compression d'images (Codec) : la compression sans pertes et la compression avec pertes. La compression sans pertes permet de reconstruire après décompression, une image identique à l'originale mais avec un faible taux de compression. Parmi les codeurs les plus utilisés, nous citons le codage Huffman **[Huffman 1952]**, le codage arithmétique, le codage Golomb-Rice **[Rice and Plaunt 1971]**, le codage Tunstall **[Tunstall 1967]** et le codage RLC (Run Length Coding). Nous distinguons aussi le codage prédictif dans le cas d'utilisation des prédicteurs simple sans introduction d'un filtrage comme le codage LPC (Linear Predictif Coding).

Ces codeurs se distinguent de la compression avec pertes qui introduit une dégradation irréversible sur l'image originale mais qui autorise une compression bien plus importante que celle obtenue par des méthodes sans pertes. Il existe trois types de compression avec pertes : le premier type est la compression d'images dans le domaine des transformations linéaires.

Parmi ces transformations, nous distinguons la transformée de Karhunen-Loève (KLT), la Transformée Cosinus Discrète (DCT), la transformée de Radon **[Radon 1917]**, la transformée Mojette **[Guédon 1995]**, la transformée fractal.

Le deuxième type est la compression d'images dans le domaine des sous-bandes où la transformée en ondelettes est appliquée à l'image entière.

Le troisième type est le codage prédictif avec pertes comme le DPCM (Differential Pulse Code Modulation) **[Chapin 1950]** où l'erreur globale lors de la reconstruction des pixels dépend sur le design et les performances du quantificateur en plus de celles du prédicteur. La quantification scalaire ou vectorielle, est au cœur de la plupart des méthodes de compression avec pertes. Elle permet de représenter une sortie issue d'une transformée par un nombre de coefficients réduit pour les coder ensuite par un codage entropique pour générer les mots-codes.

Nous nous nous s'intéressons dans le cadre de ce mémoire aux méthodes de compression avec pertes pour compresser des images issues du satellite MSG. Ces méthodes nous permettent d'assurer un taux de compression élevé tout en gardant la qualité de l'image décompressée. Le but est donc de réaliser un bon compromis entre le taux de compression et la dégradation introduite sur l'image. Cela nous a conduit à développer dans le cadre de ce mémoire, une méthode de compression d'images avec pertes basée sur la transformée de Radon et le codage prédictif LPC. La transformée de Radon est devenue un outil très important qui a séduit beaucoup de chercheurs dans le domaine de la compression d'images vu sa robustesse aux différents bruits comme le bruit blanc et le bruit de quantification par rapport aux autres types de transformations comme la transformée de Fourier et les ondelettes. Elle nous a conduit, notamment, à traiter seulement les points Radon, qui représentent les sommes des pixels aux différents angles. Le codage prédictif LPC utilisé est très intéressant car il assure une décorrélation maximale des points Radon, donc des pixels. Il permet aussi d'extraire seulement l'information utile qui réside dans les différents pixels. Ce processus introduit une certaine perte de données des images à compresser. Cela est principalement dû au design du prédicteur qui est un filtre SNN (Symmetric Nearest Neighbor) **[Harwood et al 1987]** dans le but de mieux préserver les contours. Notre choix s'est porté sur le codage arithmétique pour coder les différentes informations extraites par le codage prédictif LPC. Ce choix est justifié par son avantage qui consiste à ne pas sauvegarder les mots-codes issus du codage dans un dictionnaire comme le cas du codage Huffman. Notre but était de réaliser une compression forte avec moins de dégradation. Nous introduisons dans notre méthode le

codage RLC qui permet d'apporter une puissance à notre codeur tout en obtenant un taux de compression très élevé et fixe quel que soit l'image de départ.

L'étude présentée dans ce mémoire s'articule autour de quatre chapitres.

Le **premier chapitre** présente les notions de bases de la compression d'images en donnant les principaux concepts et les techniques associées à la compression d'images. Un état de l'art des principales méthodes de compression récemment publiées dans la littérature est présenté.

Nous nous focalisons lors du **deuxième chapitre** sur les principes de la transformée de Radon qui constitue le noyau de notre technique de compression d'images. Nous donnerons aussi son application dans le domaine de l'imagerie.

Le **troisième chapitre** est consacré à la description de notre méthode de compression que nous avons élaborée. Nous décrirons chaque bloc de notre méthode d'une façon claire et détaillée.

Le **quatrième chapitre** est consacré à l'évaluation de notre méthode de compression. Nous nous basons sur les critères de qualité les plus utilisés dans la compression d'images pour évaluer le rapport compression/dégradation.

Enfin, une synthèse du travail effectué, suivie d'une discussion sur les perspectives ouvertes par nos études, conclut le corps de ce mémoire.

# Chapitre I

# Etat de l’art sur la compression d’images

# Chapitre 1

# Etat de l'art sur la compression d'images

## 1.1. Préambule

Ce premier chapitre se concentre sur les outils utilisés en compression d'images. Le codage de source ou compression de données a pour objectif de fournir une représentation efficace des données tout en préservant l'information essentielle. Nous envisageons dans ce chapitre, une représentation des principaux concepts de base et techniques associées à la compression d'images en général. Toutes ces méthodes ne sont pas forcément dédiées spécifiquement à la compression d'images MSG. Nous évoquerons notamment quelques algorithmes de compression appliqués aux images récemment publiés dans la littérature.

## 1.2. Introduction à la compression et critères d'évaluation

La compression d'images consiste à réduire le volume de données nécessaires à la description de l'image. L'exploitation de la redondance spatiale et spectrale et les faiblesses du système psychovisuel ont permis de développer des méthodes qui conduisent à la réduction de la quantité de données nécessaires pour la représentation de l'image. Nous distinguons deux grandes classes de compression : la compression sans pertes et avec pertes.

Dans le cas d'une compression sans pertes, l'image reconstruite après décompression est identique à l'image originale. Aucune perturbation n'est introduite sur les valeurs des pixels. Dans le cas contraire, la compression avec pertes consiste à introduire une perturbation irréversible mais qui autorise une compression bien plus importante que celle obtenue par des méthodes sans pertes.

### Pourquoi la compression est-elle nécessaire ?

Nous donnons un exemple simple pour montrer la nécessité de la compression d'images.

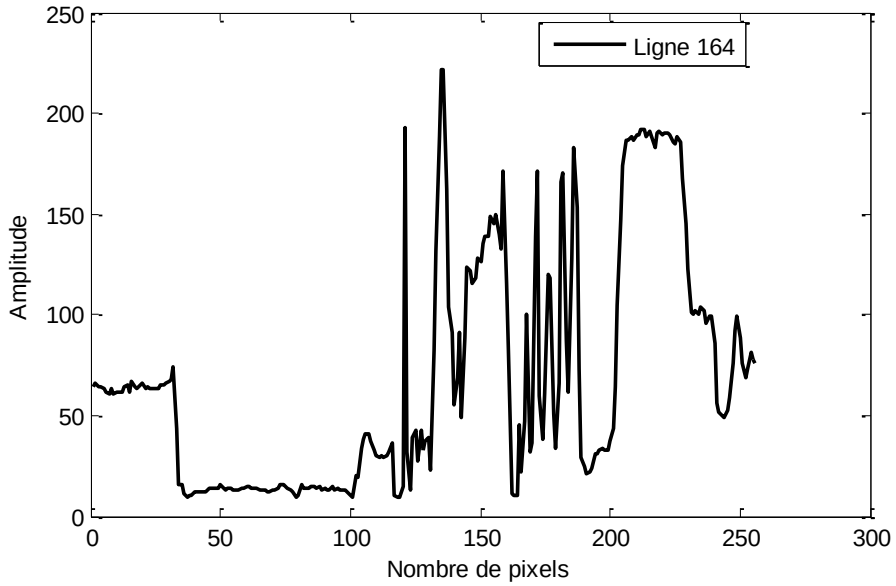
- Une image SD (Standard Definition) comporte 480 lignes, le nombre de pixels par ligne est $480 \times 4/3 = 640$. Dans le cas d'une image couleur avec 24 bpp, l'image aura besoin d'un espace mémoire de $640 \times 480 \times 3 = 921600\ octets$.
- Une image HD (High Definition) comporte 1080 lignes, le nombre de pixels par ligne est $1080 \times 16/9 = 1920$. Dans le cas d'une image couleur avec 24 bpp, l'image aura besoin d'un espace mémoire de $1920 \times 1080 \times 3 = 6220800\ octets$.
- Une source vidéo peut produire $30\ trames/second$ ou plus. Le taux d'information brut sera $221184000\ bits/second$ pour la SDTV et de $1492992000\ bits/second$ pour la HDTV.
- Le transfert de cette quantité à travers d'un canal de transmission idéal en temps réel ($1\ Hz/2\ bits$) nécessite une bande passante de 110592000 Hz pour la TVSD et de 746496000 Hz pour la télévision HDTV.
- Pratiquement, il existe aucun canal qui assure une telle immense bande passante de transmission pour une longue distance. Donc il est clair qu'un schéma efficace de compression de données est nécessaire pour abaisser un tel immense taux de transfert de données.

Dans une image, chaque pixel dans une ligne peut avoir une valeur très proche ou égale à celle de son voisin. Nous considérons comme exemple l'image test Cameraman montrée dans la figure 1.1.

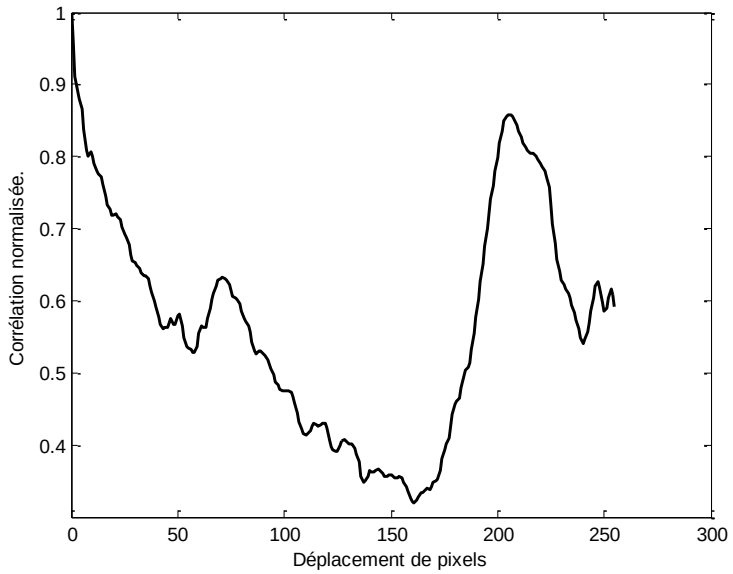
Figure 1.1. Image Cameraman.

La figure 1.2 montre le profil (a) et la corrélation correspondante (b) pour l'image Cameraman le long de la ligne 164.

Nous observons à partir de la figure 1.2 que les valeurs des pixels sont presque les mêmes sur un nombre large de voisinage : c'est la corrélation interpixel. En d'autres termes, les pixels sur une ligne ont une forte corrélation. De même, les pixels peuvent avoir aussi une forte corrélation dans une colonne. Donc, le principe fondamental de la compression d'images revient à décorréler les pixels et de coder ensuite le résultat de la décorrélation. Le résultat de la compression est en fonction de la méthode choisie pour éliminer la corrélation interpixel.




(a)




(b)

Figure 1.2. Profil de l'image Cameraman le long de la ligne 164.
(a) Intensité des pixels le long de la ligne 164.
(b) Corrélation normalisée correspondante sur un déplacement de 256 pixels.

Nous représentons une pléthore des méthodes de compression d'images dans un diagramme arborescent comme l'illustre la figure 1.3. Notons que la compression sans pertes fait toujours partie de la compression avec pertes. Elle est utilisée pour coder de façon sans pertes les différents pixels quantifiés ou les coefficients d'une transformation lors de l'étape de compression.

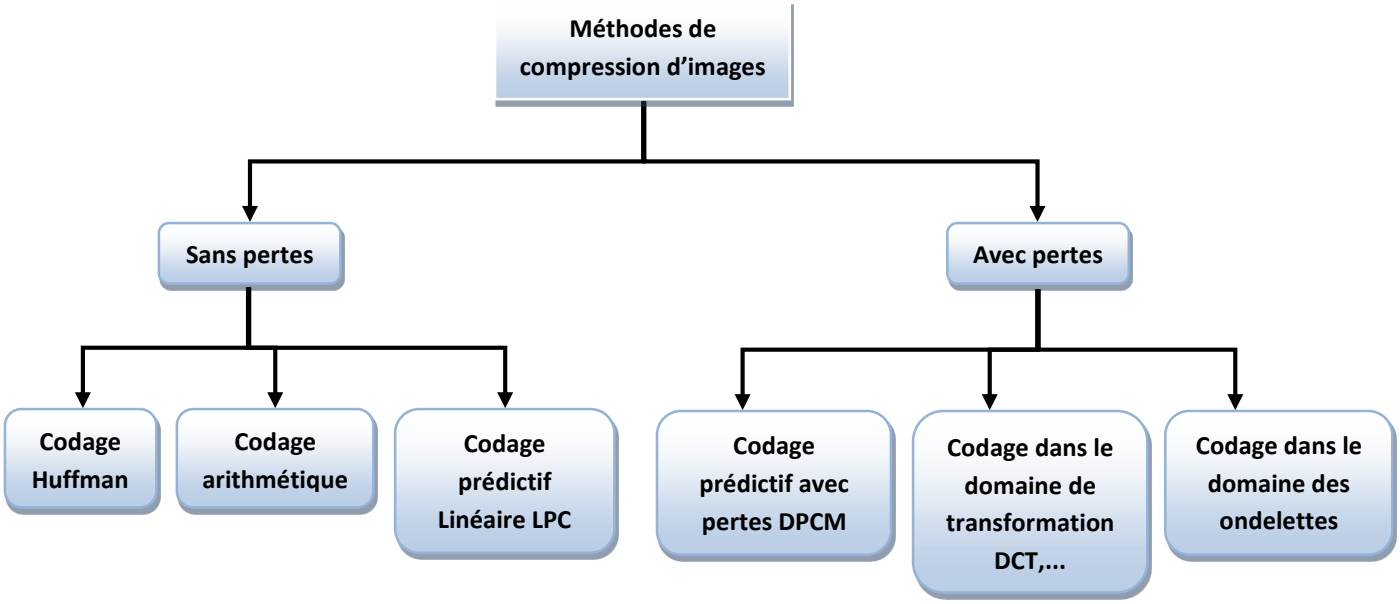



Figure 1.3. La taxonomie des méthodes de compression d'images.

Afin de pouvoir comparer ces méthodes, nous faisons appel à quatre critères globaux qui permettent d'évaluer les performances d'une méthode de compression :

- Le taux de compression qui mesure la réduction de la taille de l'image par rapport à la taille initiale.
- La distorsion mesure la qualité de l'image restituée par rapport à l'image originale dans le cas d'une méthode de compression avec pertes pour assurer le compromis débit-distorsion.
- La complexité de la méthode ou plus juste l'algorithme utilisé et sa facilité de mise en œuvre.
- Critères psychovisuels humain.

### 1.2.1. Taux de compression

Le taux de compression est le rapport entre la taille de l'image originale $(b_1)$ et la taille de l'image compressée $(b_2)$en bits (voir équation (1.1)). Le taux de compression d'une image est directement proportionnel à la quantité de redondance d'information qu'elle contient.

$$R_c = \frac{b_1}{b_2} \tag{1.1}$$

Par exemple, un taux de compression égal à 10 (10 :1) signifie que l'image originale à 10 unités d'information (en général bits) pour chaque unité dans l'image compressée.

A partir de ce rapport, le taux de compression en pourcentage $(T_c)$ est donné par l'équation (1.2) :

$$T_c = R \times 100 \tag{1.2}$$

Avec : $R = \left(1 - \frac{1}{R_c}\right)$ est la redondance des données entre l'image initiale $(b_1)$ et l'image compressée $(b_2)$ **[Marques 2011]**.

Pour le cas $b_1 = b_2$, $R_c = 1$ et $R = 0$, nous pouvons conclure que l'image originale ne contient pas des données redondantes.

Quand $b_2 << b_1$, $R_c \to \infty$ et $R = 1$, indique une forte redondance et par conséquent, une forte compression.

Finalement, quand $b_2 >> b_1$, $R_c \to 0$, et $R \to -\infty$, alors nous pouvons conclure que l'image compressée contient beaucoup plus de données par rapport à l'image originale, ce qui représente le cas indésirable.

En général, $R_c$ et $R$ se situent entre $[0, \infty[$ et $]-\infty, 1]$, respectivement.

Le taux de compression $R_c$ est relié au débit binaire qui permet de mesurer le nombre moyen de bits nécessaires pour représenter un échantillon unique **[Sayood 2012]**. Il est défini par l'équation (1.3) :

$$Debit = \frac{Nombre\ de\ bits\ par\ pixel\ dans\ l'image\ originale}{Rc}\ (bpp) \qquad (1.3)$$

Par exemple, pour une image compressée avec un taux de compression $R_c$ = 4 (4 :1). Si nous supposons que chaque pixel de l'image originale est codé sur 8 bits. Le nombre moyen de bits par pixel dans l'image compressée est 2. Ainsi, nous pouvons dire que le débit est 2 bits par pixel.

### 1.2.2. Distorsion et critère de qualité

Dans le cas d'une compression avec pertes, l'image reconstruite diffère de l'image originale. Pour cette raison et afin de déterminer l'efficacité de la méthode de compression choisie, nous devons quantifier la différence entre l'image originale et l'image reconstruite pour évaluer le compromis débit-distorsion.

Pour mesurer la qualité de l'image reconstruite, nous utilisons le $PSNR$ (Peak Signal to Noise Ratio) en décibels défini à partir de $MSE$ (Mean Square Error) par l'équation (1.4).

$$PSNR(I_0, I_c) = 20 \log_{10}\left(\frac{2^r - 1}{\sqrt{MSE(I_0, I_c)}}\right) \qquad (1.4)$$

Avec :

- $I_0$ et $I_c$ représentent respectivement l'image originale et l'image reconstruite de taille $M \times N$.
- $r$ est la résolution numérique de l'image.

Nous disposons d'autres critères qui nous permettent d'évaluer la déviation entre les pixels de l'image initiale est finale. Ces critères mesurent la ressemblance entre l'image initiale est finale après décompression. Les plus utilisés sont : l'erreur quadratique moyenne $MSE$, la racine de l'erreur quadratique moyenne $RMSE$ et l'erreur absolue moyenne $MAE$.

Pour mesurer la distorsion, nous utilisons l'erreur quadratique moyenne $MSE$ qui est donnée par l'équation (1.5) **[Delaunay 2008]** :

$$MSE = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}(I_0\,(i,j) - I_c(i,j))^2 \qquad (1.5)$$

Le $RMSE$ est la racine carrée de l'erreur quadratique moyenne.

$$RMSE = \sqrt{\frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}(I_0\,(i,j) - I_c(i,j))^2} \quad (1.6)$$

D'après **[Willmott and Matsuura 2005]**, le critère le plus important est l'erreur absolue moyenne $MAE$ car il est seulement en fonction des valeurs des pixels des images contrairement au $RMSE$ qui varie en fonction de la puissance au carrée des différences entre les pixels dans la racine carrée ce qui rend son interprétation difficile. Le $MAE$ est défini par **[Kumar and Rattan 2012]** :

$$MAE = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}|I_0(i,j) - I_c(i,j)| \quad (1.7)$$

### 1.2.3. La complexité

La complexité d'une méthode de compression est un critère difficile à évaluer. La complexité peut être définie par le temps d'exécution de la méthode de compression ou comme le nombre d'opérations par pixel **[Delaunay 2008]** : c'est le nombre moyen d'opérations qui sont nécessaires à la compression de l'image. La complexité englobe aussi le coût économique. Par exemple, pour implémenter la méthode sur un circuit électronique sur lequel le nombre de portes logiques est limité, il faut que la complexité soit la plus faible possible.

### 1.2.4. Critères psychovisuels humain

L'œil est un objet de forme d'un globe avec une lentille au centre qui fixe des objets sur la rétine. La rétine contient deux genres de récepteurs nommés les cônes et les bâtonnets. Les bâtonnets sont plus sensibles à la lumière que les cônes. Il y a trois genres de cônes. La sensibilité maximale des cônes est située dans les régions rouges, bleues, vertes du spectre de vision. Les cônes sont concentrés surtout dans des petites régions de la rétine appelées fovéa. Quoique les bâtonnets soient plus nombreux que les cônes, les cônes assure une meilleure

résolution car ils sont concentrés et serrés dans la fovéa. Les muscles de l'œil bougent le globe oculaire, positionne l'image de l'objet sur la fovéa. Cela devient désavantageux dans le cas d'une faible lumière. Dans ce cas-là, pour améliorer la vision d'un objet il faut focaliser un coté de cet l'objet. De cette manière, l'image de l'objet se forme sur les bâtonnets, qui sont plus sensible à la lumière.

Si nous illuminons par exemple un écran avec une certaine intensité $I$ et que nous éclairons cet écran par un spot d'intensité différente. Le spot ne devient visible qu'à partir d'une différence d'intensité $\Delta I$. Cette différence est nommée le seuil de perception *JND* (Just Noticeable Difference). Le rapport $\frac{\Delta I}{I}$ est connu comme la fonction de Weber ou le taux de Weber **[Sayood 2012]**.

Comment cette description du système de la vision humaine est reliée avec une méthode de compression ? Notons que le cerveau n'aperçoit pas tout ce que voit l'œil. Nous pouvons exploiter ce point pour construire des méthodes de compression pour rendre la distorsion introduite par la compression sans pertes imperceptible.

## 1.3. Les techniques de compression

Dans cette section, nous présentons une vue d'ensemble large des techniques qui constituent le cœur des méthodes de compression avec pertes et sans pertes les plus générales et qui sont actuellement utilisées.

### 1.3.1. Techniques de compression sans pertes

Les méthodes de compression sans pertes, comme leur nom l'indique, n'impliquent aucune perte d'informations. Si l'image a été compressée avec une méthode sans pertes, l'image originale peut exactement se restituer à partir de l'image compressée. La compression sans pertes est généralement utilisée pour des applications qui ne tolèrent aucune différence entre l'image originale et l'image reconstruire. Le taux de compression d'une compression sans pertes est généralement autour de 2:1 ce qui est insuffisant pour le stockage ou pour la transmission.

Avant de citer les différentes méthodes de codages sans pertes, nous devons avoir une connaissance rudimentaire sur la théorie de l'information.

### 1.3.1.1. La théorie de l'information

Une source d'information génère un symbole à un instant donné à partir d'une suite de symboles nommée l'alphabet source. Par exemple, une image en niveau de gris peut avoir 256 valeurs de gris et chaque valeur d'un pixel peut être considérée comme un symbole. Ainsi, dans ce cas l'alphabet source contient 256 symboles. Comme nous ne pouvons pas deviner d'avance quel symbole va être généré par la source, nous pouvons seulement décrire l'occurrence des symboles sources d'une manière probalistique. Ainsi, nous avons une source d'information qui génère un symbole à un instant donné à partir de son alphabet de symboles sachant que chaque symbole présente une certaine probabilité d'occurrence. La quantité d'information qu'apporte un symbole à un récepteur donné devra être non négative. Elle devra être grande si sa probabilité est petite, petite si sa probabilité est grande. Ceci s'accorde avec notre notion intuitive couverte par un message. Si nous entendons par exemple quelqu'un dire que le soleil se lève à l'Est, nous ferons la sourde oreille à ce propos parce que c'est certain que le soleil se lève seulement à l'Est. Donc il n'y a aucune information dans cette situation. D'un autre côté, si quelqu'un déclare qu'un météore s'approche de la Terre, alors nos soucis augmentent. Pourquoi, parce que la nouvelle n'est pas usuelle et en effet, c'est un événement rare donc il véhicule beaucoup d'information. Bien sûr, ces deux exemples verbaux sont sémantiques, mais nous nous s'intéressons seulement à la mesure quantitative de l'information pour la création des méthodes de compression.

#### a- Information propre

Supposons qu'une source d'information composée d'un alphabet de $M$ symboles $A = \{a_i\ , 1\ \leq i\ \leq M\}$ de probabilités $P(a_i) = p_i\ , 1 \leq i\ \leq M$. Alors, l'information propre $I_i$ associée au $i^{ème}$ symbole est définie comme : **[ISO/IEC 11172, 1993]** :

$$I_i = \log_2\left(\frac{1}{p_i}\right) = -\log_2(p_i)\ (bits) \qquad (1.8)$$

Comme $0 \leq p_i \leq 1$, l'information contenue dans un symbole est positive et augmente d'une manière monotone avec la diminution de sa probabilité.

L'équation (1.8) indique aussi qu'un symbole source avec une probabilité d'occurrence $p_i$ a besoin de $I_i$ nombre de bits pour sa représentation.

### b- Entropie d'une source

### 1- Source sans mémoire

Une source d'information ne vise pas de générer juste un symbole pour autant s'arrêter. En effet, elle génère une séquence de symboles, qui seront ensuite transmis vers un récepteur. Donc il est important de connaitre le nombre de bits moyen à utiliser pour représenter (ou coder) chaque symbole de la source. L'entropie d'une source est le nombre moyen de bits nécessaire pour coder les symboles d'une source. Supposons qu'une source d'information contient $M$ symboles de probabilités individuelles $p_i$ et $\sum_{i=1}^{M} p_i = 1$.

Supposons aussi que l'occurrence d'un symbole est indépendante des autres symboles. Une telle source fait référence à une source sans mémoire. L'entropie d'une source d'information sans mémoire est mathématiquement définie comme **[Thyagarajan 2011]** :

$$H = -\sum_{i=1}^{M} p_i \log_2 p_i \left({bits}/{symbole}\right) \quad (1.9)$$

L'entropie d'une source d'information est maximale lorsque les symboles sont équiprobables. Si une source contient $M$ symboles équiprobables, alors son entropie égale à $\log_2 M \, {bits}/{symbole}$ . Ainsi, l'entropie de la source est fixé par $0 \leq H \leq \log_2 M$.

La valeur maximale de l'entropie d'une source est souvent connue comme la capacité. La redondance d'une source est calculée en appliquant la différence entre la valeur maximale de l'entropie et celle de l'entropie.

Considérons comme exemple une source sans mémoire avec 8 symboles. Les probabilités des symboles sont 0.1, 0.2, 0.15, 0.3, 0.175, 0.02, 0.02 et 0.035. Calculons l'entropie de la source, la capacité et la redondance.

***Solution***

$H = -\sum_{i=1}^{8} \mathrm{p_i} \log_2(p_i) = 2.5633\ {}^{\mathrm{bits}}/_{\mathrm{symbole}}$. La capacité de la source est $C = \log_2 8 = 3\ {}^{bits}/_{symbole}$. Et par conséquent, la redondance est $0.4367\ {}^{bits}/_{symbole}$.

**2- Source avec mémoire**

Dans le cas d'une source d'information avec mémoire, l'occurrence d'un symbole à un instant donné est probablistiquement dépendante à celles des symboles aux instants précédents. En d'autres termes, il existe une dépendance statistique avec l'occurrence des symboles successifs. Cette propriété est utile pour réaliser une forte compression car nous groupement $K$ symboles au même temps à un instant donné au lieu de prendre seulement un symbole au même instant.

Considérons une source d'information qui génère une séquence d'alphabets $\{x_1, x_2, \ldots., x_n, \ldots\ldots\}$. Groupons maintenant $K$ alphabets au même temps pour former un symbole $x$ de probabilité $p(x)$. Ainsi, nous pouvons considérer tout vecteur $x$ comme une réalisation d'une variable aléatoire $X$. L'entropie des symboles de longueur $K$ est exprimée comme suit :

$$H_K(X) = -\frac{1}{K}\sum_{sur\ tout\ x} p(x) \log_2\big(p(x)\big)\ {}^{bits}/_{symboles} \qquad (1.10)$$

L'entropie de la source est obtenue comme la valeur limite de $H_x(X)$, qui est :

$$H(X) = \lim_{K\to\infty} H_K(X) \qquad (1.11)$$

### 1.3.1.2. Le codage Huffman

Le codage Huffman **[Huffman 1952]** est un code à longueur variable. Il est optimal pour une source qui possède un modèle de probabilité donné **[Mitchell et al 1996]**. Ici, l'optimalité indique que la longueur moyenne du code est proche de l'entropie de la source. Pour le codage Huffman, les symboles les plus probables reçoivent les mots-code les plus

petits et les symboles moins probables reçoivent les mots-code les plus longs. Ainsi, le codage Huffman peut nécessiter un dictionnaire (livre de mot-code) important. Le principe illustré par la figure 1.4 est le suivant :

1. Les symboles sont classés par ordre de probabilités décroissantes. Nous assignons alors aux deux symboles de probabilité les plus faibles les bits « 0 » et « 1 ».
2. Les deux symboles sont combinés en un symbole fictif dont la probabilité est la somme des probabilités des symboles élémentaires. Nous attribuons arbitrairement un nouveau bit « 0 « et « 1 » à chaque branche. Le fait que le code Huffman n'est pas unique, nous devons maintenir la même règle d'assignement des bits jusqu'à la fin.
3. La procédure 1) et 2) est réitérée jusqu'à ce que la liste ne comporte plus que deux éléments auxquels nous affectons les bits « 0 » et « 1 ». Ce qui forme donc la racine de l'arbre de Huffman.
4. Le code pour chaque symbole initial est alors obtenu en partant de la racine de l'arbre jusqu'aux feuilles.

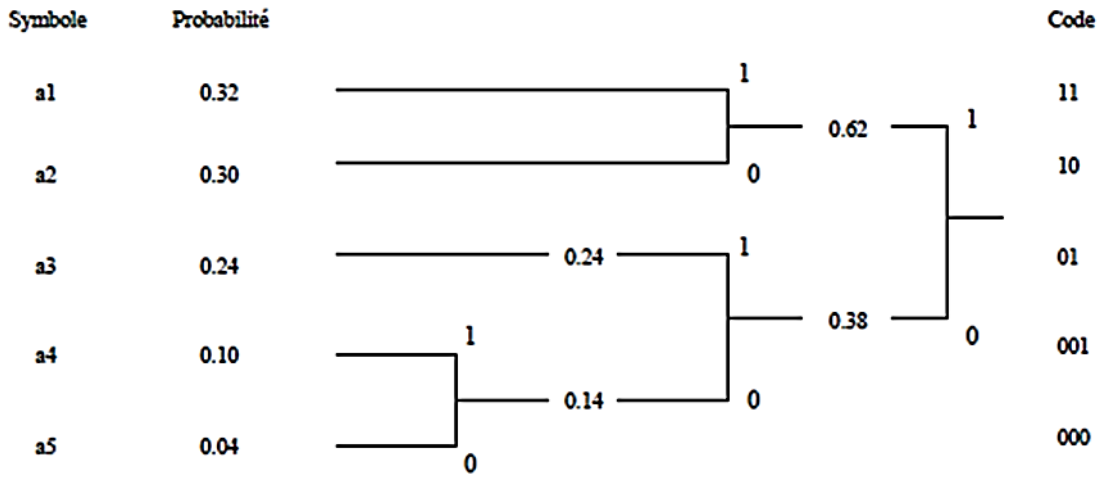



Figure 1.4. Principe du codage Huffman.
(Codage Huffman d'une source d'alphabets de cinq lettres).

Nous pouvons trouver la longueur moyenne du code $\overline{L}$ après le codage Huffman en appliquant l'équation suivante :

$\overline{L} = \sum_{i=1}^{5} l_i p_i = 2 * 0.32 + 2 * 0.30 + 2 * 0.24 + 3 * 0.10 + 3 * 0.04 = 2.14 \ {}^{bits}/_{symbole}$ .

L'entropie de cette source est calculée à partir de l'équation (1.9) ce qui donne $2.0592\ {}^{\text{bits}}/_{\text{symbole}}$.

L'efficacité du codage Huffman est définie comme le rapport entre l'entropie et la longueur moyenne du code, qui est $96.22\ \%$ pour cette source.

### 1.3.1.3. Le codage arithmétique

Le codage arithmétique est une méthode de codage sans pertes à longueur variable qui a connu un succès grandissant depuis les 20 dernières années **[Rissanen and Langdon 1979]**. Dans le cas du codage arithmétique, il n'y a aucune raison de mémoriser le dictionnaire. Comparons ceci avec le codage Huffman qui nécessite un dictionnaire pour mémoriser des mots-code de taille $K^M$ avec $K$ représente la taille de l'alphabet source et $M$ est la longueur du symbole. Pour une source de 4 lettres et une longueur de symbole égale à 8, le nombre des mots-code qui seront mémorisés dans le dictionnaire est $4^8 = 65536$ !

Le codage arithmétique est basé essentiellement sur le calcul la Fonction de Distribution Cumulative (*CFD*) des probabilités des symboles d'une séquence et de représenter ensuite les valeurs numériques du résultat en code binaire. La valeur numérique codée en binaire est nommée *identificateur*. L'utilisation de l'identificateur permet au décodeur de déchiffrer d'une manière unique la séquence codée.

#### a- Codage arithmétique d'une séquence de symboles

Le codage arithmétique est divisé en deux parties : génération de l'identificateur et représentation de l'identificateur comme une fraction binaire tronquée. Soit une source d'information avec un modèle de probabilité dont chaque symbole présente une certaine probabilité d'occurrence. La valeur de l'identificateur est située au milieu de l'intervalle ouvert $[0, 1)$ puisque la CFD est située entre « 0 » et « 1 ». Au début, l'intervalle unitaire est divisé selon le nombre des symboles de la source avec des longueurs correspondantes à leurs probabilités. D'une autre manière, l'intervalle unitaire est divisé tel que les intervalles successifs correspondent à la CFD des probabilités des symboles de la source. A l'arrivé du premier symbole, l'intervalle contenant le symbole est divisé selon le nombre des symboles

de la source en proportion de leurs probabilités et les autres intervalles sont écartés. Après l'arrivée du deuxième symbole, l'intervalle contenant ce dernier est divisé en proportion des probabilités des symboles. Ce processus est continué jusqu'au dernier symbole de la séquence. La valeur de l'identificateur est située dans le dernier intervalle. Toute valeur de cet intervalle peut être utilisée comme identificateur. Une possibilité de choix, d'ailleurs c'est le cas le plus fréquent, est d'utiliser le milieu du dernier intervalle comme identificateur. Une fois l'identificateur est déterminé, il est donc représenté comme une fraction binaire. Selon la valeur de l'identificateur, il sera peut être nécessaire de tronquer la fraction binaire pour avoir un nombre de chiffres binaires fini. Illustrons le processus du codage arithmétique par un exemple.

***Exemple***

Considérons une source d'information contenant quatre lettres dans son alphabet $A = \{a_1, a_2, a_3, a_4\}$ de probabilités 0.5, 0.3, 0.1 et 0.1 respectivement. La séquence à coder est $a_1 a_3 a_2 a_4 a_1$ . Nous devons générer l'identificateur et calculer ensuite sa fraction binaire.

***Solution***

Premièrement, nous divisons l'intervalle unitaire en proportion des probabilités à partir de la valeur zéro. Ainsi, nous trouvons les points limites à 0.5, 0.8, 0.9 et 1. Puisque le premier symbole dans la séquence est $a_1$, nous divisons l'intervalle $[0, 0.5)$ en proportion des quatre probabilités, comme le montre la figure 1. 5. A cet instant, l'identificateur réside dans l'intervalle $[0, 0.5)$. Les points limites après le premier symbole qui constituent aussi les bornes des sous intervalles sont 0.25, 0.4, 0.45 et 0.5.

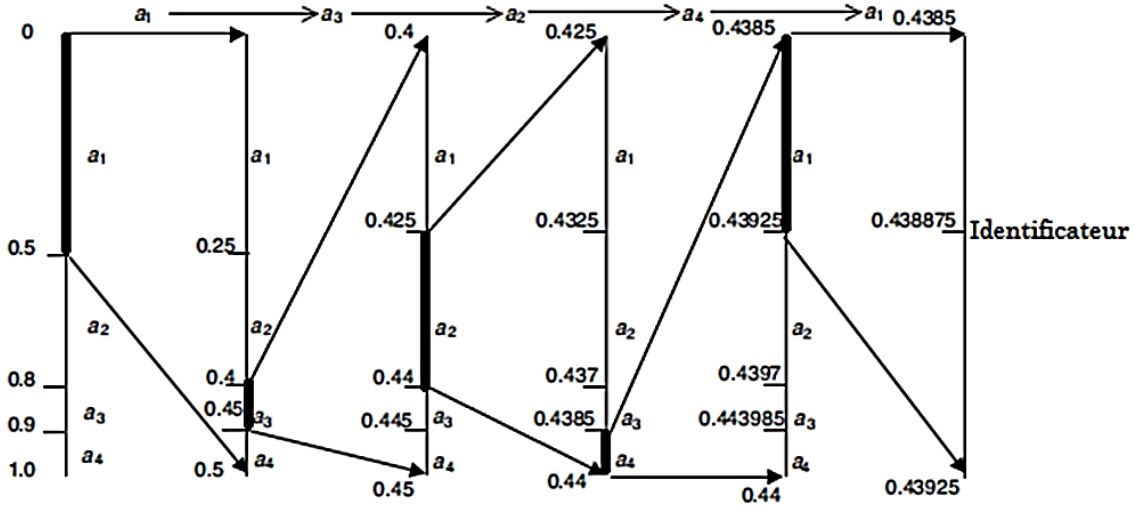



Figure 1.5. Codage arithmétique d'une séquence.

Comme le deuxième symbole est $a_3$, qui se trouve dans l'intervalle $[0.4, 0.45)$, nous devisons cet intervalle en proportion des probabilités. De nouveau, l'identificateur se trouve cette fois-ci dans l'intervalle $[0.4, 0.45)$. Nous répétons la même procédure, nous trouvons les intervalles d'identificateurs $[0.425, 0.440)$ , $[0.4385, 0.440)$ et $[0.4385, 0.43925)$ correspondent aux trois derniers symboles de la séquence dans cet ordre. Le dernier identificateur réside dans le dernier intervalle $[0.4385, 0.43925)$. Si nous décidons de choisir le milieu de ce dernier intervalle comme identificateur, nous obtenons la valeur de l'identificateur comme $0.438875$. Si nous représentons cette fraction décimale en une fraction binaire, nous obtenons $0.01110000010110$ ... Effectuons une troncation à cette fraction binaire pour prendre seulement 12 digits, l'identificateur correspondant à cette nouvelle valeur est $0.438720$. Ainsi, le code arithmétique pour la séquence de symboles de cinq lettres est les 12 bits de la fraction binaire $0.01110000010$. L'entropie de la source est $1.6855\ {bits}/{symbole}$ et la longueur moyenne du code arithmétique est $\frac{12}{5} = 2.4\ {bits}/{symbole}$. Dans ce cas particulier, l'efficacité est seulement $70.23\%$.

Nous observons à partir de la figure 1.5 que chaque intervalle successif est un sous-intervalle de l'intervalle précédent et qui est unique. Le seul problème est que la fraction devient de plus en plus petite en valeur et que nous avons besoin d'une grande précision pour représenter la valeur de l'identificateur en code binaire. Ceci est l'un des inconvénients du codage arithmétique.

Nous pouvons généraliser les procédures en haut par un comptage séquentielle des bornes supérieures et inférieures de l'intervalle où le courant symbole y il ait. Supposons que la séquence à coder est $\{a_1, a_2, a_3, \dots\dots\dots, a_n\}$. Choisissons l'identificateur comme le milieu de l'intervalle final, les bornes supérieures et inférieures de l'identificateur sont données comme suit :

1. $I_I(0) = 0\ et\ I_S(0) = 1$
2. $I_I(k) = I_I(k-1) + \big(I_S(k-1) - I_I(k-1)\big) \times CFD(m_k - 1)$
3. $I_S(k) = I_I(k-1) + \big(I_S(k-1) - I_I(k-1)\big) \times CFD(m_k)$
4. $Identificateur = \frac{I_S(k) + I_I(k)}{2}$

Dans l'exemple précédent, la séquence des symboles prend les valeurs $1, 3, 2, 4\ et\ 1$ dans cet ordre. Commençons par $I_I(0) = 0\ et\ I_S(0) = 1$, on a $m_1 = 1$ :

$$I_I(1) = 0 + (1 - 0) \times CFD(0) = 0$$
$$I_S(1) = 0 + (1 - 0) \times CFD(1) = 1 \times 0.5 = 0.5$$

Ensuite avec $m_2 = 3$ :

$$I_I(2) = 0 + (0.5 - 0) \times CFD(2) = 0.5 \times 0.8 = 0.40$$
$$I_S(2) = 0 + (0.5 - 0) \times CFD(3) = 0.5 \times 0.9 = 0.45$$

Pour le troisième symbole $m_3 = 2$, les bornes supérieures et inferieures sont :

$$I_I(3) = 0.4 + (0.45 - 0.40) \times CFD(1) = 0.425$$
$$I_S(3) = 0.4 + (0.45 - 0.40) \times CFD(2) = 0.440$$

Le quatrième symbole est $m_4 = 4$. Donc, les bornes sont :

$$I_I(4) = 0.425 + (0.44 - 0.425) \times CFD(3) = 0.4385$$
$$I_S(4) = 0.425 + (0.44 - 0.425) \times CFD(4) = 0.440$$

Le dernier symbole dans la séquence de message est $m_5 = 1$, nous obtenons donc les bornes du dernier intervalle :

$$I_I(5) = 0.4385 + (0.44 - 0.43855) \times CFD(0) = 0.4385$$
$$I_S(5) = 0.4385 + (0.44 - 0.43855) \times CFD(1) = 0.43925$$

L'identificateur est lié à ce dernier intervalle. Nous choisissons sa valeur comme le milieu de cet intervalle. Ainsi, $Identificateur = (0.43925 + 0.4385)/2 = 0.438875$. Le

nombre minimal des digits binaires nécessaire $\overline{L}$ pour représenter l'identificateur est déterminé à partir de l'équation suivante :

$$\overline{L} = \lceil -\log_2 P \rceil + 1 \qquad (1.12)$$

Dans l'équation (1.12), $P$ est le produit des probabilités des symboles dans une séquence et $\lceil x \rceil$ représente la fonction plafond (ceiling operation). Pour une source sans mémoire, $P$ est le produit des probabilités des symboles dans la séquence.

Pour l'exemple précédent, nous avons la séquence $a_1 a_3 a_2 a_4 a_1$, donc la valeur de $P$ est :

$P = 0.5 * 0.1 * 0.3 * 0.1 * 0.5 = 0.00075$. Donc la valeur de $\overline{L}$ est :

$\overline{L} = \lceil -\log_2 P \rceil + 1 = \lceil -\log_2 0.00075 \rceil + 1 = \lceil 10.3808 \rceil + 1 = 11 + 1 = 12$, d'où le choix de 12 digits pour représenter l'identificateur en fraction binaire.

**b- Décodage arithmétique d'une séquence de symboles**

Les procédures du codage arithmétique sont assez simples. Et pour le décodage ? Comme nous le verrons, la procédure du décodage arithmétique est aussi assez simple. La procédure de décodage peut être fixée comme suit :

1. Fixer les valeurs des bornes initiales comme : $I_I(0) = 0$ $et$ $I_S(0) = 1$. Mettre $k = 1$. Fixer $M$ comme la longueur de la séquence de symboles.
2. Tant que $k \leq M$, répéter les étapes suivantes.
3. Calculer $T_1 = \big(Identificateur - I_I(k-1)\big)/\big(I_S(k-1) - I_I(k-1)\big)$.
4. Déterminer $m_k$ tel que $CFD\ (m_k - 1) \leq T_1 \prec CFD(m_k)$.
5. Actualiser les bornes de l'intervalle courant en utilisant les équations

$$I_I(k) = I_I(k-1) + \big(I_S(k-1) - I_I(k-1)\big) \times CFD\ (m_k - 1)$$

Et

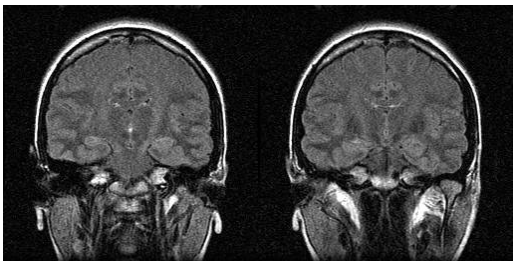
$$I_S(k) = I_I(k-1) + \big(I_S(k-1) - I_I(k-1)\big) \times CFD\ (m_k)$$

6. Incrémenter $k$.

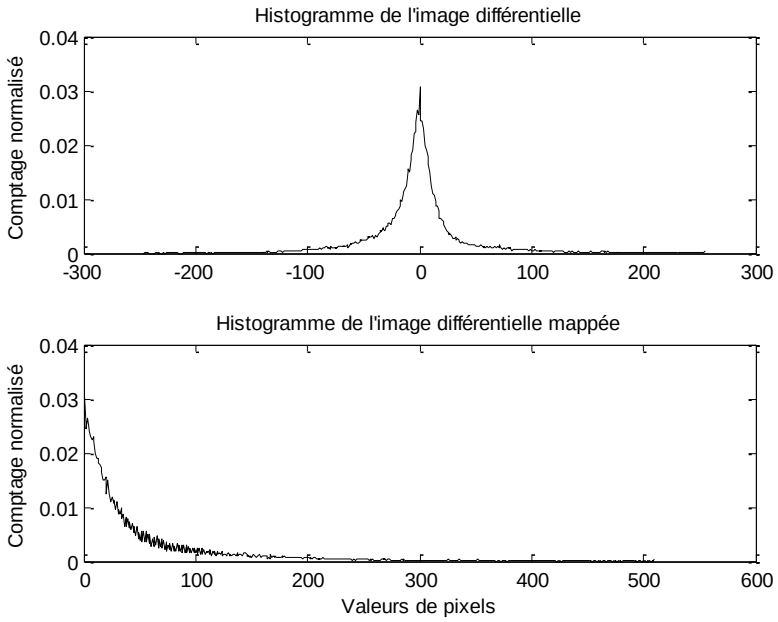
### 1.3.1.4. Le codage Golomb-Rice

En plus du codage Huffman et arithmétique, il existe un autre schéma du codage nommé codage de Golomb-Rice (GR) **[Rice and Plaunt 1971]** qui est simple et efficace pour le codage des entiers qui ont une distribution Laplaciénne **[Thyagarajan 2011]**. Contrairement au codage Huffman, il n'y a aucune raison de mémoriser les mots-code dans un dictionnaire. Dans le codage des images en mouvement, les pixels différentiels sont codés au lieu leurs valeurs réelles. Ces pixels différentiels ont typiquement une distribution exponentielle. La figure 1.6-a montre deux images consécutives (image 11 et 12) à partir de la séquence « mristack ». L'histogramme de l'image différentielle ($image\ 12 - image\ 11$) est montré par la figure 1.6-b où le graphe en haut est l'histogramme de l'image différentielle. Il est quasiment symétrique autour de zéro et il semble avoir une forme exponentielle.

Le codage GR encode des entiers positifs. Soit $n$ un entier positif à coder par le codage GR et soit $m = 2^k$, avec $k$ est un entier positif. Alors $n = Q \times m + R$ avec $Q = \left\lfloor \frac{n}{m} \right\rfloor$ représente le quotient et $R = n - mQ$ représente le reste. Le code GR de $n$ est obtenu par la concaténation du code unaire de $Q$ avec la représentation binaires du reste $R$ par les $k$ bits. Le code unaire de 3, par exemple, est trois zéros suivi par un « 1 », qui est 0001. Le « 1 » indique la fin du code unaire. A titre d'un exemple, supposons que $n = 7$ et $m = 2^2 = 4$. Alors, $Q = \left\lfloor \frac{7}{4} \right\rfloor = 1$. et $R = 3$. le code unaire de $Q$ est 01 et la représentation binaire des 2 bits de $R = 3$ est 11. Concaténons le code unaire et le code binaire, nous obtenons le code GR de 7 comme 0111.



(a)




(b)

Figure 1.6. Codage GR pour l'exemple « mristack ».

(a) Les deux images 11 (à gauche) et 12 (à droite) de la séquence « mristack ».
(b) Histogramme de l'image différentielle ($image\ 12 - image\ 11$) et l'histogramme de l'image différentielle mappée.

Bien que le code GR soit appliqué seulement aux entiers positifs, les entiers négatifs peuvent aussi être codés par le codage GR en les mappant tout d'abord vers des entiers positifs. Ce mappage est assuré par l'équation suivante :

$$M(n) = \begin{cases} 2n & n \geq 0 \\ 2|n| - 1 & n < 0 \end{cases} \qquad (1.13)$$

L'équation (1.13) mappe les entiers positifs en entiers pairs et les entiers négatifs en entiers impairs. Ainsi, il n'y a aucune raison d'utiliser un bit en plus pour indiquer le signe. Les procédures du codage GR peuvent être décrites comme suit **[Thyagarajan 2011]** :

- **Procédures du codage GR**

Soit $n$ un entier à coder et un $m = 2^k$ donné, avec $k$ est un entier positif.

1. Mapper $n$ en $\hat{n}$ en utilisant l'équation (1.13).
2. Calculer le quotient $Q = \left\lfloor \frac{\hat{n}}{m} \right\rfloor$ et le reste $R = \hat{n} - mQ$.
3. Concaténer le code unaire de $Q$ avec les $k$ bits du code binaire de $R$.

- **Procédures du décodage GR**

Soit un code GR d'un entier avec une valeur de $m$ donnée.

1. Calculer $k = \log_2 m$.
2. Compter le nombre de zéros en commençant par le bit le plus significatif jusqu'à la rencontre d'un « 1 ». Le nombre de zéros donne la valeur de $Q$. Calculer $P = m \times Q$.
3. Traiter les $k$ digits restant dans le code GR (après le premier « 1 » rencontré du code) comme la représentation binaire des $k$ bits de l'entier $R$, $\hat{n} = P + R$.
4. L'entier codé est donné par

$$n = \begin{cases} \dfrac{\hat{n}}{2} & si\ \hat{n}\ est\ pair \\ -\dfrac{\hat{n}+1}{2} & si\ \hat{n}\ est\ impair \end{cases}$$

Par exemple, si $m = 4$, et le code GR est 0000110, alors $Q = 4$, $P = 4 \times 4 = 16$, $R = 2$ et on a $k = \log_2 4 = 2$, donc la representation binaire de $R$ est 10, ce qui donne $R = 2$ et$\hat{n} = 16 + 2 = 18$. Par conséquent, $n = \frac{18}{2} = 9$. Par contre, si le code GR est 0000101, alors $\hat{n} = 16 + 1 = 17$ qui est un entier impair, donc $\frac{-(17+1)}{2} = -9$.

### 1.3.1.5. Le code Tunstall

La plupart des codages à longueur variable encodent les lettres d'un alphabet source en utilisant des mots-code avec un nombre de bits variable : mots-code avec peu de bits pour des lettres qui figurent plus fréquemment et des mots-code avec plus de bits pour des lettres qui figurent moins fréquemment. Dans le cas du codage Tunstall **[Tunstall 1967]** tous les mots-

code sont à longueur égale. Néanmoins, chaque mot-code représente un nombre différent de lettres. Un exemple d'un code Tunstall pour un alphabet de 2 bits $\beta = \{A, B\}$ est représenté dans le tableau 1.1. L'avantage majeur du code Tunstall est que les erreurs dans les mots-code ne sont pas propagées contrairement aux autres codes à longueur variable, comme le code Huffman où l'erreur dans un mot-code causera une série d'erreurs.

Tableau 1.1. Code Tunstall 2 bits

| Séquence | Mots-code |
|---|---|
| AAA | 00 |
| AAB | 01 |
| AB | 10 |
| B | 11 |

### 1.3.1.6. Le codage RLC (Run Length Coding)

Le code RLC est l'une des techniques de compression les plus simples. Il consiste à remplacer une séquence (Run) de symboles identiques par une paire contenant le symbole et la longueur de la séquence **[Devaki and Raghavendra 2012]**. Il est utilisé comme une technique de compression primaire dans le standard 1D CCITT Group 3 fax et par conjonction avec d'autres techniques dans le standard JPEG **[Marques 2011]**. Dans le cas d'une image noire et blanc, une séquence de zéros représente le nombre de zéros consécutives entre des uns. D'une manière similaire, une séquence des uns est le nombre des uns consécutifs entre des zéros.

### 1.3.1.7. Le codage prédictif linéaire LPC

Le codage prédictif linéaire semble être l'un des bons choix car il prend en considération la corrélation entre les pixels du voisinage **[Devi and Mini 2012]**. L'idée consiste à prédire la valeur du pixel et de coder ensuite la différence entre la valeur actuelle du pixel et sa valeur prédite. Cette différence représente la nouvelle information à coder. La figure 1.7 représente le codage LPC pour une image.

A partir de la figure 1.7, chaque valeur d'un pixel de l'image originale est prédite à partir des valeurs des pixels vus précédemment. Cette valeur prédite est ensuite soustraite du

pixel actuel. Cette différence est nommée la *prédiction résiduelle* et elle représente la nouvelle information qui sera codée d'une manière sans pertes par un codeur entropique pour la représenter ensuite par une séquence de bits compressées. Le décodeur reproduit la même prédiction à partir des prédictions précédentes et ajoute la prédiction résiduelle actuelle pour reproduire la valeur du pixel original **[Pearlman and Said 2011]**.

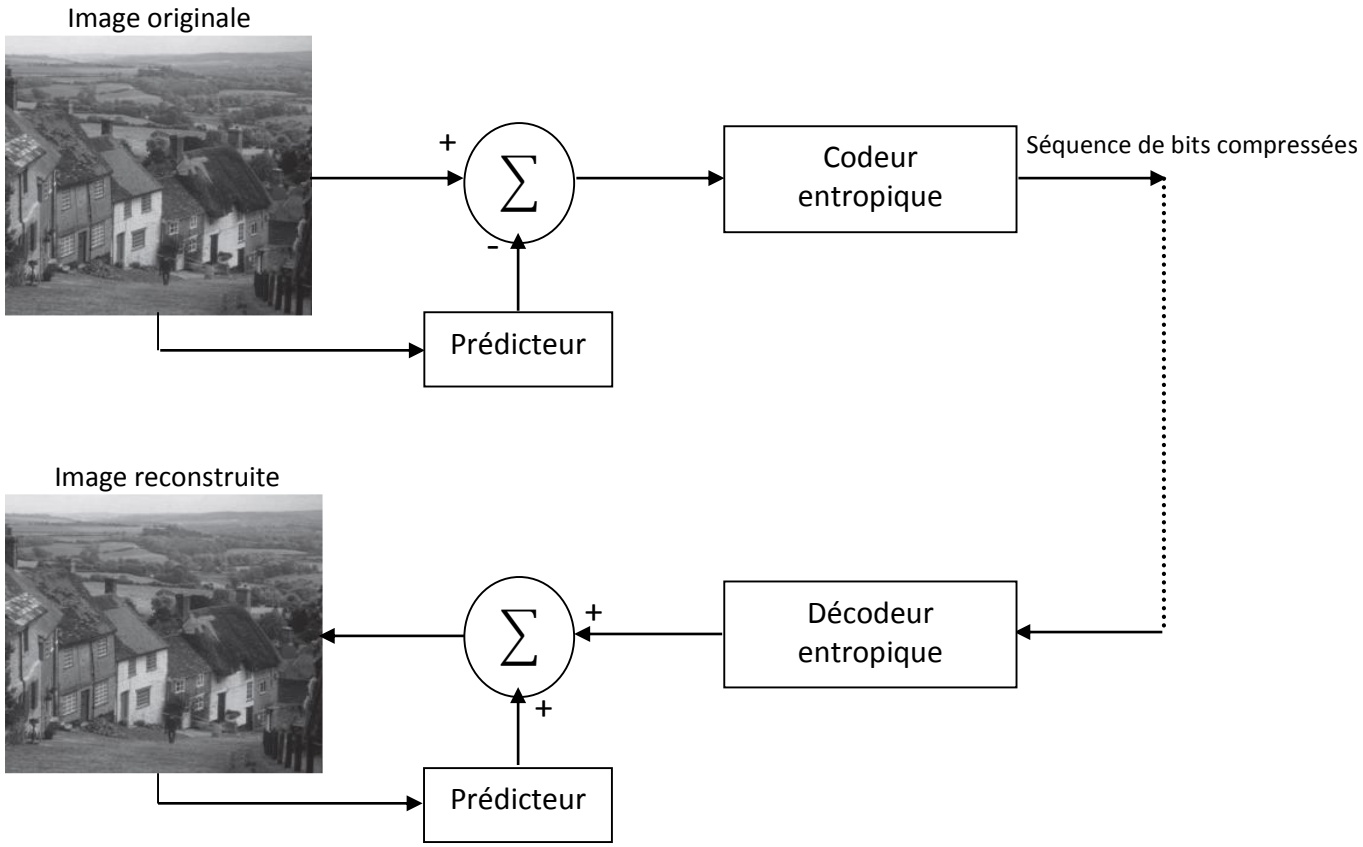



Figure 1.7. Codage/décodage prédictif linéaire LPC d'une image.

Mathématiquement, soit $f_n$ la valeur d'un pixel de l'image originale, $\widehat{f_n}$ représente la valeur prédite du pixel basée sur les valeurs des pixels précédents, $e_n$ est la prédiction résiduelle du pixel actuel.

$$e_n = f_n - \widehat{f_n} \tag{1.14}$$

Le décodeur reconstruit la prédiction résiduelle à partir les mots-code pour effectuer l'opération inverse.

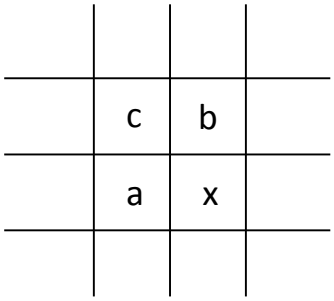
$$f_n = e_n + \widehat{f_n} \quad (1.15)$$

Pour générer la valeur prédite d'un pixel donné, nous faisons appel à une combinaison linéaire des $m$ pixels précédents **[Woods et al 2009]** :

$$\widehat{f_n} = round[\sum_{i=1}^{m} \alpha_i f_{n-i}] \quad (1.16)$$

Avec $m$ est l'ordre du prédicteur linéaire, « round » est la fonction utilisée pour arrondir la valeur de $\widehat{f_n}$ et $\alpha_i$ pour $i = 1, 2, \ldots\ldots, m$ sont les coefficients de prédiction.

La figure 1.8 donne le mode de prédiction utilisé dans le standard JPEG sans pertes pour un voisinage donné.




| № de prédiction | Prédiction |
|---|---|
| 0 | Aucune |
| 1 | A |
| 2 | B |
| 3 | C |
| 4 | a+b+c |
| 5 | a+ (b-c)/2 |
| 6 | b+ (a-c)/2 |
| 7 | (a+b)/2 |

Figure 1.8. Voisinages et mode de prédiction pour JPEG sans pertes.

### 1.3.2. Techniques de compression avec pertes

La compression avec pertes entraine quelques pertes d'information et les données qui ont été compressées d'une manière sans pertes ne seront pas fidèlement reconstruites. En acceptant cette distorsion lors de la reconstruction, nous pouvons généralement obtenir des taux de compression beaucoup plus élevés à ceux obtenus par les méthodes de compression sans pertes. Cette méthode est applicable surtout lors de stockage ou de la transmission du signal vocal où la valeur exacte de chaque échantillon reconstruit du signal vocal n'est pas nécessaire car l'essentiel est de comprendre le message transmis par téléphone par exemple.

D'une manière similaire, la vidéo est généralement compressée en utilisant une compression avec pertes. Nous allons citer les méthodes de compression avec pertes les plus utilisées actuellement.

### 1.3.2.1. La compression d'images dans le domaine de transformation

La compression dans le domaine de transformation est généralement basée sur la quantification des blocs de coefficients de la transformation suivie par un codage entropique des coefficients quantifiés **[ISO/IEC 11172, 1993]**. L'application d'une transformation unitaire à un bloc de pixels a pour objectif de décorréler les pixels et de compacter l'énergie résiduelle dans les pixels dans un nombre de coefficients réduit de sorte que seulement peu de coefficients significatifs qui seront codés.

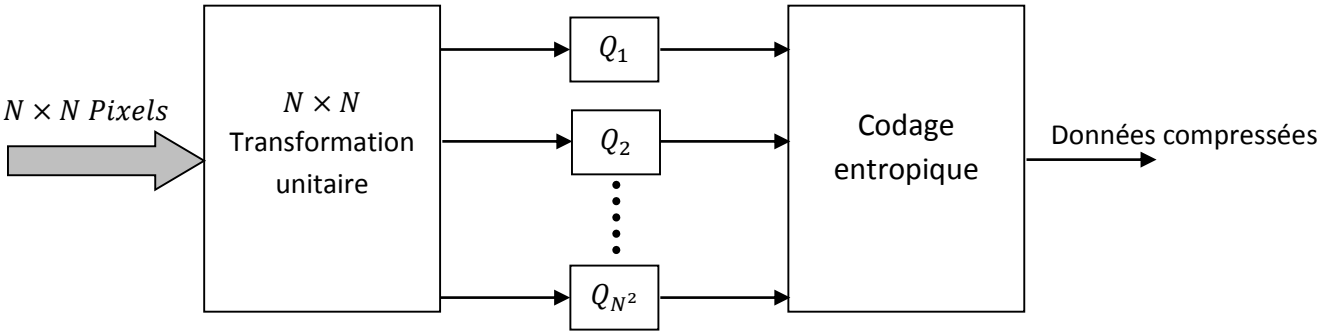



Figure 1.9. Schéma bloc du codage par transformation.

Le principe basique du codage par transformation comme le montre la figure 1.9 débute par une subdivision de l'image à coder en $N \times N$ blocs non chevauchés. Ainsi, une transformation unitaire particulière de taille $N \times N$ est appliquée à chaque bloc de pixel. Ensuite, les coefficients de la transformation dans chaque bloc sont quantifiés en un nombre de bits différent selon l'importance des coefficients. Comme les blocs quantifiés peuvent être constitués de longues séquences de zéros, le codage RLC des différentes séquences est trouvé pour coder ensuite les mots-code générés par un codage entropique pour transmission ou stockage. Les mêmes étapes sont appliquées dans l'ordre inverse lors du décodage.

La transformée la plus efficace est celle qui utilise le plus petit nombre de coefficients pour reconstruire l'image pour une quantité de distorsion donnée.

Nous citerons quelques transformées célèbres actuellement utilisées dans le domaine de compression d'images avec pertes.

### a- La transformée de Karhunen-Loève (KLT)

La transformée de Karhunen-Loève (KLT), connue aussi sous le nom de la transformée de Hotelling, est l'une des meilleures transformées utilisables pour réaliser une compression d'images. Elle permet d'éliminer complètement la corrélation entre les pixels dans le domaine de transformation. Ainsi, la matrice d'autocorrélation de l'image à transformée est diagonale. Néanmoins, la matrice de la KLT dépend de l'image et donc elle n'est pas assez convenable pour une compression en temps réel ni pour la standardisation **[Thyagarajan 2011]**.

### b- La Transformée Cosinus Discrète

La Transformée Cosinus Discrète (DCT) **[Ahmed et al 1974]** tire son nom du fait que les lignes de la matrice de la transformée $C$ de taille $N \times N$ sont obtenues en fonction du cosinus **[Sayood 2012]** :

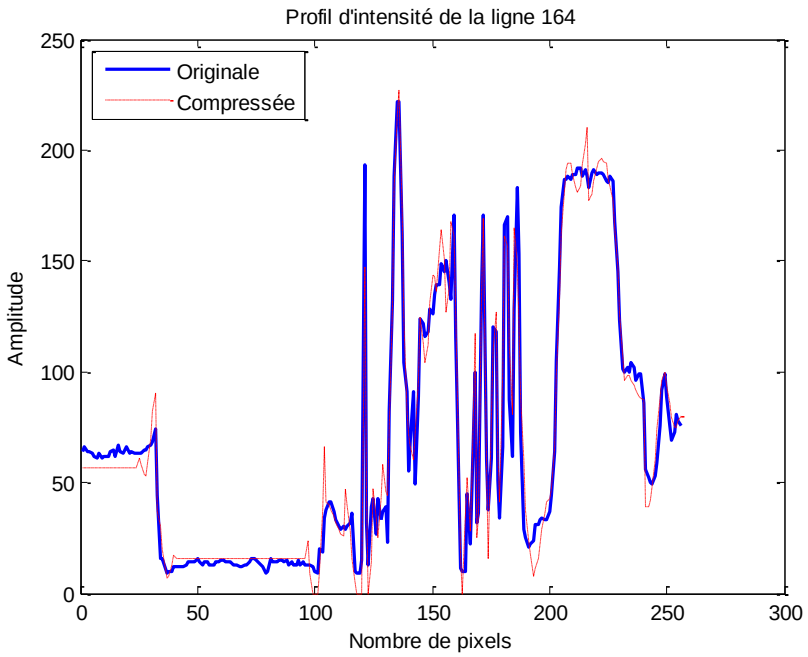
$$[\mathrm{C}]_{\mathrm{ij}} = \begin{cases} \sqrt{\frac{1}{N}} \cos\frac{(2j+1)i\pi}{2N} & i = 0, j = 0, 1, \dots\dots, N-1 \\ \sqrt{\frac{2}{N}} \cos\frac{(2j+1)i\pi}{2N} & i = 0,1, \dots\dots, N, j = 0, 1, \dots\dots., N-1 \end{cases} \tag{1.17}$$

La capacité de compression de la DCT est très proche de celle de la KLT. Elle est aussi presque égale à la KLT dans sa capacité de compaction d'énergie **[Pennebaker 1996]**. Contrairement à la KLT, la DCT est indépendante de l'image en question, c'est-à-dire, le noyau de sa matrice est fixé à une taille donnée. Donc, aucune information au côté du récepteur à propos de la taille des blocs n'est nécessaire pour la reconstruction de l'image originale. La DCT fait partie de nombreuses normes internationales y compris JPEG, MPEG et CCITT H.261 (Comité Consultatif International Téléphonique et Télégraphique (maintenant ITU-T)), entre autres.

L'un des inconvénients de la compression d'image en utilisant la DCT est l'effet mosaïque ou la pixellisation. Lors d'une quantification forte, la division en bloc devient visible car un bloc entier se trouve codé avec la même valeur (peu de coefficients non nuls).

La figure 1.10 montre un exemple d'effet mosaïque d'une DCT $8 \times 8$. Les blocs sont distincts d'une façon claire dans la figure 1.10-a. Les intensités faibles et fortes et leurs variations le long de la ligne 164 sont représentées dans la figure 1.10-b.


(a)




(b)

Figure 1.10. Effet de pixellisation.

(a) Image Cameraman montre l'effet de pixellisation dû à la quantification.

(b) profil d'intensité le long de la ligne 164 de l'image représentée par (a).

**c- La transformée de Radon**

La transformée de Radon permet de représenter une image dans le domaine de Radon comme une collection de projections tout le long des différentes directions pour chaque angle donné **[Radon 1917]**. En d'autres termes, la transformée de Radon mappe une ligne en un point dans le domaine de Radon ; une ligne particulière en un point particulier. La transformée de Radon sera étudiée en détail dans le deuxième chapitre car elle constituera la base de notre méthode de compression.

Beaucoup de travaux de compression d'images basées sur cette transformée ont été réalisés récemment, parmi ces travaux :

- A. Kingston et F. Autrusseau **[Kingston and Autrusseau 2008]** ont appliqué la transformée Mojette, qui est une transformée discrète exacte de la transformée Radon avec le codage DPCM (Differential Pulse Code Modulation) adaptatif pour la compression des images des musés en France dans le but de la réduction des bases de données.
- S. A. Pradeep et R. Manavalan **[Pradeep and Manavalan 2013]** ont opté la transformer de Radon et la DCT pour réaliser une compression d'images avec pertes qui permet d'assurer un bon résultat par rapport à la compression d'images basée sur la transformée d'ondelettes discrète.

**d- La transformée en fractal**

C'est une technique totalement différente conçue au milieu des années 80s et qui a connu un certain degré de succès. Le codage par fractal est basée sur les similarités existantes dans l'image **[Sayood 2012]**. Les similarités sont obtenues par des transformations géométriques. A présent, la qualité de reconstruction de l'image originale par l'approche de fractal est proche de celle obtenue par l'approche DCT employée lors de la compression JPEG **[M. Lahdir et al 2005]**.

### 1.3.2.2. La compression d'images dans le domaine des ondelettes

La transformée en ondelettes consiste à grouper des coefficients dans des sous bandes appartenant à des résolutions ou échelles avec une séparation de fréquence d'octave

**[Pearlman and Said 2011]**. La différence principale entre le codage par transformation et le codage dans le domaine des ondelettes est que la transformée en ondelettes est typiquement appliquée à l'image entière contrairement au codage par transformée où la transformée est généralement appliquée aux sous blocs de l'image. Lorsque nous parlons de la transformée d'ondelettes, nous désignons la transformée d'ondelettes discrète DWT. Les coefficients sont le résultat d'un filtrage avec des filtres FIR (Finite Impulse Response) de réponse d'impulsion finie agissants sur l'image originale. La majorité de l'énergie de l'image est compactée dans les sous bandes de basses fréquences. Un exemple des sous bandes à trois niveaux d'une DWT en deux dimensions est donné par la figure 1.11.

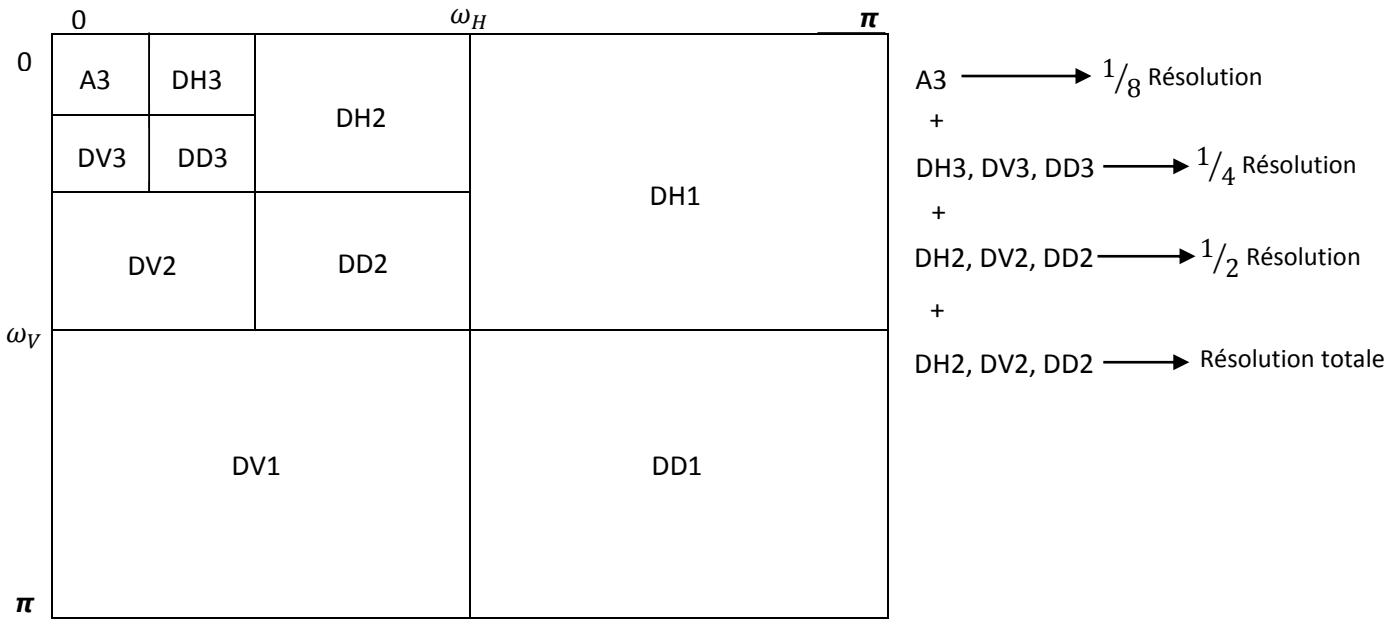



Figure 1.11. Décomposition d'une image en sous bandes.

Les bandes sont dénotées A3 pour les coefficients d'approximation, DH pour les coefficients de détails avec bords orientés horizontalement, DV pour les coefficients de détails avec bords orientés verticalement, et DD pour les coefficients de détails avec bords orientés diagonalement. Le nombre dans chaque bande indique le niveau de la DWT. Ainsi, A3 correspond aux coefficients d'approximation dans le niveau 3.

***Application à la compression d'images***

- S. Ameur et al. **[Ameur et al. 2002]** ont réalisé une compression d'images en sous bandes avec association d'une quantification vectorielle et la DCT. La sous bande lissée (basses fréquences) est compressée selon la norme JPEG.
- M. Lahdir et al. **[Lahdir et al. 2007]** ont exploité les ondelettes bi-orthogonales et les fractales pour la compression d'images satellitaires par un algorithme non itératif.

### 1.3.2.3. Le codage prédictif avec pertes DPCM

Le DPCM (Differential Pulse Code Modulation) est l'une des techniques basiques de compression d'images **[Chapin 1950]**. Les pixels d'une image sont fortement corrélés dans les voisinages rectangulaires. Une forte corrélation indique qu'une valeur d'un pixel est prédite à partir les valeurs des pixels de son voisinage. Donc, nous pouvons stocker ou transmettre les différences entre les pixels au lieu les pixels eux-mêmes pour réaliser une compression car les pixels différentiels sont très proche de zéro avec une forte probabilité et nécessitent moins de bits pour les coder **[Thyagarajan 2011]**.

Nous parlons de DPCM à une dimension (DPCM-1D) Si le prédicteur est construit de manière à éliminer seulement la corrélation entre les pixels des lignes ou des colonnes. Si nous exploitons la corrélation en deux dimensions pour avoir des résultats de compression meilleurs de ceux obtenus avec le DPCM-1D, alors nous utilisons un prédicteur à deux dimensions : DPCM-2D.

Nous aurons une bonne performance lorsque le quantificateur sera capable d'une façon ou d'une autre de s'adapter aux changements des paramètres statistiques de l'image à coder avec le DPCM. Nous utilisons dans ce cas le DPCM adaptatif (ADPCM) **[Tyagi and Sharma 2012]**.

### 1.3.2.4. La quantification

Dans la plupart des applications de compression avec pertes, nous sommes obligés à représenter chaque sortie d'une source par un nombre de mots-code réduit. Le nombre des valeurs distinctes possibles à la sortie d'une source est généralement beaucoup plus grand que le nombre des mots-code disponibles pour leurs représentations. Le processus de

représentation d'un large ensemble de valeurs par un ensemble réduit s'appelle la quantification. Si par exemple une source donnée génère des nombres entre $-10$ et $10$. La quantification consiste à représenter chaque sortie de la source par un entier proche à cette sortie. Si nous prenons par exemple $2.47$, il sera représenté par $2$. De même, si la sortie est $3.1254987$, sa représentation sera $3$. Cette approche réduit la taille de l'alphabet nécessaire à la représentation de la sortie de la source ; l'infinité des nombres entre $-10$ et $10$ sera remplacée par $21$ valeurs seulement ($\{-10, \dots\dots, 0, \dots\dots 10\}$) ce qui causera la perte d'information car si nous prenons par exemple la valeur $3$, nous pouvons pas dire est ce que cette valeur représente $3.025$ ou bien $2.99$ ou bien d'autres valeurs infinies de la sortie de la source. D'où le mot « avec pertes » dans beaucoup de méthodes de compression d'images. Les valeurs issues d'une quantification peuvent être des scalaires ou des vecteurs. Si sont des scalaires, nous parlons de la quantification scalaire. Si sont des vecteurs, alors nous parlons de la quantification vectorielle.

#### 1.3.2.4.1. La quantification scalaire

La quantification scalaire est l'application qui permet d'associer à un scalaire $x_i$ d'une séquence de pixels $U$ un autre scalaire $y_i$ d'un ensemble $Y = \{y_i, i = 1, \dots, L\}$ correspond au dictionnaire des mots-code. Ainsi, l'opération de quantification $Q(\cdot)$ est définie par :

$$U \to Y = \{y_1, y_2, \dots\dots, y_i, \dots\dots y_L\}$$

$$x_i \to y_i = Q(x_i)$$

Les valeurs $y_i$ sont appelées les valeurs de quantification. Dans la pratique, le nombre de représentants $L$ est defini et peut être représenté par une puissance de 2 tel que $L = 2^b$ **[C. Valade 2006]**. La figure 1.12 donne un exemple de quantification en 3-bits pour une séquence de pixels de $256$ nivaux de gris. Par exemple, toutes les valeurs entre $32$ et $64$ seront représentées par le même mot-code $001$.

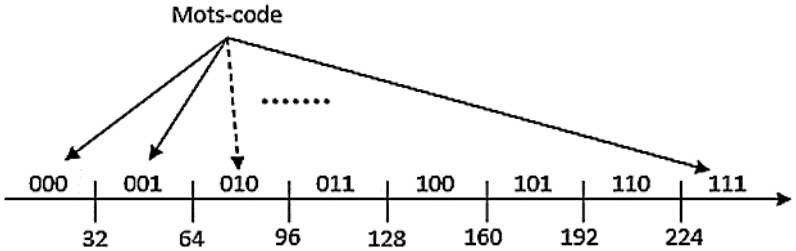



Figure 1.12. Quantification scalaire en 3-bits de 256 niveaux de gris.

#### 1.3.2.4.2. La quantification vectorielle

La quantification vectorielle assure la quantification simultanée de plusieurs coefficients. Elle est coûteuse en termes de complexité calculatoire. Dans le cas d'une quantification vectorielle, nous groupons les valeurs de sortie d'une source de données dans des blocs ou vecteurs. Nous pouvons prendre par exemple un bloc de *L* pixels à partir d'une image et de traiter chaque valeur d'un pixel comme une composante d'un vecteur de taille ou de dimension *L*. Ce vecteur des valeurs de sortie d'une source forme l'entrée de données pour le quantificateur vectoriel. Au niveau du codeur et du décodeur du quantificateur vectoriel, nous disposons d'un ensemble de vecteurs de *L* dimension nommé le dictionnaire du quantificateur vectoriel. Les vecteurs dans ce dictionnaire, connus comme vecteurs-code, sont sélectionnés pour être représentatifs des vecteurs que nous générons à partir de la sortie de la source. Un index binaire est assigné à chaque vecteur-code. Au niveau du codeur, le vecteur d'entrée est comparé à chaque vecteur-code afin de trouver le vecteur-code le plus proche au vecteur d'entrée qui est le vecteur de reproduction. Les éléments de ce vecteur-code sont les valeurs quantifiées de la sortie de la source. Afin d'informer le décodeur à propos de vecteur-code trouvé très proche du vecteur d'entrée, nous transmettons ou nous stockons l'index binaire du vecteur-code. Puisque le décodeur dispose du même dictionnaire, il peut récupérer le vecteur-code à partir de son index binaire. Une représentation illustrée de ce processus est donné par la figure 1.13 **[Sayood 2012]**.

Bien que le codeur peut exécuter un calcul considérable pour trouver le vecteur de reproduction le plus proche au vecteur de la sortie de la source, le décodeur dispose d'une table de recherche.

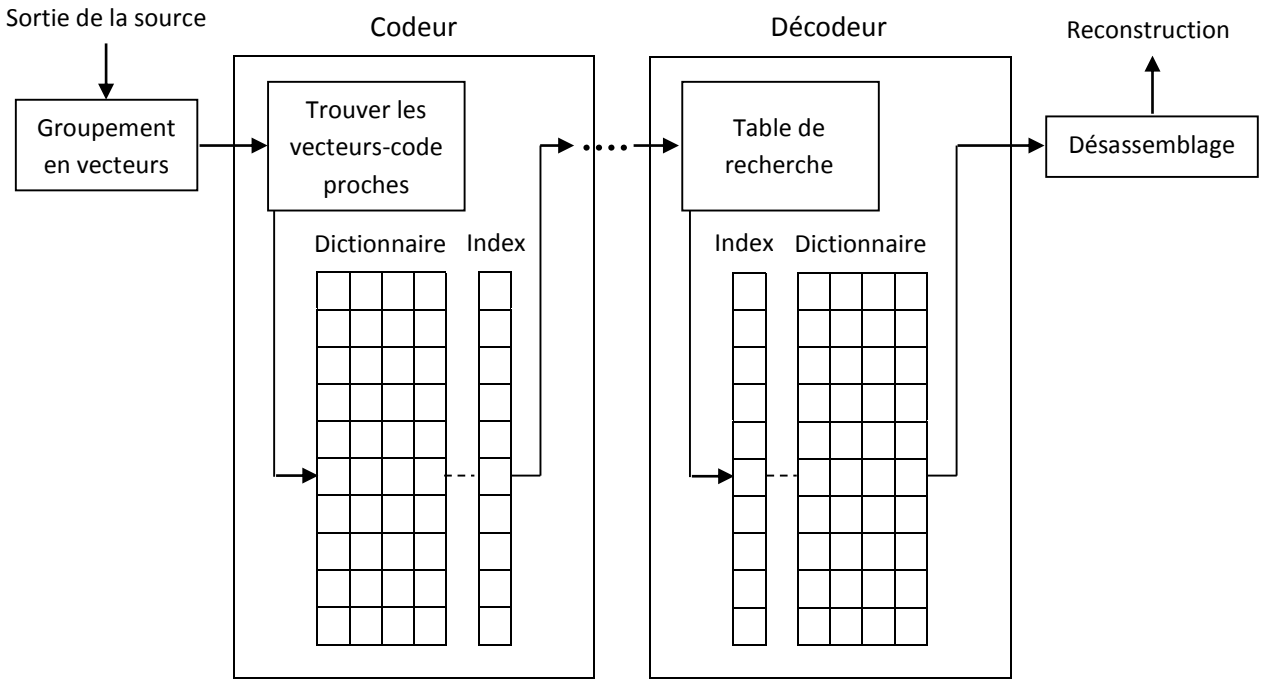



Figure 1.13. Processus de la quantification vectorielle.

## 1.4. Discussion

Dans ce premier chapitre, nous avons introduit la terminologie des méthodes de compression d'images sans pertes et avec pertes. Les méthodes de compression sans pertes les plus utilisées sont basées sur des codeurs comme : le codage Huffman, le codage arithmétique, et le codage prédictif linéaire LPC. Les méthodes de compression avec pertes sont le codage dans le domaine de transformation, le codage dans le domaine des ondelettes, et le codage prédictif avec pertes DPCM. Chacune de ces méthodes présente des avantages et des inconvénients. Nous avons détaillé notre étude pour les méthodes sans pertes par rapport aux méthodes avec pertes car ils forment, en général, le cœur de la compression avec pertes comme par exemple le codage des coefficients issus d'une transformée comme le cas des ondelettes. Avant d'entamer l'étude des méthodes de compression et de citer quelques exemples de leurs applications dans la standardisation comme le cas pour le JPEG, ou bien comme travaux récents, nous avons donné un aperçu sur la théorie de l'information et les différents critères qui nous permettent de juger l'efficacité des différentes méthodes de compression d'images. La transformée de Radon constitue la base de plusieurs méthodes de compression d'images développées récemment et qui ont donné des bonnes performances.

Cette transformée est un bon sujet de recherche pour le domaine de compression d'images. Elle constituera la base de notre méthode. La transformée de Radon sera étudiée en détail dans le second chapitre.

# Chapitre II

# La transformée de Radon

# Chapitre 2

# La transformée de Radon

## 2.1. Préambule

L'application d'une transformée unitaire à un bloc de pixels consiste à décorréler les pixels et de compacter leur énergie dans un nombre de coefficients réduit dans le but de coder seulement les coefficients significatifs. La transformée de Radon est l'une des transformée qui a trouvé beaucoup d'application dans le domaine de la compression d'images.

Nous allons présenter dans ce chapitre, la transformée de Radon dans sa morphologie mathématique ainsi que son application dans le domaine de l'imagerie.

## 2.2. Définition de la transformée de Radon

La transformée de Radon est une technique mathématique développée par le mathématicien Autrichien Johann Radon en 1917 **[Radon 1917]**. Cette transformée permet de convertir une fonction (image) de 2-D en une série de projections de 1-D pour tout angle $\theta \in [0,\ 2\pi]$. Une projection à un angle $\theta$ donné est obtenue comme l'intégration linéaire de la fonction sur toutes les lignes parallèles. L'unes des propriétés les plus importantes de la transformée de Radon est qu'elle est inversible d'où la possibilité de la reconstruction de la fonction projetée à partir de la connaissance de ses intégrations le long d'hyperplans de son espace **[Lohner 1983]**. Ainsi, la structure interne d'un objet peut être déterminée d'une manière non destructive à partir de ses projections. La transformée de Radon est fréquemment utilisée dans des domaines différents comme la tomographie, l'astronomie et la sismologie, et récemment dans le domaine de traitement d'images comme la compression et l'analyse.

## 2.3. De l'approche mathématique vers l'imagerie

### 2.3.1. Dans le domaine continu

Considérons $\hat{f} = Rf$ comme la transformée de Radon de la fonction $f$, à valeur dans $R$ definie dans $R^n$. Considérons un point de $R^n$ défini par son vecteur de position $\vec{r}$ de composantes $(x_1, x_2, \dots, x_n)$. Nous définissons un hyperplan $p = \vec{\zeta}\vec{r}$, avec $\vec{\zeta}$ un vecteur unitaire. La transformée de Radon de $f$ s'écrie comme **[Courmontagne 1998]** :

$$Rf = \hat{f}\left(p, \vec{\zeta}\right) = \int \dots \int f(\vec{r})\, \delta\left(p - \vec{\zeta}\vec{r}\right) d^n\vec{r} \qquad (2.1)$$

Avec $\delta$ représente la fonction de Dirac.

La transformée de Radon de la fonction $f$ représente l'intégration de cette dernière sur tous les hyperplans de l'espace.

Nous limitons à une étude sur $R^2$ où les hyperplans sont des droites. Dans ce cas, le vecteur unitaire $\vec{\zeta}$ aura comme composante $(\cos\theta\,, \sin\theta)$, avec $p = \vec{\zeta}\vec{r} = x\cos\theta + y\sin\theta$ qui représente l'équation définissant la droite $L$ passant par $P$ de coordonnées $(x, y) = (p\cos\theta\,, p\sin\theta)$ comme le présente la figure 2.1.

La transformée de Radon de la fonction $f$ sur une région d'intérêt $D$ de $R^2$ devient :

$$Rf = \hat{f}(p, \theta) = \int_L f(x, y)\, ds \qquad (2.2)$$

Avec $ds$ représente les variations élémentaires le long de la droite du plan $L$.

L'équation (2.2) permet de calculer l'intégrale de l'image $f(x, y)$ le long de la ligne $L$ qui est distante de $P$ à partir de l'origine pour un angle $\theta$ par rapport à l'axe $x$.

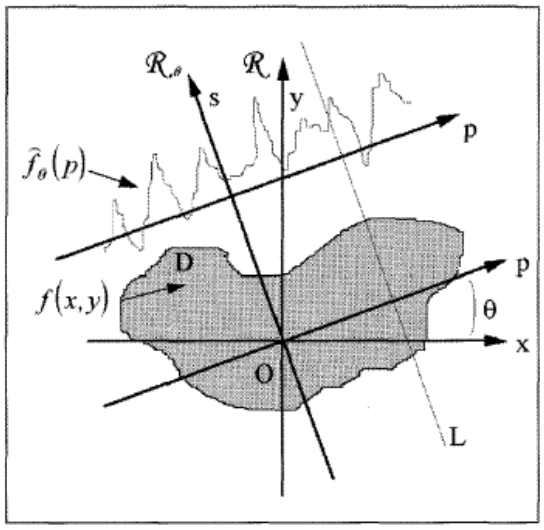



Figure 2.1. Représentation de la projection $\hat{f}_\theta(p)$.

Nous remarquons d'après la figure 2.1 la possibilité d'obtention d'un repère cartésien $\mathfrak{K}_\theta(O,p,s)$ à partir du repère $\mathfrak{K}_\theta(O,x,y)$ par un changement de base obtenu par une rotation du centre $O$ et d'angle $\theta \in [0,\pi]$. Nous pouvons donc exprimer $x$ et $y$ en fonction de $s$, $p$ et $\theta$ :

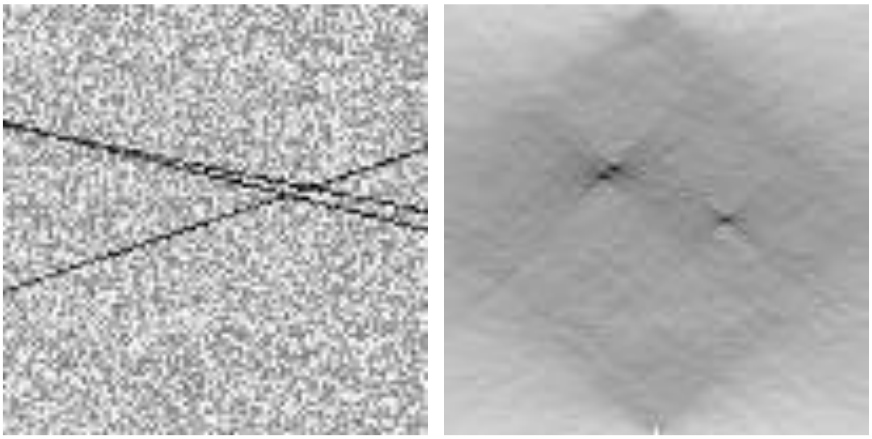
$$\begin{pmatrix} x \\ y \end{pmatrix} = \underbrace{\begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}}_{Matrice\ de\ passage} \begin{pmatrix} p \\ s \end{pmatrix}$$

L'utilisation de la fonction de Dirac $\delta$ nous permet de définir la transformée de Radon de $f$ sur $R^2$ comme suit :

$$\hat{f}(p,\theta) = \hat{f}_\theta(p) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} f(\vec{r})\,\delta(p - x\cos\theta - y\sin\theta)d\vec{r} \qquad (2.3)$$

Cette fonction représente la projection de $f$ suivant la droite $L$ sur une droite orthogonale à $L$ pour un angle $\theta$ fixe. Donc dans le domaine de Radon, une droite sera représentée par un point. Dans le cas de l'imagerie, des caractéristiques particulières dans une image comme des lignes seront représentées par des pics ou bien des regroupements de points dans le domaine de Radon. Cette caractéristique constitue la puissance de cette transformée et elle justifie son utilisation dans le domaine du traitement d'images.

La figure 2.2-a montre une image contenant trois lignes dont deux sont très proches avec un bruit aditif. La figure 2.2-b donne la transformée de Radon de l'image.



$(a)$ $(b)$

Figure 2.2. Détection de lignes par la transformée de Radon.

(a) Image bruitée avec trois lignes.
(b) La transformée de Radon de l'image : Chaque pic correspond à une ligne dans l'image.

Comme le montre la figure 2.2, la transformée de Radon peut résoudre un problème de détection difficile dans l'image originale par une simple étude des pics dans le domaine de Radon pour récupérer ensuite les paramètres des lignes par un seuillage.

Cette propriété de détection de ligne vient du fait que la transformée de Hough est simplement un cas particulier de la transformée de Radon **[Deans 1983]**.

La représentation graphique de la transformée de Radon donne un *Sinogramme* **[Fessler 2009]**. La figure 2.3 nous donne un exemple de sinogramme de l'image Lena.

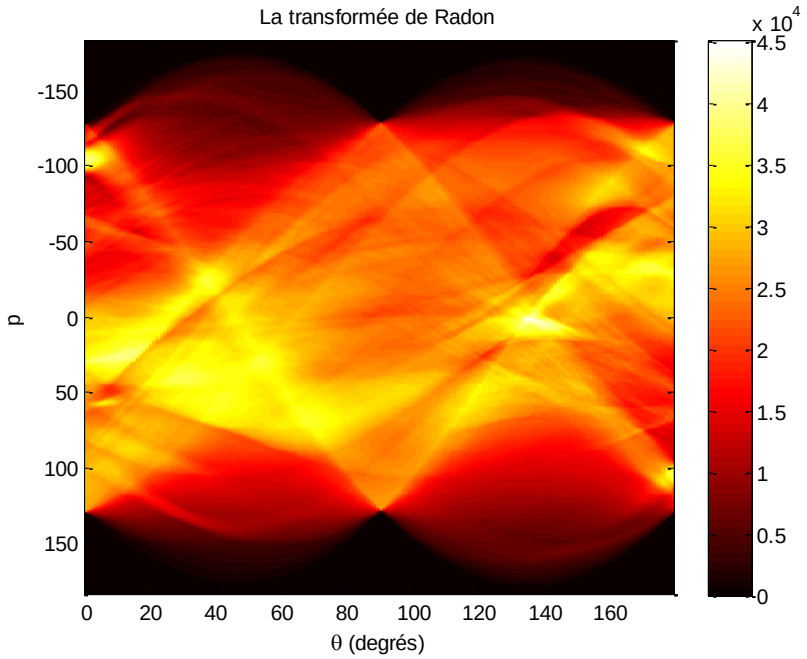



Figure 2.3. Sinogramme correspond à la transformée de Radon de l'image Lena.

Chaque point $(x, y)$ de l'image contribue par une sinusoïde unique d'amplitude $\sqrt{x^2 + y^2}$ dans le sinogramme. La distance du point entre l'origine et la phase de la sinusoïde dépend de l'angle entre $x$ et $y$. Un sinogramme est donc une superposition de toutes les sinusoïdes. Chaque sinusoïde correspond à la fonction $= x_0 \cos\theta + y_0 \sin\theta$ .

### 2.3.1.1. Propriétés de base de la transformée de Radon

Nous allons citer quelques propriétés de la transformée de Radon utiles dans le domaine du traitement d'images.

Prenons comme notation $f(x, y) \overset{Radon}{\longleftrightarrow} \hat{f}_\theta(p)$.

#### 2.3.1.1.1. Linéarité

Si $g(x, y) \overset{Radon}{\longleftrightarrow} \hat{g}_\theta(p)$, alors

$$\alpha f + \beta g \overset{Radon}{\longleftrightarrow} \alpha \hat{f} + \beta \hat{g}$$

#### 2.3.1.1.2. Décalage/translation

$$f(x - x_0, y - y_0) \overset{Radon}{\longleftrightarrow} \hat{f}_\theta(p - x_0 \cos\theta - y_0 \sin\theta)$$

Donc une translation d'une image de $(x_0, y_0)$ se traduit par une translation $p_0$ dans la direction de $p$.

#### 2.3.1.1.3. Rotation

Supposons que l'angle $\theta$ tourne d'une valeur $\theta'$, la transformée de Radon est calculée à partir de la transformée de Radon de $f(x, y)$.

$$f(x\cos\theta' + y\sin\theta', -x\sin\theta' + y\cos\theta') \overset{Radon}{\longleftrightarrow} \hat{f}_{\theta-\theta'}(p)$$

Si $\theta' = \pi$, alors

$$\hat{f}_\theta(p) = \hat{f}_{\theta-\pi}(p), \qquad \forall\, p \in \Re$$

#### 2.3.1.1.4. Symétrie/périodicité

$$\hat{f}_\theta(p) = \hat{f}_{\theta\pm\pi}(-p) = \hat{f}_{\theta\pm k\pi}((-1)^k p), \quad \forall k \in \mathcal{Z}$$

Il suffit donc de collecter des projections seulement pour une période $\pi$. Ce qui nous permet de gagner énormément du temps.

Notons que dans le cas d'une reconstruction tomographique, il est nécessaire d'effectuer une rotation sur $2\pi$ pour étudier des structures plus profondes **[Dubois 1998]**.

#### 2.3.1.1.5. Changement d'échelle

$$f(\alpha x, \alpha y) \overset{Radon}{\longleftrightarrow} \frac{1}{|\alpha|}\hat{f}_\theta(\alpha p), \qquad \alpha \neq 0$$

#### 2.3.1.1.6. Résistibilité aux bruits

Un autre avantage de la transformée de Radon est sa robustesse vis-à-vis des bruits comme le bruit blanc **[Murphy 1986].**

#### 2.3.1.1.7. Convolution

Si $f(x,y) \overset{Radon}{\longleftrightarrow} \hat{f}_\theta(p)$ et $g(x,y) \overset{Radon}{\longleftrightarrow} \hat{g}_\theta(p)$, alors

$$f(x,y) ** g(x,y) \overset{Radon}{\longleftrightarrow} \hat{f}_\theta(p) * \hat{g}_\theta(p)$$

#### 2.3.1.1.8. La transformée de Radon inverse/Reconstruction

L'une des propriétés constituant la puissance de la transformée de Radon est l'inversion. Cette propriété nous permet de reconstruire l'objet projeté par une rétroprojection (backprojection) comme le cas d'une tomographie. L'une des idées exploitées pour tenter à récupérer un objet à partir de $\hat{f}_\theta(p)$ est d'épandre dans l'espace de l'objet les projections. Donc, étaler chaque valeur du sinogramme dans l'espace de l'objet le long du rayon correspondant comme le montre la figure 2.4.

Puisque chaque point $(x_0, y_0)$ objet contribue par sa propre sinusoïde dans le sinogramme, il est normal que la somme le long de la sinusoïde nous permet de récupérer la valeur $f(x_0, y_0)$. Malheureusement dans sa forme simple, cette procédure ne nous permet pas de récupérer l'objet de départ $f(x,y)$, mais sa version floue $f_f(x,y)$. Cette version floue connue sous le nom de *Laminogramme* **[Fessler 2009]**, est causée par du fait que l'épandage s'effectue dans des zones où il n'existe pas des objets à reconstruire dans l'image.

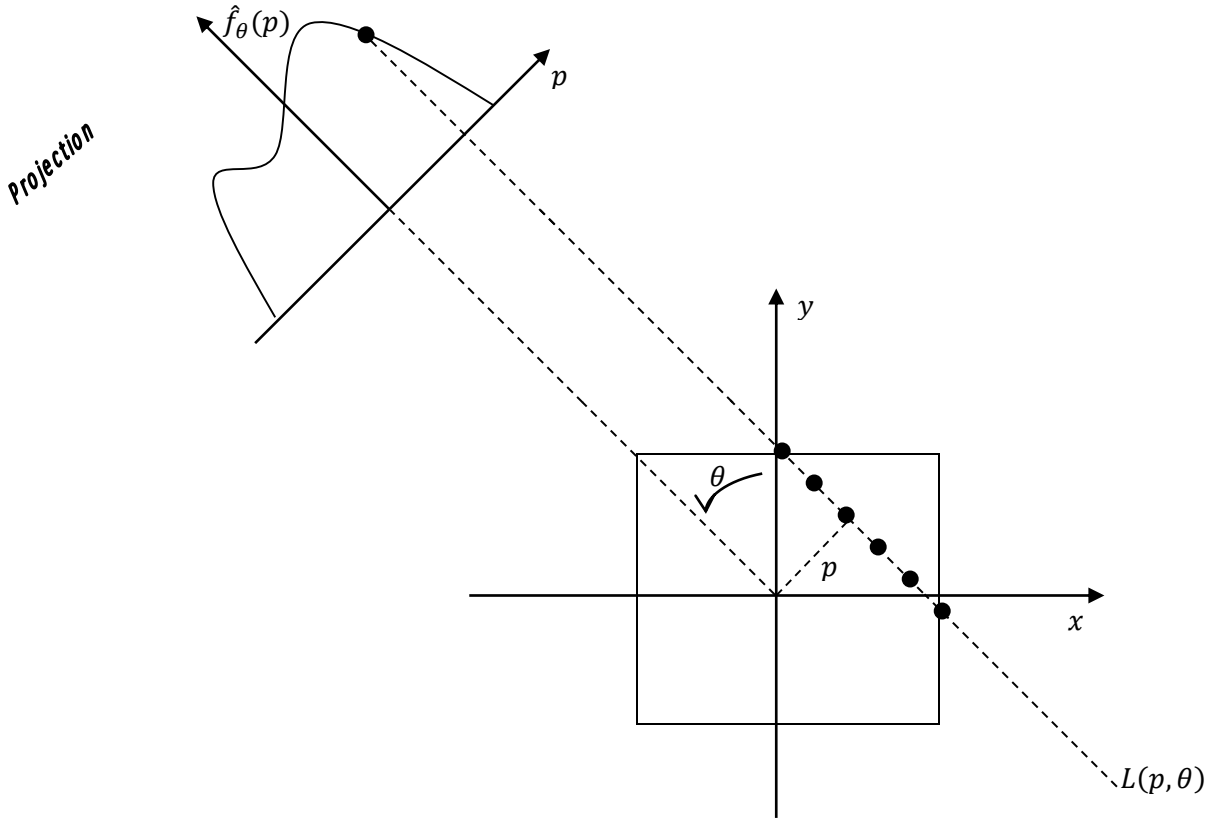



Figure 2.4. Illustration de l'opération de la rétroprojection d'un point objet.

Mathématiquement, la version floue $f_f(x,y)$ est donnée par la relation :

$$f_f(x,y) = \int_0^{\pi} \hat{f}_\theta(x\cos\theta + y\sin\theta)d\theta \qquad (2.4)$$

Nous allons par la suite citer deux méthodes de reconstruction. Ces méthodes sont : la reconstruction Fourier directe basée sur le théorème de coupe centrale ou section centrale (Fourier-slice theorem), et la rétroprojection des projections filtrées FBP (Filter-BackProject). Avant de décrire ces méthodes, nous devons faire un passage par le théorème de coupe centrale.

#### 2.3.1.1.8.1. Théorème de coupe centrale

L'énoncé de ce théorème est comme suit : si $\hat{f}_\theta(p)$ est désignée comme la transformée de Radon de $f(x,y)$, alors la transformée de Fourier à une dimension 1-D de $\hat{f}_\theta(\cdot)$ égale à la section ou à la coupe à l'angle $\theta$ obtenue par la transformée de Fourier 2-D de $f(x,y)$ comme le montre la figure 2. 5.

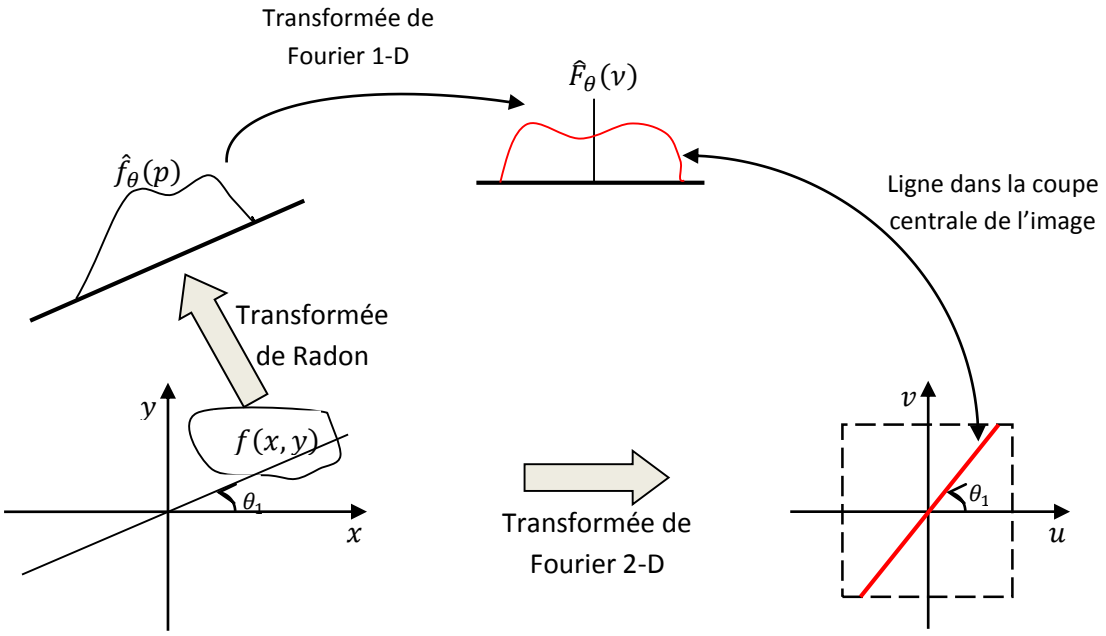



Figure 2.5. Théorème de la coupe centrale.

Donc la transformée de Fourier d'une projection à un angle $\theta$ correspond à une ligne passant par l'origine de la transformée de Fourier 2-D de l'image pour le même angle.

Soit $\hat{F}_\theta(v)$ la transformée de Fourier 1-D de $\hat{f}_\theta(p)$, alors :

$$\hat{F}_\theta(v) = \int_{-\infty}^{+\infty} \hat{f}_\theta(p)\, e^{-i2\pi vp}\, dp$$

Posons $F(u,v)$ la transformée de Fourier 2-D de $f(x,y)$, donc :

$$F(u,v) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} f(x,y) e^{-i2\pi(ux+vy)}\, dxdy$$

Ainsi, mathématiquement, le théorème de coupe centrale donne la relation suivante:

$$\hat{F}_\theta(\nu) = F(\nu\cos\theta\,, \nu\sin\theta) = F_0(\nu,\theta),\ \forall\nu \in \Re,\ \forall\theta \in \Re \qquad (2.5)$$

Avec $F_0(\nu,\theta)$ représente la forme polaire de $F(u,v)$.

Nous remarquons d'après ce théorème que la relation entre la transformée de Radon et la transformée de Fourier 2-D donne la possibilité de reconstruire n'importe quel objet par une application d'une transformée de Fourier 2-D inverse (reconstruction Fourier directe). **[Fessler 2009]**.

### a- Reconstruction Fourier directe

La méthode de reconstruction Fourier directe est basée directement sur le théorème de coupe centrale. Pour reconstruire $f(x,y)$ à partir de l'ensemble de ses projections $\hat{f}_\theta(p)$ par la méthode de Fourier directe, nous passons par les étapes suivantes:

- Calculer la transformée de Fourier 1-D de chaque $\hat{f}_\theta(\cdot)$ pour récupérer $\hat{F}_\theta(\cdot)$ pour chaque $\theta$.
- Trouver la représentation polaire $F_0(p,\theta)$ de la transformée de Fourier 2-D de l'objet $F(u,v)$ en utilisant la relation du théorème de coupe centrale : $F_0(p,\theta) = \hat{F}_\theta(p)$.
- Convertir la représentation polaire $F_0(p,\theta)$ vers la représentation Cartésienne $F(u,v)$.
- Effectuer la transformée de Fourier 2-D inverse de $F(u,v)$ pour obtenir $f(x,y)$.

Dans le cas pratique, nous utilisons la FFT-2D inverse (Fast Fourier Transform) pour la $F(u,v)$ échantillonnée. Bien que la relation $F_0(p,\theta) = \hat{F}_\theta(p)$ est intrinsèquement polaire d'où le besoin d'une interpolation.

Cette méthode donne des bons résultats pour une projection $\hat{f}_\theta(p)$ sans bruit. Le désavantage de cette méthode réside dans la nécessité de la transformée de Fourier 2-D et que le maillage, comme le montre la figure 2.6 peut causer des artefacts d'interpolation **[Herman 1972]**.

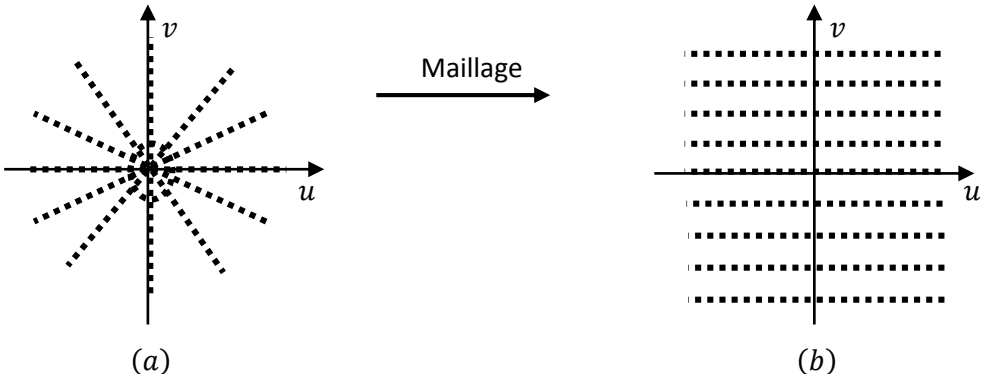



Figure 2.6. Illustration des artefacts d'interpolation.
(a) Grille polaire dans le domaine de Fourier.
(b) Grille cartésienne dans le domaine de Fourier.

**b- La rétroprojection des projections filtrées FBP**

Cette technique comme le montre son schéma bloc consiste à filtrer chaque projection de l'image $\hat{f}_\theta(\cdot)$ par un filtre 1-D de réponse fréquentielle $|\nu|$. Ce filtre est le filtre rampe.

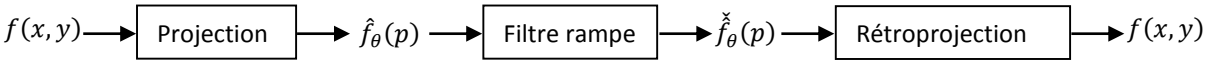



Figure 2.7. Schéma bloc de la méthode FBP.

La projection filtrée $\check{\hat{f}}_\theta(p)$ est définie par l'intégrale suivante :

$$\check{\hat{f}}_\theta(p) = \int_{-\infty}^{+\infty} \hat{F}_\theta(\nu)\,|\nu| e^{i2\pi\nu p} d\nu \qquad (2.6)$$

Les étapes de la méthode FBP sont résumées comme suit :

- Pour chaque angle de projection $\theta$, calculer la transformée de Fourier 1-D de la projection $\hat{f}_\theta(p)$ pour obtenir $\hat{F}_\theta(\nu)$.
- Multiplier $\hat{F}_\theta(\nu)$ par$|\nu|$ (filtre rampe).
- Pour chaque $\theta$, calculer la transformée de Fourier 1-D inverse de $|\nu|\hat{F}_\theta(\nu)$ pour obtenir la projection filtrée $\check{\hat{f}}_\theta(p)$ en appliquant l'équation (2.6). En pratique, puisque le filtre rampe de réponse fréquentielle $|\nu|$ n'est pas limité, nous utilisons sa version

limitée en bande de fréquence à une fréquence spatiale $\nu_0$ donnée. Ce filtre porte le nom du filtre Ram-Lak **[Ramachandran and Lakshminarayanan 1971]**.

- Rétro-projeter le sinogramme filtré $\check{\check{f}}_\theta(p)$ en utilisant l'équation (2.4) pour obtenir $f(x,y)$:

$$f(x,y) = \int_{-\infty}^{+\infty} \check{\check{f}}_\theta(p)\,(x\cos\theta + y\sin\theta)d\theta$$

L'avantage de la technique FBP consiste à exploiter seulement la transformée de Fourier 1-D sans faire le passage par la transformée de Fourier 2-D ce qui permet de limiter les artefacts lors de l'interpolation.

Le filtre Ram-Lak est défini comme suit :

$$H(\nu) = \begin{cases} |\nu| & \text{Si } -\nu_{max} < \nu < \nu_{max} \\ 0 & \text{Sinon} \end{cases}$$

Le filtre Ram-Lak est montré par la figure 2.8.

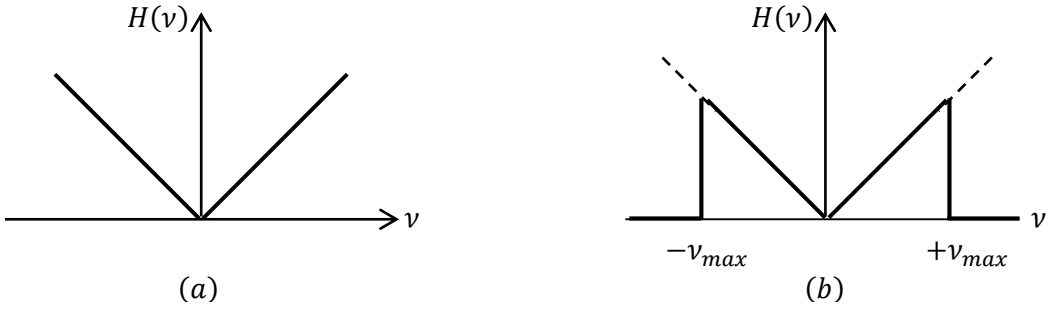



Figure 2.8. Réponse fréquentielle du filtre rampe.
(a) Filtre rampe idéal.
(b) Filtre Ram-Lak limité à $|\nu|$.

Le filtre Ram-Lak permet d'amplifier les hautes fréquences pour générer des transitions rapides dans le signal. Ce processus permet d'introduire des valeurs négatives dans le but d'annuler progressivement les artefacts de projection représentés par des valeurs positives. Les valeurs négatives viennent du fait que le filtre Ram-Lak met à zéro la moyenne du signal et donc, la composante continue **[Dubois 1998]**.

La figure 2.9 illustre la différence entre une simple rétroprojection et une rétroprojection filtrée par le filtre Ram-Lak de l'image Lena.

(a)

(b)

Figure 2.9. Rétroprojection simple et filtrée de l'image Lena.

(a) Rétroprojection simple.
(b) Rétroprojection des projections filtrées $FBP$.

### 2.3.2. Dans le domaine discret

Comme les données projetées et rétroprojetées d'une image sont discrètes, l'implémentation de la transformée de Radon et son inverse devront être discrétisées. Plusieurs méthodes imposent le filtrage et l'interpolation des données. Il existe aussi des transformées de Radon discrètes qui s'opèrent directement avec des données de nature discrète.

#### 2.3.2.1. La transformée de Radon avec interpolation

Nous savons que l'application de la technique FBP permet de récupérer l'image projetée. Malheureusement, l'image rétroprojetée présente un maillage diffèrent de celui de départ d'où la nécessité d'une étape d'interpolation numérique pour récupérer le maillage natif **[Courmontagne 1998].** Cela est dû, comme le représente la figure 2.10 à l'effet de rotation de l'image lors du calcul de la transformée de Radon.

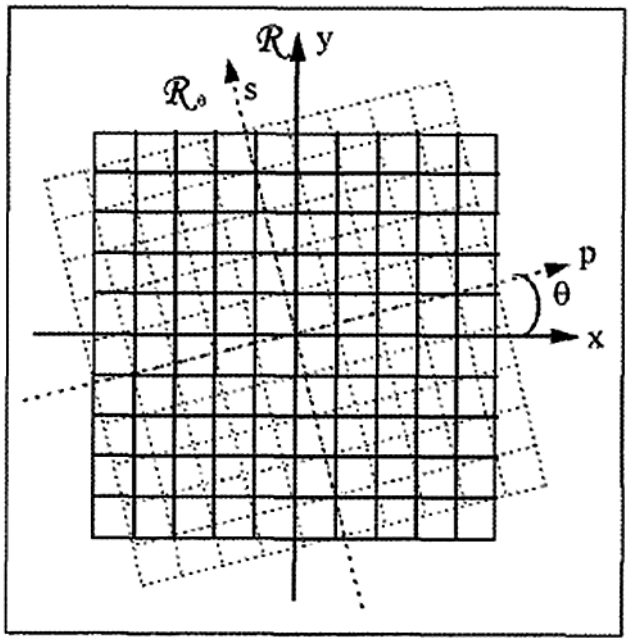



Figure 2.10. Image dans le repère en rotation.

Nous remarquons d'après la figure 2.10, que les nouvelles coordonnées de l'image dans le repère en rotation ne sont pas des valeurs entières et donc sont différentes de celles du maillage natif de l'image. Nous allons étudier cette technique en détail lors du troisième chapitre.

#### 2.3.2.2. La technique de reconstruction algébrique ART

L'algorithme ART (Algebric Reconstruction Technique) fait partie des méthodes itératives. Son principe revient à trouver l'image $f$ en connaissant sa matrice de projection $R$ et ses projections acquises $P$. La valeur d'une projection en un point vaut **[Bloch]** :

$$p_j = \sum_{i=1}^{n} R_{ji} f_i \qquad (2.7)$$

Le problème peut s'exprimer sous forme matricielle comme suit :

$$P = Rf \qquad (2.8)$$

Avec :

- $P$ représente le vecteur de mesures dont chaque composante représente une valeur de projection. Sa taille égale à $m$ définie comme le produit entre le nombre de projections par le nombre de points par projection.
- $f$ est le vecteur des valeurs recherchées en chaque pixel. Chaque composante correspond à la valeur d'atténuation en un pixel. Sa taille égale au nombre de pixels $n$.
- $R$ est la matrice de projection de taille $n \times m$. Les coefficients $R_{ji}$ correspondent à l'intersection du rayon $j$ avec le pixel $i$, avec :

$$R_{ji} = \begin{cases} 1 & \text{Si le ravon } j \text{ rencontre le pixel } i \\ 0 & \text{Sinon} \end{cases}$$

Une version plus avancée consiste à choisir des coefficients proportionnels à la surface d'intersection entre la surface élémentaire représentant le pixel et le rayon.

Le but principal est de corriger les coefficients $f_i$ de $f$ en utilisant seulement une seule projection par itération.

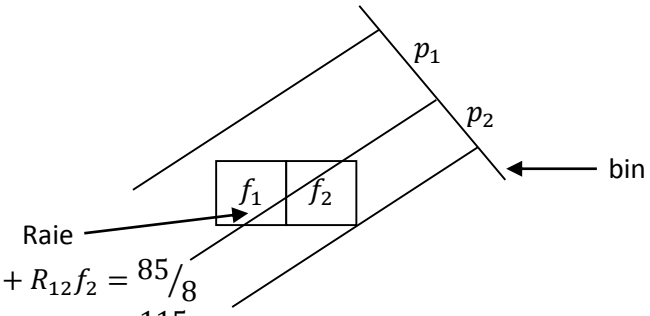
Mathématiquement, la correction des coefficients $f_i$ est basée sur la méthode de Kaczmarz **[Bourguet]** :

$$f_i^{(k)} = f_i^{(k-1)} + R_{ji} \frac{p_j - R_j f^{(k-1)}}{\|R_j\|^2} \quad (2.9)$$

Avec :

$f_i^{(k)}$ : itération corrigée, $f_i^{(k-1)}$ : itération précédente corrigée, $p_j$ : la vraie projection, $R_j f^{(k-1)}$ : la projection recalculée, $\|R_j\|^2$ : la norme de la ligne de la matrice *R* correspondant à la ligne *j* (nombre de pixels traversés par le rayon *j*).

Nous donnons un exemple simple de deux pixels et deux raies de projection pour mettre cette méthode en application.




$p_1 = R_{11} f_1 + R_{12} f_2 = {}^{85}/_{8}$

$p_2 = R_{21} f_1 + R_{22} f_2 = {}^{115}/_{8}$

$R = \begin{bmatrix} {}^{7}/_{8} & {}^{1}/_{8} \\ {}^{1}/_{8} & {}^{7}/_{8} \end{bmatrix}$ ($f_1 = 10, f_2 = 15$) Indication

déterminons $f_1$ et $f_2$.

$$p_1 = 7/8\, f_1 + 1/8\, f_2 = {}^{85}/_{8} \quad (1)$$

$$p_2 = 1/8\, f_1 + 7/8\, f_2 = {}^{115}/_{8} \quad (2)$$

Utilisation d'une seule équation de raie par itération, et mise à jour de chaque inconnu à partir de cette équation :

Itération 1 : Estimation de $f_1$ à partir de l'équation 1 seulement

Estimation de $f_2$ à partir de l'équation 1 seulement

Itération 2 : Estimation de $f_1$ à partir de l'équation 2 seulement

Estimation de $f_2$ à partir de l'équation 2 seulement

Itération 3 : Estimation de $f_1$ à partir de l'équation 1 seulement

Estimation de $f_2$ à partir de l'équation 1 seulement

etc…

Itération $k = 1$, estimation de $f_1$ et $f_2$ :

$$f_1^1 = 0 + 7/8 \frac{10.6}{\left\|{}^7/_8\right\|^2 + \left\|{}^1/_8\right\|^2} = 11.872$$

$$f_2^1 = 0 + 1/8 \frac{10.6}{\left\|{}^7/_8\right\|^2 + \left\|{}^1/_8\right\|^2} = 1.696$$

Itération $k = 2$, estimation de $f_1$ et $f_2$ :

$$f_1^2 = 11.872 + 1/8 \frac{14.375 - [1/8\,(11.872) + 7/8\,(1.696)]}{\left\|{}^7/_8\right\|^2 + \left\|{}^1/_8\right\|^2} = 13.6971$$

$$f_2^2 = 1.696 + 7/8 \frac{14.375 - [1/8\,(11.872) + 7/8\,(1.696)]}{\left\|{}^7/_8\right\|^2 + \left\|{}^1/_8\right\|^2} = 14.4719$$

A chaque itération, nous effectuons une mise à jour successive de tous les inconnus (ici $f_1$ et $f_2$) à partir de l'équation correspondant à une seule raie de projection. Tous les résultats sont listés dans le tableau suivant :

Tableau 2.1. Résultats des différentes itérations.

| Itération | $k = 1$ | $k = 2$ | $k = 3$ | $k = 4$ | $k = 5$ |
|---|---|---|---|---|---|
| $f_1$ | 11.872 | 13.6971 | 10.10 | 10.30 | 10.0 |
| $f_2$ | 1.696 | 14.4719 | 13.90 | 14.90 | 15.0 |

Nous arrivons alors à la correction des coefficients $f_1$et $f_2$ après cinq itérations.

#### 2.3.2.3. La transformation Mojette

La transformée Mojette est une version discrète exacte de la transformée de Radon définie pour des angles de projections spécifiques. Cette transformée et son inverse ont été définies par J. Guédon en 1995 **[Guédon 1995]** lors des travaux réalisés par l'équipe IVC de laboratoire IRCCyN inspirés des travaux de Katz **[Katz 1977]**.

Le mot « Mojette » est connu dans la région de Poitiers. Il décrit dans la France ancienne un type d'haricot blanc. Ce haricot blanc était un outil de base pour les enfants pour apprendre les opérations d'additions et de soustractions. Il était aussi un outil pour calculer le nombre de victoires pour les jeux de cartes en Espagne dans le moyen âge. Le choix de ce mot

pour cette transformation a pour but de rappeler que nous pouvons réaliser des calculs simplement avec des additions et de soustractions lors de l'implémentation de cette transformée.

### a- La transformée Mojette directe

La transformée Mojette permet de représenter une image ou une région $f(k,l)$ par une série de projections discrètes finies. La projection est choisie pour des angles discrets définis par un ensemble de vecteurs $(p,q)$ comme $\theta = \tan^{-1}\left({}^{q}/_{p}\right)$. La transformée Mojette 2-D pour une image de taille $P \times Q$, qui est linéaire est définie pour chaque angle de projection par :

$$M\{f(k,l)\} = Proj(p,q,b) = \sum_{k=0}^{P-1}\sum_{l=0}^{Q-1} f(k,l)\,\Delta(b+kq-lp) \qquad (2.10)$$

Avec : $\Delta(x) = \begin{cases} 1\ si\ x = 0 \\ 0\ si\ x \neq 0 \end{cases}$ correspond à la fonction de Kronecker discrète.

$(k,l)$ correspond à la position du pixel dans l'image.

La valeur d'un bin égale à la somme des pixels de la ligne $b = lp - kq$ comme le montre l'exemple de la figure 2.11.

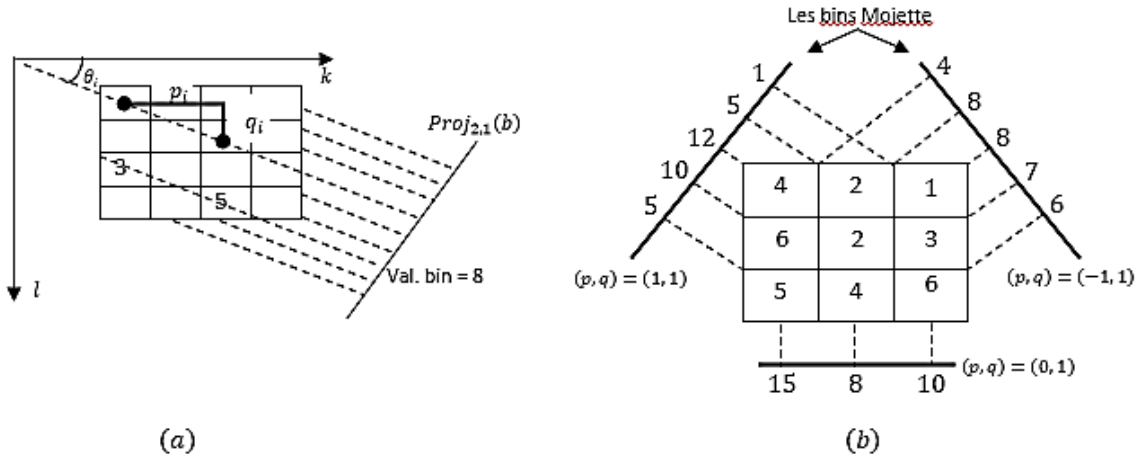



Figure 2.11. Principe de la transformée Mojette.

$(a)$ La représentation de $(p_i, q_i)$ avec l'angle $\theta_i$ correspondant. $i \in [1 \ldots . . I]$ est l'ensemble des projections pour différents angles indexés par $i$.
$(b)$ Une transformation Mojette d'une image $3 \times 3$ en utilisant les vecteurs de direction $\{(1,0),(1,1),(-1,1)\}$.

Notons que l'espace entre les lignes adjacentes de sommation varie avec l'angle de projection. La figure 2.12 donne un exemple de calcul des lignes $b = lp - kq$ pour la direction $(p, q) = (1, 1)$ **[Moreno et al 2011].**

La différence majeure avec la transformée de Radon est l'échantillonnage des projections **[Kingston and Autrusseau 2008]**. Le pas d'échantillonnage au $i$-ème projection est $h_i = 1 \Big/ \sqrt{p_i^2 + q_i^2}$. La conséquence de cet échantillonnage est que le nombre des bins sur une projection dépond de la valeur du vecteur $(p, q)$ et de la forme de l'image projetée.

| Pixel $(k, l)$ | $b = lp - kq$ |
|---|---|
| $(1,1)$ | $b = 1 - 1 = 0$ |
| $(2,1)$ | $b = 1 - 2 = -1$ |
| $(3,1)$ | $b = 1 - 3 = -2$ |
| $(1,2)$ | $b = 2 - 1 = 1$ |
| $(2,2)$ | $b = 2 - 2 = 0$ |
| $(3,2)$ | $b = 2 - 3 = -1$ |
| $(1,3)$ | $b = 3 - 1 = 2$ |
| $(2,3)$ | $b = 3 - 2 = 1$ |
| $(3,3)$ | $b = 3 - 3 = 0$ |

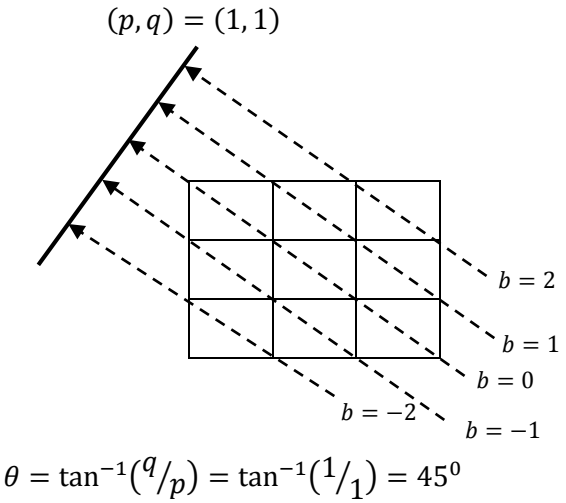



Figure 2.12. Calcul des lignes $b = lp - kq$ pour la direction $(p, q) = (1, 1)$.

Pour une image de taille $P \times Q$, le nombre de bins pour une projection sur une direction $(p, q)$ donnée est assuré par l'équation suivante :

$$B_{(p,q)} = (P - 1)|q| + (Q - 1)|p| + 1 \qquad (2.11)$$

Pour le cas de la figure 2.11, sur $(p, q) = (0, 1)$,$B_{(0,1)} = (3 - 1)|1| + (3 - 1)|0| + 1 = 3$.

Cette transformée linéaire génère une redondance calculée usuellement par :

$$red = \frac{\sum_i B_{(p,q)}}{\sum Pixels} - 1 \qquad (2.12)$$

Pour l'exemple précédent, la redondance est : $red = \frac{13}{9} - 1 = 44.44\%$.

**b- La transformée Mojette inverse**

La reconstruction de l'image est basée sur le critère de Katz **[Katz 1977]** qui est le critère d'existence d'une rétroprojection unique à partir d'un ensemble de projection $I$. Il montre que si l'équation (2.13) est satisfaite, une image de taille $P \times Q$ peut être reconstruite d'une manière unique :

$$P \leq \sum_{i=1}^{I} |p_i| \quad ou \quad Q \leq \sum_{i=1}^{I} |q_i| \tag{2.13}$$

La transformée Mojette est obtenue à partir des additions et des soustractions **[Normand and Guédon 1996]**. La reconstruction est basée sur la rétroprojection des bins en correspondance univoque avec un pixel. La valeur du bin rétroprojetée dans le pixel est soustraite de toutes les projections. Cette technique permet de faire apparaitre des nouvelles correspondances univoques. Ainsi, Le nombre de pixels contribuant aux bins correspondant est décrémenté de « 1 ».

La rétroprojection est très simple à comprendre à partir d'un exemple, comme le montre la figure 2.13, mais elle est très complexe à l'implémenter algorithmiquement.

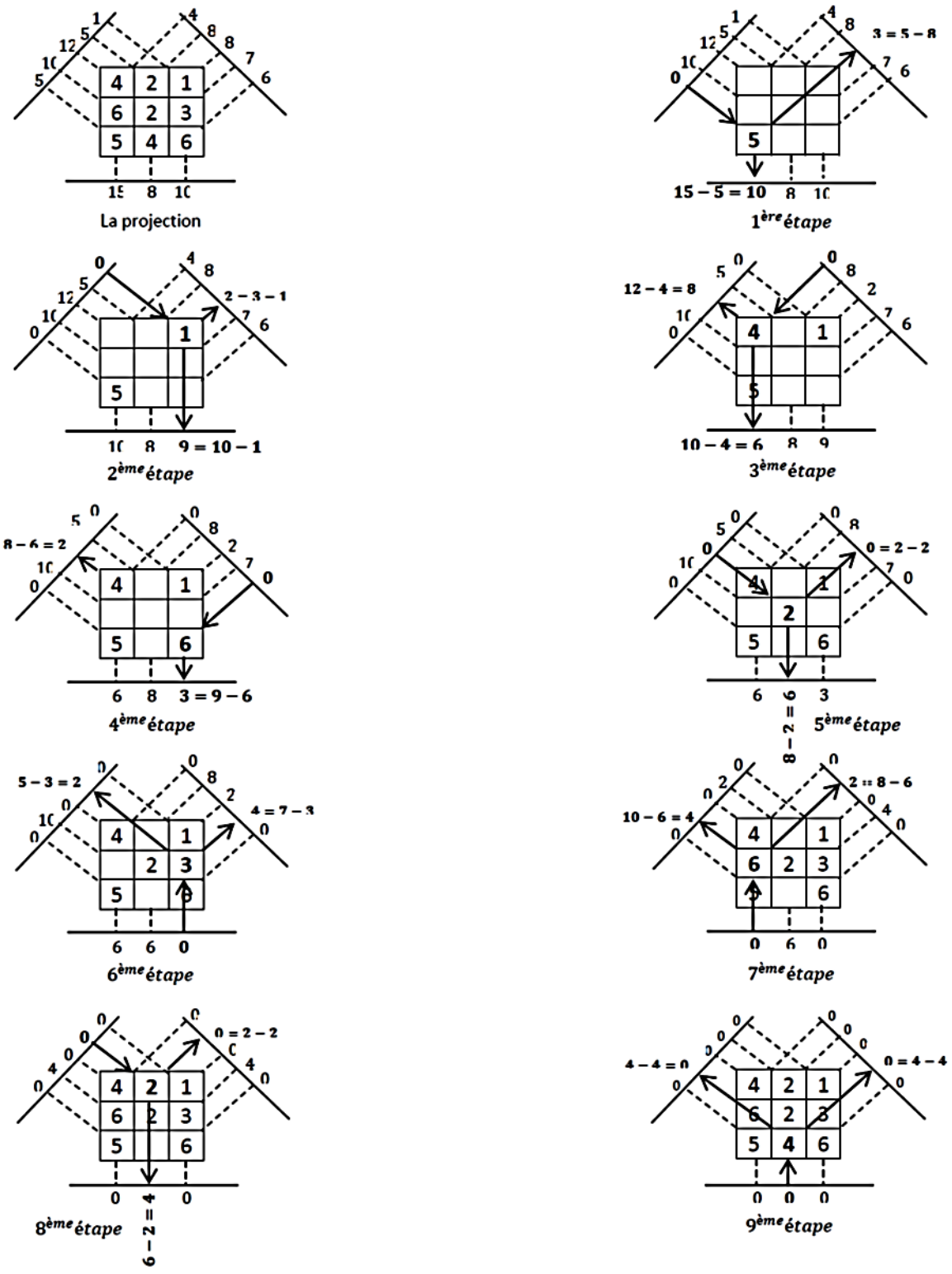



Figure 2.13. Les étapes possibles de la transformée Mojette inverse.

## 2.4. Discussion

Ce chapitre nous a permis de faire une étude détaillée de la transformée de Radon dans son aspect mathématique et son application dans le domaine de l'imagerie. Cette transformée constitue la base de la tomographie pour déterminer la structure interne d'un objet à partir de ses projections. Elle permet de convertir une image en une série de projections 1-D pour tout angle $\theta \in [0,\ 2\pi]$. Ces projections donnent l'avantage de travailler sur un nombre de données réduites par rapport à celles de départ ce qui a donné l'importance à cette transformée et attirer les chercheurs pour développer des méthodes basées sur cette dernière. La transformée de Radon comporte beaucoup de propriétés, mais la plus importante est l'inversion qui nous permet la reconstruction de l'image de départ projetée à partir de ses projections d'une manière non destructive.

Notre étude est partagée en deux parties selon le domaine de l'application : la première partie est consacrée à l'application de la transformée de Radon dans le domaine continu. Nous avons remarqué lors de cette partie que la reconstruction d'un objet à partir de ses projections dans sa forme simple donne une image floue de cet objet, cela nous a conduit à définir deux méthodes dans le but de reconstruire l'image d'une manière claire. Ces méthodes sont : la reconstruction Fourier directe basée sur le théorème de coupe centrale, qui est utile pour une projection non bruitée et la reconstruction par la rétroprojection des projections filtrées FBP (Filter-BackProject). La deuxième partie a été consacrée à l'application de la transformée de Radon dans le domaine discret. De même, nous avons défini trois méthodes qui sont : la transformée de Radon discrète basée sur l'interpolation, la technique de reconstruction algébrique (ART) qui est un algorithme itératif et la transformation Mojette qui est une version discrète exacte de la transformée de Radon.

A partir de la démarche observée au cours de ces deux premiers chapitres, nous consacrons le chapitre suivant à la description détaillée de notre méthode de compression d'images.

# Chapitre III

## Compression d'images basée sur la transformée de Radon et le codage prédictif linéaire LPC

# Chapitre 3

# Compression d'image basée sur la transformée de Radon et le codage prédictif linéaire LPC.

## 3.1. Préambule

La transformée de Radon permet de convertir une image en une série de projections 1-D pour tout angle $\theta \in [0,\ 2\pi]$. Cette transformée nous permet de définir le premier niveau de confidentialité de l'image lors du codage **[Devaki and Raghavendra 2012]**. Nous avons constaté que le codage LPC permet de réduire énormément la redondance interpixel et de coder seulement la nouvelle information représentée dans chaque pixel. Ce troisième chapitre a pour objectif de tirer parti des deux chapitres précédents afin de proposer une méthode de compression d'images basée sur la transformée de Radon et le codage prédictif LPC.

## 3.2. Principe de la méthode

La figure 3.1 montre le principe de notre méthode de compression d'images. Au niveau de la source, la première étape consiste à appliquer la transformée de Radon à l'image d'entrée pour une plage d'angle $\theta \in [0,\ 2\pi]$ dans le but de récupérer une matrice contenant les projections 1-D de l'image. Un codage prédictif LPC est appliqué à la matrice des projections pour extraire seulement l'information utile présentant dans chaque projection des pixels corrélés pour la coder ensuite par un codage arithmétique. L'étape suivante est l'application du codage RLC. Cette étape permet d'assurer un taux de compression fixe et élevé.

Au niveau de la réception, nous effectuons dans une première étape un décodage RLC pour récupérer les séquences des symboles qui seront ensuite décodés par un décodeur arithmétique pour récupérer les prédictions résiduelles. Un décodage LPC est effectué aux prédictions résiduelles pour récupérer la matrice des projections. Ce décodage sera basé sur les coefficients de prédictions linéaires du même prédicteur initial. La dernière étape consiste à appliquer une transformée de Radon inverse à la matrice des projections 1-D pour la même

plage $\theta \in [0,\ 2\pi]$ pour récupérer l'image de départ en utilisant la méthode d'interpolation avec filtrage.

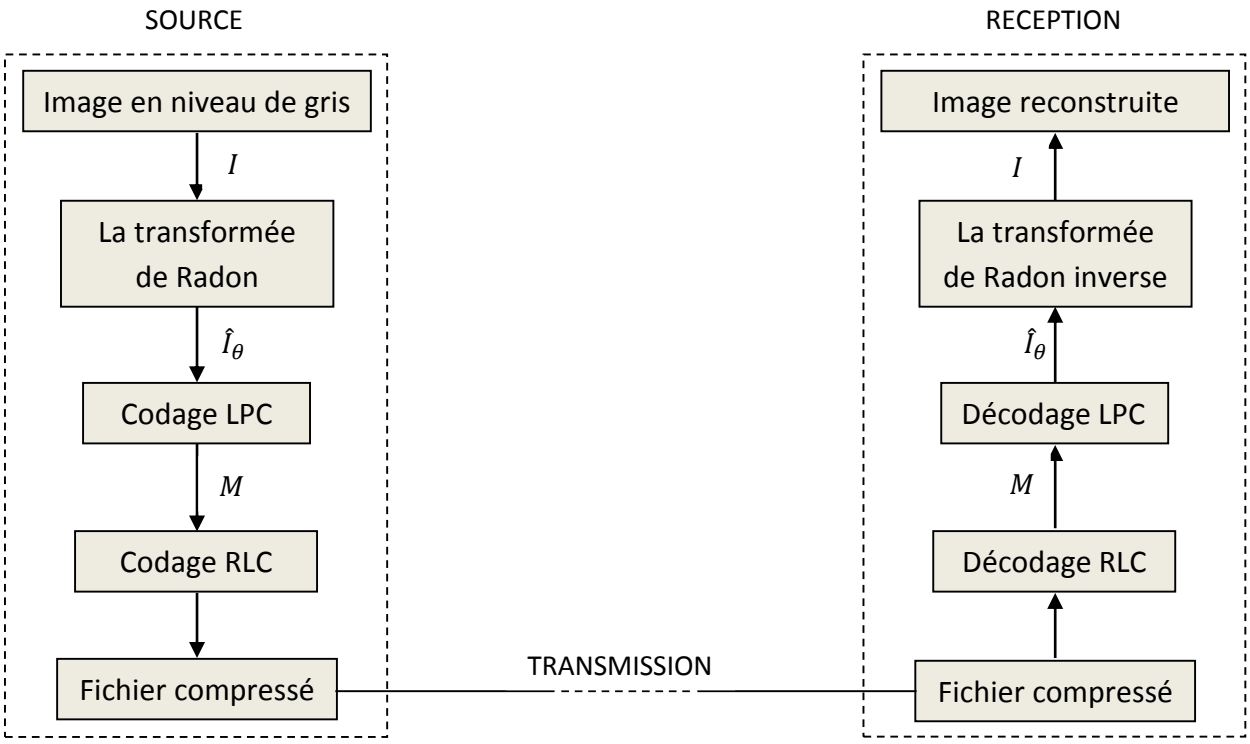



Figure 3.1. Schéma bloc de la méthode proposée.

## 3.3. Au niveau de la source

### 3.3.1. La transformée de Radon

Initialement, une transformée de Radon est appliquée pour une plage $\theta \in [0,\ 2\pi]$ à l'image d'entrée $I$ . Cette transformée est assurée par l'équation suivante :

$$\hat{I}_\theta(p) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} I(\vec{r})\, \delta(p - x\cos\theta - y\sin\theta) d\vec{r} \qquad (3.1)$$

Avec :

$I$ : Image d'entrée.

$\vec{r}$ : Vecteur de position.

$p = \vec{\zeta}\vec{r}$ : Un hyperplan, avec $\vec{\zeta}$ un vecteur unitaire.

$\delta$ : La fonction de Dirac

Nous récupérons par la suite, la matrice des projections 1-D $\hat{I}_\theta$ qui représente la somme des pixels pour chaque angle $\theta$.

### 3.3.2. Le codage prédictif LPC

Afin de décorréler les points Radon qui représentent les sommes des pixels pour coder seulement l'information utile, la matrice des projections $\hat{I}_\theta(p)$ sera soumise à un codage prédictif LPC. Le principe du codeur est représenté par la figure 3.2.

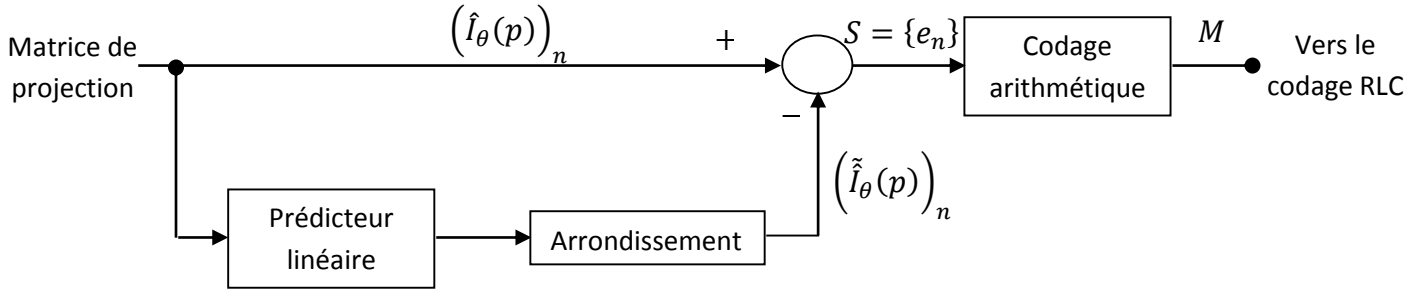



Figure 3.2. Le codage LPC.

L'extraction de la prédiction résiduelle représentant dans chaque point Radon est effectuée de la manière suivante :

Soit $\left(\hat{I}_\theta(p)\right)_n$ la valeur d'un point Radon donnée, $\left(\tilde{\hat{I}}_\theta(p)\right)_n$ représente la valeur prédite du même point Radon basée sur les valeurs des points Radon précédents. $e_n$ représente la prédiction résiduelle du point Radon actuel qui sera codée par la suite par un codage arithmétique.

Cette prédiction résiduelle est donnée par l'équation suivante :

$$e_n = \left(\hat{I}_\theta(p)\right)_n - \left(\tilde{\hat{I}}_\theta(p)\right)_n \quad (3.2)$$

Notre choix est fixé pour un prédicteur linéaire pour générer la valeur prédite pour chaque point Radon. L'équation générale du prédicteur est une combinaison linéaire des $m$ points Radon précédents :

$$\left(\tilde{\hat{I}}_\theta(p)\right)_n = round\left[\sum_{i=1}^{m} \alpha_i \left(\hat{I}_\theta(p)\right)_{n-1}\right] \quad (3.3)$$

Avec :

$m$ : Ordre du prédicteur.

$round$ : Arrondissement de la valeur prédite $\left(\tilde{\hat{I}}_\theta(p)\right)_n$.

$\alpha_i$ : Les coefficients de prédiction pour $i = 1, 2, \ldots\ldots, m$.

Nous donnons par la suite un exemple pour montrer le fonctionnement du codage LPC. Considérons une matrice $A$ de dimension $3 \times 3$ définie comme suit :

$$\hat{I}_\theta(p) = \begin{bmatrix} 10 & 2 & 3 \\ 8 & 4 & 5 \\ 9 & 6 & 1 \end{bmatrix}$$

Le but est donc de trouver les valeurs prédites et les prédictions résiduelles.

Pour faciliter les calculs, considérons $m = 1$. Donc nous sommes dans le cas d'un prédicteur linéaire 1-D et que $\alpha_i = \alpha = 1 \; \forall \; i$.

Pour une matrice de 2-D, l'équation (3.3) peut s'écrire dans ce cas-là comme suit :

$$\tilde{\hat{I}}_\theta(x, y) = round\left[\alpha \, \hat{I}_\theta(x, y - i)\right]$$

Nous avons d'après la matrice $\hat{I}_\theta(p)$ : $\hat{I}_\theta(1,1) = 10$, $\hat{I}_\theta(1,2) = 2$, et ainsi de suite.

Calculons les $e(x, y) = \hat{I}_\theta(x, y) - \tilde{\hat{I}}_\theta(x, y)$.

Nous avons au début $e(1,1) = 10$.

$\tilde{\hat{I}}_\theta(1,2) = round\left(\alpha * \hat{I}_\theta(1,2-1)\right) = round\left(1 * \hat{I}_\theta(1,1)\right) = round(1 * 10) = 10.$

$e(1,2) = \hat{I}_\theta(1,2) - \tilde{\hat{I}}_\theta(1,2) = 2 - 10 = -8.$

$\tilde{\hat{I}}_\theta(1,3) = round\left(\alpha * \hat{I}_\theta(1,3-1)\right) = round\left(1 * \hat{I}_\theta(1,2)\right) = round(1 * 2) = 2.$

$e(1,3) = \hat{I}_\theta(1,3) - \tilde{\hat{I}}_\theta(1,3) = 3 - 2 = 1.$

$e(2,1) = 8.$

$\tilde{\hat{I}}_\theta(2,2) = round\left(\alpha * \hat{I}_\theta(2,2-1)\right) = round\left(1 * \hat{I}_\theta(2,1)\right) = round(1 * 8) = 8.$

$e(2,2) = \hat{I}_\theta(2,2) - \tilde{\hat{I}}_\theta(2,2) = 4 - 8 = -4.$

$\tilde{\hat{I}}_\theta(2,3) = round\left(\alpha * \hat{I}_\theta(2,3-1)\right) = round\left(1 * \hat{I}_\theta(2,2)\right) = round(1 * 4) = 4.$

$e(2,3) = \hat{I}_\theta(2,3) - \tilde{\hat{I}}_\theta(2,3) = 5 - 4 = 1.$

$e(3,1) = 9.$

$\tilde{\hat{I}}_\theta(3,2) = round\left(\alpha * \hat{I}_\theta(3,2-1)\right) = round\left(1 * \hat{I}_\theta(3,1)\right) = round(1 * 9) = 9.$

$e(3,2) = \hat{I}_\theta(3,2) - \tilde{\hat{I}}_\theta(3,2) = 6 - 9 = -3.$

$\tilde{\hat{I}}_\theta(3,3) = round\left(\alpha * \hat{I}_\theta(3,3-1)\right) = round\left(1 * \hat{I}_\theta(3,2)\right) = round(1 * 6) = 6.$

$e(3,3) = \hat{I}_\theta(3,3) - \tilde{\hat{I}}_\theta(3,3) = 1 - 6 = -5.$

Nous récupérons à la fin la matrice des prédictions résiduelles :

$$e(x,y) = \begin{bmatrix} 10 & -8 & 1 \\ 8 & -4 & 1 \\ 9 & -3 & -5 \end{bmatrix}$$

Dans notre cas, nous utilisons des coefficients d'un filtre SNN (Symmetric Nearest Neighbor) au lieu des coefficients égaux à l'unité. Cette procédure a pour effet d'effectuer un filtrage pour les points Radon et d'engendrer une certaine perte d'information lors de la compression.

Le filtre SNN, ou le filtre de Harwood **[Harwood et 1987]** est considéré comme le meilleur filtre pour la conservation des propriétés des contours. Le masque du filtre SNN est représenté par la figure 3.3.

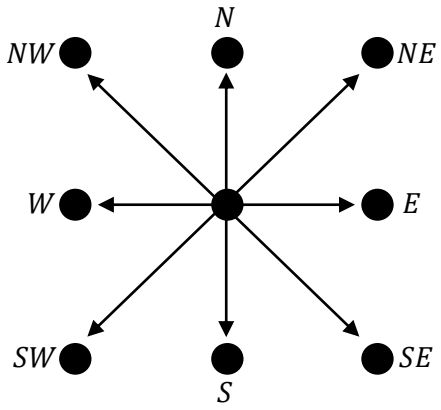



Figure 3.3. Le masque du filtre SNN.

Les voisinages du pixel central dans la fenêtre représentées par le masque sont considérés comme quatre paires des pixels symétriques ($N-S$,$W-E$, $NW-SW$, $NE-SW$). Nous choisissons dans chaque paire, le pixel qui a la valeur la plus proche de celle du pixel central. Ainsi, la moyenne de ces quatre pixels sera affectée au pixel central.

Nous donnons l'exemple suivant pour mieux expliquer le fonctionnement du filtre SNN.

Soit le masque suivant :

$$\begin{bmatrix} 100 & 99 & 113 \\ 215 & 230 & 150 \\ 90 & 66 & 200 \end{bmatrix}$$

Nous effectuons le filtrage de la manière suivante :

Considérons les quatre paires des points Radon opposants au point Radon central $230$.

$S_1 = \{99, 66\}$, $S_2 = \{215, 150\}$, $S_3 = \{100, 200\}$, $S_4 = \{113, 90\}$.

Nous choisissons la valeur du point Radon la plus proche en valeur absolue de celle du point Radon central, par exemple pour $S_1$:

$|99 - 230| = 131$ et $|66 - 230| = 164$. Donc nous prenons la valeur $99$. De même pour les autres paires. Nous trouvons à la fin l'ensemble $Min$ des quatre points Radon les plus proches du point Radon central qui est :

$$Min = \{99, 215, 200, 113\}$$

Nous calculons la moyenne de l'ensemble $Min$ pour l'affecter par la suite au point Radon central. Dans notre cas, cette valeur est :

$$Moy(Min) = Moy\{99{,}215{,}200{,}113\} = 156.7500$$

L'étape suivante du codage LPC consiste à coder le vecteur séquence qui contient toutes les prédictions résiduelles par le codage arithmétique. Le codage arithmétique a été abordé en détails dans le premier chapitre avec exemple.

Les étapes du codage arithmétique sont décrites comme suit :

1. Récupérer le vecteur séquence $S = (S_1, S_2, \dots\dots, S_n)$ contenant les prédictions résiduelles.
2. Créer la table des prédictions pour calculer pour chaque symbole sa fréquence d'apparition dans le vecteur séquence.
3. Compter la Fonction de Distribution Cumulative (CFD) des probabilités du vecteur séquence $S$. Avec $CFD(m_k) = \sum_{k=1}^{i} P(S_k)$. Et nous avons $\sum_{k=1}^{n} P(S_k) = 1$.
4. Diviser l'intervalle unitaire initial $[0, 1]$ en proportion des probabilités pour trouver les points limites de cet intervalle qui correspondent aux valeurs de la CFD.
5. Calculer d'une manière séquentielle la borne supérieure et inférieure de l'intervalle où le symbole courant est lié.
6. Choisir à la fin l'identificateur qui est le milieu de l'intervalle final.
7. L'identificateur est sous forme $0.v$, donc nous le transformons en code binaire. Ainsi, nous calculons le nombre minimal des digits binaires nécessaire $\overline{L}$ pour représenter l'identificateur. A la fin, le mot code sera tout simplement la représentation binaire de $v$.

Nous reprenons l'exemple du premier chapitre. Nous avons l'identificateur $0.438875 = 0.01110000010110\dots$, comme $\overline{L} = 12$ donc $0.438875 = 0.01110000010$. Le code de la séquence $a_1a_3a_2a_4a_1$ sera $01110000010$.

8. Nous effectuons en final un découpage du code binaire pour créer des symboles décimaux pour les mieux compressés par la suite par le codage RLC. Nous récupérons à la fin une séquence des mots-code $M$.

### 3.3.3. Le codage RLC

Le codage RLC est l'un des codages utilisés pour réaliser une compression de manière efficace lorsque nous disposons d'une longue séquence de symboles identiques. Le résultat consiste à générer pour chaque séquence de même mots-code une paire contenant le mot-code et la longueur de cette même séquence. Cette étape nous permet de réaliser un taux de compression élevé et fixe quel que soit l'image introduite car nous disposons après découpage de l'identificateur binaire presque les mêmes symboles décimaux après transformation décimale. Les étapes du codage RLC sont résumées de la manière suivante :

1. Récupérer les mots-code décimaux issus du codage arithmétique $M = \{M_1, M_2, \dots\dots, M_n\}$. chaque mot-code est une transformation décimale d'un découpage en 8 $bits$.
2. Trouver les séquences de même mots-code de longueur fixe $b$. La longueur maximale de la séquence est $L$. Le nombre d'apparition est $C = L/b$.
3. Générer pour chaque séquence une paire contenant le mot-code et la longueur de la séquence.
4. Formation du fichier compressé pour le transmettre au récepteur.

Nous donnons un exemple du codage RLC pour montrer son fonctionnement.

Soit une séquence de bit $110111100000101111101111$. Nous distinguons les différentes séquences suivantes :

$$\underbrace{11}_{1}\,\underbrace{0}_{0}\,\underbrace{1111}_{1}\,\underbrace{000000}_{0}\,\underbrace{1}_{1}\,\underbrace{0}_{0}\,\underbrace{11111}_{1}\,\underbrace{0}_{0}\,\underbrace{1111}_{1}$$

Donc les paires des mots code ainsi les longueurs des séquences seront représentées comme suit :

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| La séquence : | 110111100000101111101111 | | | | | | | | |
| Mot-code : | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |
| Longueur de la séquence : | 2 | 1 | 4 | 6 | 1 | 1 | 5 | 1 | 4 |

## 3.4. Au niveau du récepteur

Au niveau du récepteur, après la récupération du fichier compressé, nous effectuons les mêmes étapes lors du codage dans l'ordre inverse. Nous donnons les étapes de ce décodage.

### 3.4.1. Décodage RLC

Nous effectuons au fichier compressé un décodage RLC en se basant sur le nombre d'apparition des mots-code. Ce décodage nous permet de récupérer la séquence des mots-code issus du codage arithmétique. Les étapes du décodage RLC sont :

1. Pour chaque mot-code donné de longueur $b$ fixe, générer la séquence et sa longueur maximale en se basant sur le nombre d'apparition de ce même mot-code.
2. Récupérer le vecteur des mots-code $M$.

### 3.4.2. Décodage prédictif LPC

Le processus du décodage LPC commence premièrement par un décodage arithmétique pour récupérer le vecteur séquence $S$ qui contient les prédictions résiduelles.

Le schéma du décodage LPC est donné par la figure 3.4.

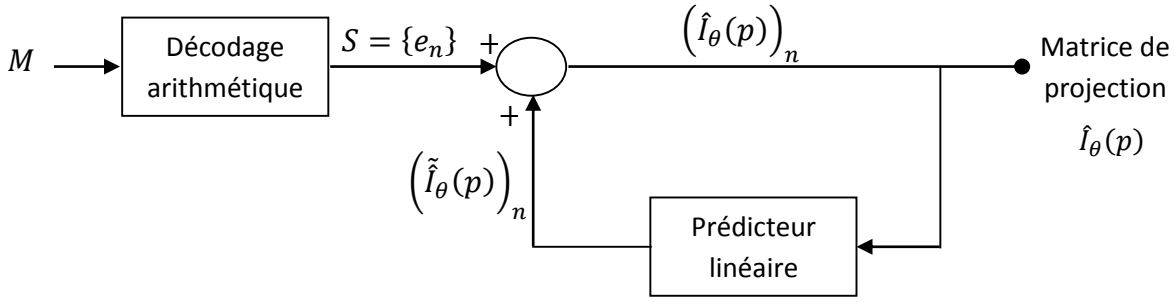



Figure 3.4. Le décodage LPC.

Les étapes du décodage arithmétique pour déchiffrer l'identificateur sont décrites comme suit :

1. Transformer les symboles décimaux en digits binaires pour récupérer le $v$.
2. Transformer l'identificateur sous forme de $0.v$ en décimal.

3. Initialiser $I_I(0) = 0\ et\ I_S(0) = 1$.
4. Pour chaque $k$, Calculer $T_1 = \big(Identificateur - I_I(k-1)\big)/\big(I_S(k-1) - I_I(k-1)\big)$.
5. Déterminer $m_k$ tel que $CFD\ (m_k - 1) \leq T_1 \prec CFD(m_k)$.
6. Actualiser les bornes $I_I(k)$ et $I_S(k)$.
7. Continuer jusqu'à ce que la totalité de la séquence soit décodée. Nous récupérons par la suite le vecteur séquence $S$.

Nous avons donné lors du premier chapitre un exemple du codage. Pour montrer comment le processus du décodage fonctionne. Nous reprenons le même exemple et nous essayons de déchiffrer l'identificateur pour retrouver la séquence de départ.

Nous avons les valeurs de la $CFD$ qui sont définies comme suit : $CFD(k) = 0$ si $k \leq 0$, $CFD(1) = 0.5$, $CFD(2) = 0.8$, $CFD(3) = 0.9$, $CFD(4) = 1$, $CFD(k) = 1$ si $k \geq 4$. Et nous avons la valeur de l'identificateur qui est $0.438875$.

1. Pour $k = 1$

Nous avons au départ $T_1 = (0.438875 - 0)/(1 - 0) = 0.438875$. Nous déterminons par la suite la valeur de $m_1$ tel que $CFD\ (m_1 - 1) \leq T_1 \prec CFD(m_1)$. D'une autre manière $T_1 \in [CFD\ (m_1 - 1), CFD\ (m_1))$.

Si par exemple nous prenons $m_1$=1, alors $T_1 \in \big[CFD\ (0), CFD\ (1)\big) \leftrightarrow T_1 \in [0, 0.5)$. Et si nous prenons $m_1$=2, nous aurons $T_1 \in \big[CFD\ (1), CFD\ (2)\big) \leftrightarrow T_1 \in [0.5, 0.8)$ ce qui est impossible. Donc nous avons $m_1$=1.

Nous faisons l'actualisation des bornes

$$I_I(k) = I_I(k-1) + \big(I_S(k-1) - I_I(k-1)\big) \times CFD\ (m_k - 1)$$

Et

$$I_S(k) = I_I(k-1) + \big(I_S(k-1) - I_I(k-1)\big) \times CFD\ (m_k)$$

Donc :

$$I_I(1) = I_I(0) + \big(I_S(0) - I_I(0)\big) \times CFD\ (0) = 0$$

$$I_S(1) = I_I(0) + \big(I_S(0) - I_I(0)\big) \times CFD\ (1) = 0.5$$

2. Pour $k = 2$

$T_2 = (0.438875 - 0)/(0.5 - 0) = 0.87775$

$m_2$ tel que $CFD\ (m_2 - 1) \leq T_2 \prec CFD(m_2) \leftrightarrow CFD\ (m_2 - 1) \leq 0.87775 \prec CFD(m_2)$

Si nous prenons $m_2 = 3$ , nous remarquons bien que $CFD\ (2) \leq 0.87775 \prec CFD(3) \leftrightarrow T_2 \in [0.8, 0.9)$

Actualisation des bornes :

$$I_I(2) = I_I(1) + \big(I_S(1) - I_I(1)\big) \times CFD\ (2) = 0.4$$

$I_S(2) = I_I(1) + \big(I_S(1) - I_I(1)\big) \times CFD\ (3) = 0.45$

3. Pour $k = 3$

$T_3 = (0.438875 - 0.4)/(0.45 - 0.4) = 0.7775$

$m_3$ tel que $CFD\ (m_3 - 1) \leq T_3 \prec CFD(m_3) \leftrightarrow CFD\ (m_3 - 1) \leq 0.7775 \prec CFD(m_3)$

Si nous prenons $m_3 = 2$, nous remarquons que $CFD\ (1) \leq 0.7775 \prec CFD(2) \leftrightarrow T_3 \in [0.5, 0.8)$

Actualisation des bornes :

$$I_I(3) = I_I(2) + \big(I_S(2) - I_I(2)\big) \times CFD\ (1) = 0.425$$

$I_S(3) = I_I(2) + \big(I_S(2) - I_I(2)\big) \times CFD\ (2) = 0.44$

4. Pour $k = 4$

$T_4 = (0.438875 - 0.425)/(0.44 - 0.425) = 0.925$

$m_4$ tel que $CFD\ (m_4 - 1) \leq T_4 \prec CFD(m_4) \leftrightarrow CFD\ (m_4 - 1) \leq 0.925 \prec CFD(m_4)$

Si nous prenons $m_4 = 4$ , nous remarquons que $CFD\ (3) \leq 0.925 \prec CFD(4) \leftrightarrow T_4 \in [0.9, 1)$

Actualisation des bornes :

$$I_I(4) = I_I(3) + \big(I_S(3) - I_I(3)\big) \times CFD\ (3) = 0.4385$$

$I_S(4) = I_I(3) + \big(I_S(3) - I_I(3)\big) \times CFD\ (4) = 0.44$

5. Pour $k = 5$

$T_5 = (0.438875 - 0.4385)/(0.44 - 0.4385) = 0.25$

$m_5$ tel que $CFD\ (m_5 - 1) \leq T_5 \prec CFD(m_5) \leftrightarrow CFD\ (m_5 - 1) \leq 0.25 \prec CFD(m_5)$

Si nous prenons $m_5 = 1$, nous remarquons que $CFD\ (0) \leq 0.925 \prec CFD(1) \leftrightarrow T_4 \in [0, 0.5)$

Actualisation des bornes :

$$I_I(5) = I_I(4) + \big(I_S(4) - I_I(4)\big) \times CFD\ (0) = 0.4385$$

$I_S(5) = I_I(4) + \big(I_S(4) - I_I(4)\big) \times CFD\ (1) = 0.43925$

Nous trouvons à la fin l'intervalle qui contient l'identificateur $0.438875 \in [0.4385, 0.43925)$. Ainsi, nous arrivons à décoder toute la séquence qui est $\{m_1 = 1, m_2 = 3, m_3 = 2, m_4 = 4, m_5 = 1\}$ et qui correspond à notre séquence de départ $a_1 a_3 a_2 a_4 a_1$. Le processus est arrêté car nous avons au préalable la longueur de la séquence qui est 5.

La deuxième phase du décodage LPC consiste à récupérer la matrice des projections $\hat{I}_\theta(p)$. Cette procédure est assuré par le même prédicteur linéaire pour reproduire les mêmes prédictions résiduelles $e_n$ afin d'assurer le décodage. La reconstruction des points Radon est donnée par l'équation suivante :

$$\left(\hat{I}_\theta(p)\right)_n = e_n + \left(\tilde{\hat{I}}_\theta(p)\right)_n \qquad (3.4)$$

Les étapes de récupération de la matrice des projections sont :

1. Récupérer le vecteur séquence $S$ qui contient les predictions residuelles $e_n$.
2. Calculer chaque point Radon par l'ajout de sa valeur prédite basée sur les points Radon précédents avec sa prédiction résiduelle en utilisant l'équation (3.4).
3. L'ensemble des points Radon forment la matrice des projections 1-D $\hat{I}_\theta(p)$.

Reprenons l'exemple du codage, nous savons que $\tilde{\hat{I}}_\theta(x, y) = round\big[\alpha\ \hat{I}_\theta(x, y - i)\big]$. Et nous avons la matrice des prédictions résiduelles :

$$e(x,y) = \begin{bmatrix} 10 & -8 & 1 \\ 8 & -4 & 1 \\ 9 & -3 & -5 \end{bmatrix}$$

L'équation (3.4) dans ce cas peut s'écrire comme : $\hat{I}_\theta(x,y) = e(x,y) + \tilde{\hat{I}}_\theta(x,y)$.

Nous aurons donc : $\hat{I}_\theta(1,1) = e(1,1) = 10$, $\hat{I}_\theta(1,2) = e(1,2) + \tilde{\hat{I}}_\theta(1,2) = -8 + 10 = 2$. $\hat{I}_\theta(1,3) = e(1,3) + \tilde{\hat{I}}_\theta(1,3) = 1 + 2 = 3$. De la même manière, nous calculons les autres points :

$\hat{I}_\theta(2,1) = e(2,1) = 8$, $\hat{I}_\theta(2,2) = e(2,2) + \tilde{\hat{I}}_\theta(2,2) = 4$, $\hat{I}_\theta(2,3) = e(2,3) + \tilde{\hat{I}}_\theta(2,3) = 5$.

$\hat{I}_\theta(3,1) = e(3,1) = 9$, $\hat{I}_\theta(3,2) = e(3,2) + \tilde{\hat{I}}_\theta(3,2) = 6$, $\hat{I}_\theta(3,3) = e(3,3) + \tilde{\hat{I}}_\theta(3,3) = 1$.

Nous retrouvons à la fin la même matrice des points Radon de départ :

$$\hat{I}_\theta(p) = \begin{bmatrix} 10 & 2 & 3 \\ 8 & 4 & 5 \\ 9 & 6 & 1 \end{bmatrix}$$

### 3.4.3. La transformée de Radon inverse

La dernière étape consiste à appliquer la transformée de Radon inverse à la matrice des projections 1-D pour la même plage $\theta \in [0, 2\pi]$ pour reconstruire l'image originale. Nous avons vu lors du deuxième chapitre que le maillage de l'image reconstruite est différent du maillage natif. Donc l'application d'une interpolation est nécessaire pour retrouver des valeurs exactes des pixels reconstruits.

La méthode d'interpolation que nous utilisons dans notre cas est la méthode spline cubique qui donne un meilleur résultat par rapport aux autres méthodes comme l'interpolation du voisin le plus proche (Nearest Neighbour interpolation) et l'interpolation linéaire.

Les étapes de la transformée de Radon inverse sont indiquées dans le schéma bloc suivant :

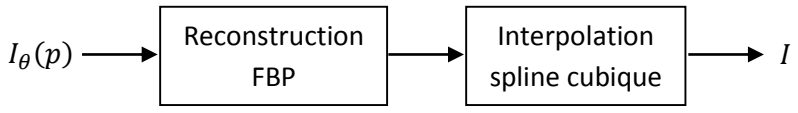



Figure 3.5. Schéma bloc de la méthode filtrage-interpolation.

L'interpolation spline nous permet de déterminer après la rétroprojection les valeurs des pixels de l'image situés entre leurs échantillons. Ce processus nous permet de reconstruire le maillage natif de l'image de départ et de récupérer des valeurs exactes pour les pixels. Cette méthode nous permet de prendre en considération 16 pixels de voisinage et d'estimer les valeurs les plus proches par une spline cubique comme le montre l'exemple de la figure 3.6.

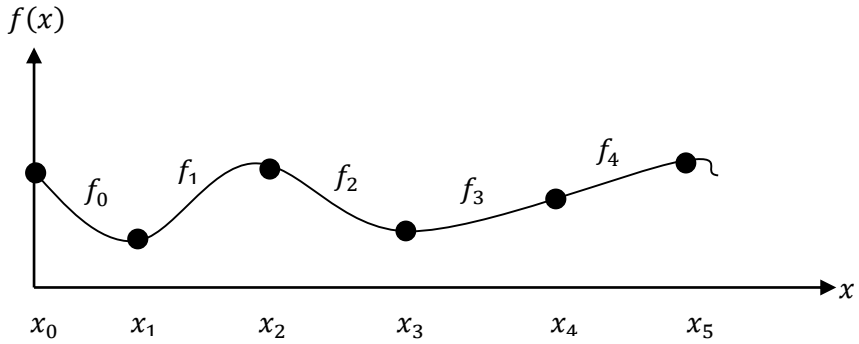



Figure 3.6. Une spline de cinq morceaux polynomiaux cubiques.

Une spline cubique est un polynôme de troisième degré continu par morceaux. Elle permet de réaliser une interpolation polynomiale par morceaux. Les spline donc passent par les pixels situés entre leurs échantillons. Ces pixels sont connus aussi sous le nom des points de control **[Wolberg 1994]**.

Une spline d'interpolation est un polynôme de troisième degré donné par l'équation suivante :

$$f(x) = y_k = ax^3 + bx^2 + cx + d \qquad (3.5)$$

Les coefficients du polynôme sont donnés comme suit :

$d = y_k$

$c = y'_k$

$b = 3\Delta y_k - 2y'_k - y'_{k+1}$

$a = 2\Delta y_k + y'_k + y'_{k+1}$

Et $\Delta y_k = y_k - y_{k+1}$

## 3.5. Discussion

Ce troisième chapitre nous a permis de détailler notre méthode de compression d'images qui tire profit de la transformée de Radon et du codage prédictif LPC. Nous avons en particulier :

- Réalisé la transformée de Radon de l'image d'entrée au niveau de la source.
- Réalisé le codage prédictif LPC pour la matrice des projections 1-D. Cette étape est déroulée en deux phases : la phase d'extraction des prédictions résiduelles et la phase du codage arithmétique.
- Réalisé le codage RLC pour créer un fichier compressé qui sera transmis au récepteur.
- Décodé avec succès le fichier compressé par le décodage RLC pour reconstruire les prédictions résiduelles.
- Réalisé le décodage prédictif LPC en deux étapes : la phase de décodage arithmétique et la phase de récupération de la matrice des projections 1-D.
- Restauré avec succès l'image de départ en appliquant la transformée de Radon inverse avec la méthode filtrage-interpolation.

Nous consacrerons le chapitre suivant pour réaliser une comparaison entre les performances de notre méthode avec ceux obtenus par d'autres méthodes de compression d'images.

# Chapitre IV

## Résultats et discussions

# Chapitre 4

# Résultats et discussions

## 4.1. Préambule

Ce chapitre a pour but de présenter nos résultats et de comparer notre méthode de compression d'images avec d'autres méthodes de compressions abordées dans ce mémoire et de vérifier le processus d'intégration **[Murphy 1986]**.

Notre étude comparative englobe les méthodes suivantes :

- RAD-LPC : notre méthode de compression d'images, basée sur la transformée de Radon et le codage prédictif LPC.
- RAD-DCT : méthode proposée par S. A. Pradeep et R.Manavalan **[Pradeep and Manavalan 2013]**. Cette méthode est basée sur la transformée de Radon et la transformée DCT. Nous appliquons dans notre cas cette méthode pour un niveau de quantification égal à 15.
- DCT : méthode de compression basée seulement sur la DCT sans passage par la transformée de Radon pour un niveau de quantification égal à 15.
- Bi-orthogonale DWT : méthode de compression d'images basée sur la transformée en ondelettes discrète **[Muhit et al 2005].**

Ces méthodes sont appliquées sur deux types d'images : tests et MSG.

## 4.2. Critères de qualité

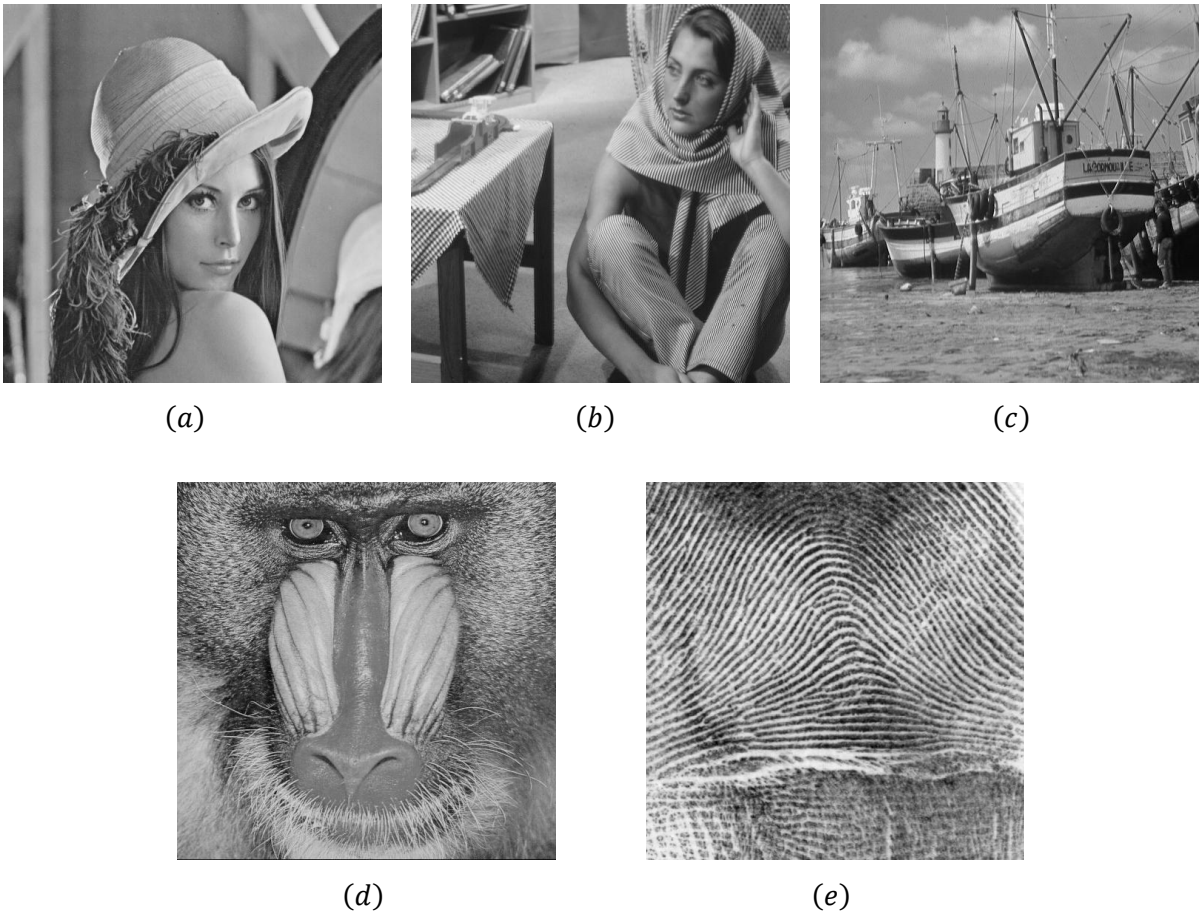
Nous avons opté comme critères de qualité pour évaluer notre méthode et de la positionner par rapport aux autres :

- Le taux de compression $T_c(\%)$.
- Le rapport signal sur bruit crête $PSNR(db)$.
- L'erreur absolue moyenne $MAE$.
- La racine carrée de l'erreur quadratique moyenne $RMSE$.

## 4.3. Présentation de la base de données

### 4.3.1. Images tests

Notre choix est porté sur plusieurs images tests très utilisées dans la compression d'images de résolution $512 \times 512$ avec 8 bits par pixel. Ces images sont de textures et de complexités différentes pour bien analyser la puissance de notre méthode. Ces images, comme le représente la figure 4.1, sont : Lena, Barbara, Boat, Mandrill et Fingerprint.



$(a)$ $(b)$ $(c)$

$(d)$ $(e)$

Figure 4.1. Les différentes images tests de la base de données.

(a) Lena; (b) Barbara; (c) Boat; (d) Mandrill; (e) Fingerprint.

### 4.3.2. Images du satellite MSG

Le satellite MSG (Meteosat Second Generation) est un satellite géostationnaire qui tourne à 100 tours par minute autour d'un axe parallèle à l'axe Nord-Sud de la Terre **[Eduscol]**. Il est équipé d'un capteur radiométrique SEVIRI (Spinning Enhanced Visible &

InfraRed Imager) qui permet l'acquisition de douze fichiers images qui correspondent aux douze canaux du visible, de l'infrarouge moyen et de l'infrarouge thermique toutes les quinze minutes. La mission principale du satellite MSG consiste à améliorer les prévisions météorologiques quotidiennes et anticiper les phénomènes violents. Mais il permet aussi de suivre quelques activités comme : le suivi des espaces urbanisés, l'évolution du couvert végétal, la détection des incendies de forêts…

Les principales caractéristiques du capteur SEVIRI sont données par le tableau suivant **[Education MétéoFrance] :**

Tableau 4.1. Caractéristiques et fonctions des canaux du capteur SERIVI.

| **Numéro du canal** | **Nom du canal** | **Bandes spectrales longueurs d'onde en µm** | **Fonctions des différents canaux** |
|---|---|---|---|
| C1 | VIS 0.6 | 0.56 à 0.71 | Détection et suivi des nuages. Surveillance de la surface des terres et des aérosols. Leur combinaison permet d'obtenir des indices de végétation. |
| C2 | VIS 0.8 | 0.74 à 0.88 | |
| C3 | NIR 1.6 | 1.5 à 1.78 | Distinction entre la neige et les nuages, entre les nuages de glace et les nuages d'eau. Informations sur les aérosols. |
| C4 | IR 3.9 | 3.48 à 4.36 | Détection nocturne et propriétés des nuages bas et du brouillard. Mesures nocturnes des températures du sol et de la mer. Détection des feux de forêt. |
| C5 | WV 6.2 | 5.35 à 7.15 | Mesure de la vapeur d'eau à mi-atmosphère. Chaque canal correspond à une couche atmosphérique différente. Calculs des vents. |
| C6 | WV 7.3 | 6.85 à 7.85 | |
| C7 | IR 8.7 | 8.3 à 9.1 | Informations quantitatives sur les cirrus. Distinction entre les nuages de glace et les nuages d'eau. |
| C8 | IR 9.7 | 9.38 à 9.94 | Surveillance de l'ozone total. Mesure de sa concentration dans la basse stratosphère. Indication des champs de vents à cette altitude. |
| C9 | IR 10.8 | 9.8 à 11. 8 | Infrarouge thermique. Mesure de la température des nuages et de la surface. Calcul des vents. Estimation de l'instabilité atmosphérique. |
| C10 | IR 12 | 11 à 13 | |
| C11 | IR 13.4 | 12.4 à 14.4 | Absorption du $CO_2$. Estimation de l'instabilité atmosphérique. Estimation de la température de la basse troposphère. Mesure de la hauteur des nuages semi-transparents. |
| C12 | HRV | 0.5 à 0.9 | Large bande dans le visible comme le canal VIS de Météosat mais une résolution spatiale plus fine : 1km au lieu de 2,5 km. |

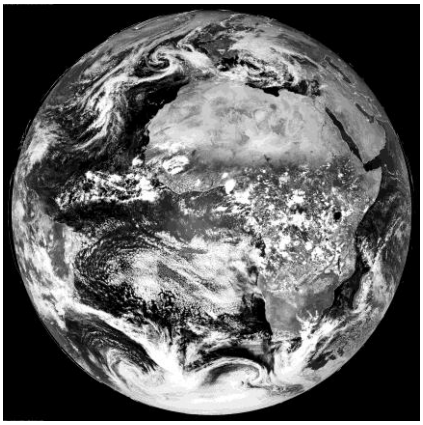


Figure 4.2. Image de la surface terrestre.

La figure 4.2 représente une image terrestre acquise par le satellite MSG pour le canal $VIS$ 0.6 en date du 22 Octobre 2007. Les images MSG que nous avons utilisées couvrent essentiellement la zone sud de l'Europe et le nord de l'Afrique. Ces images sont issues de différents canaux comme le montre la figure 4.3.

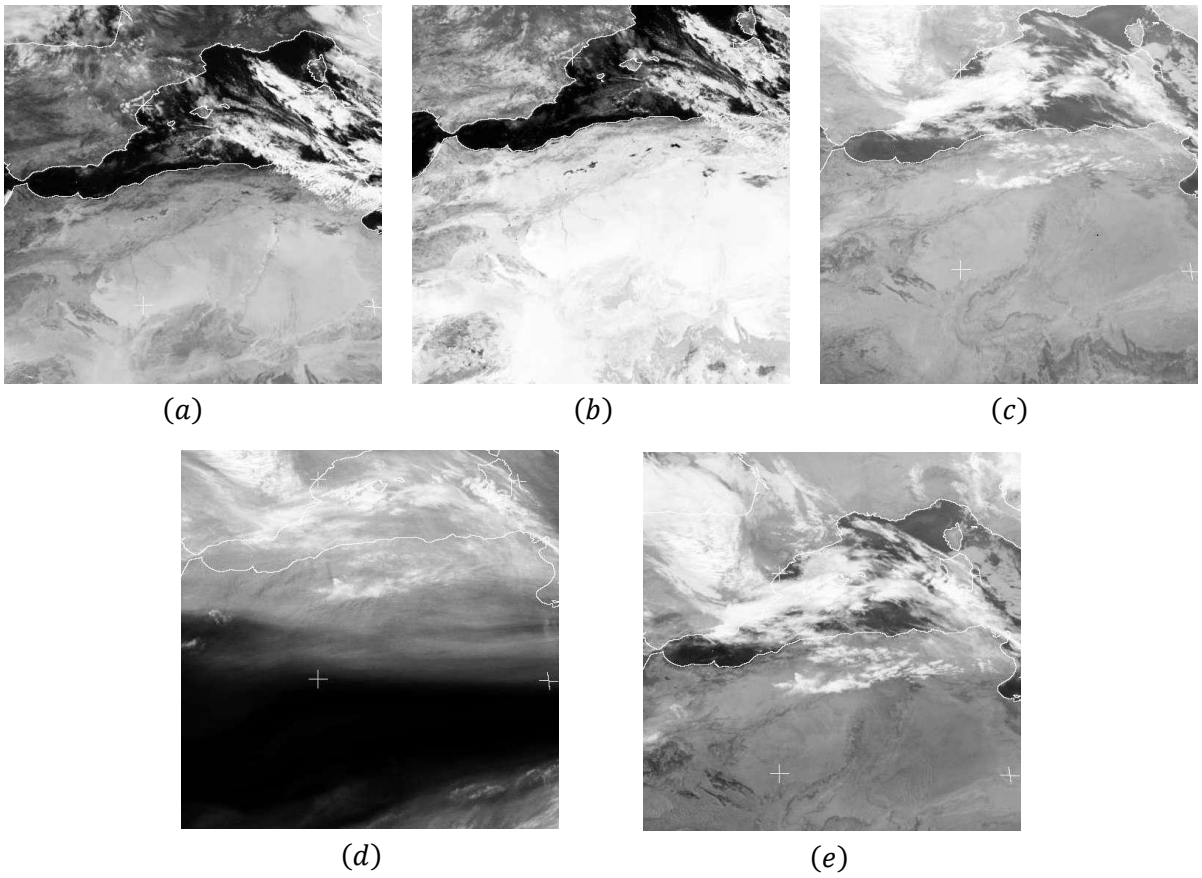



Figure 4.3. Les différentes images MSG de la base de données.
(a) VIS 0.6; (b) NIR 1.6; (c) IR 3.9; (d) WV 6.2; (e) IR 12.

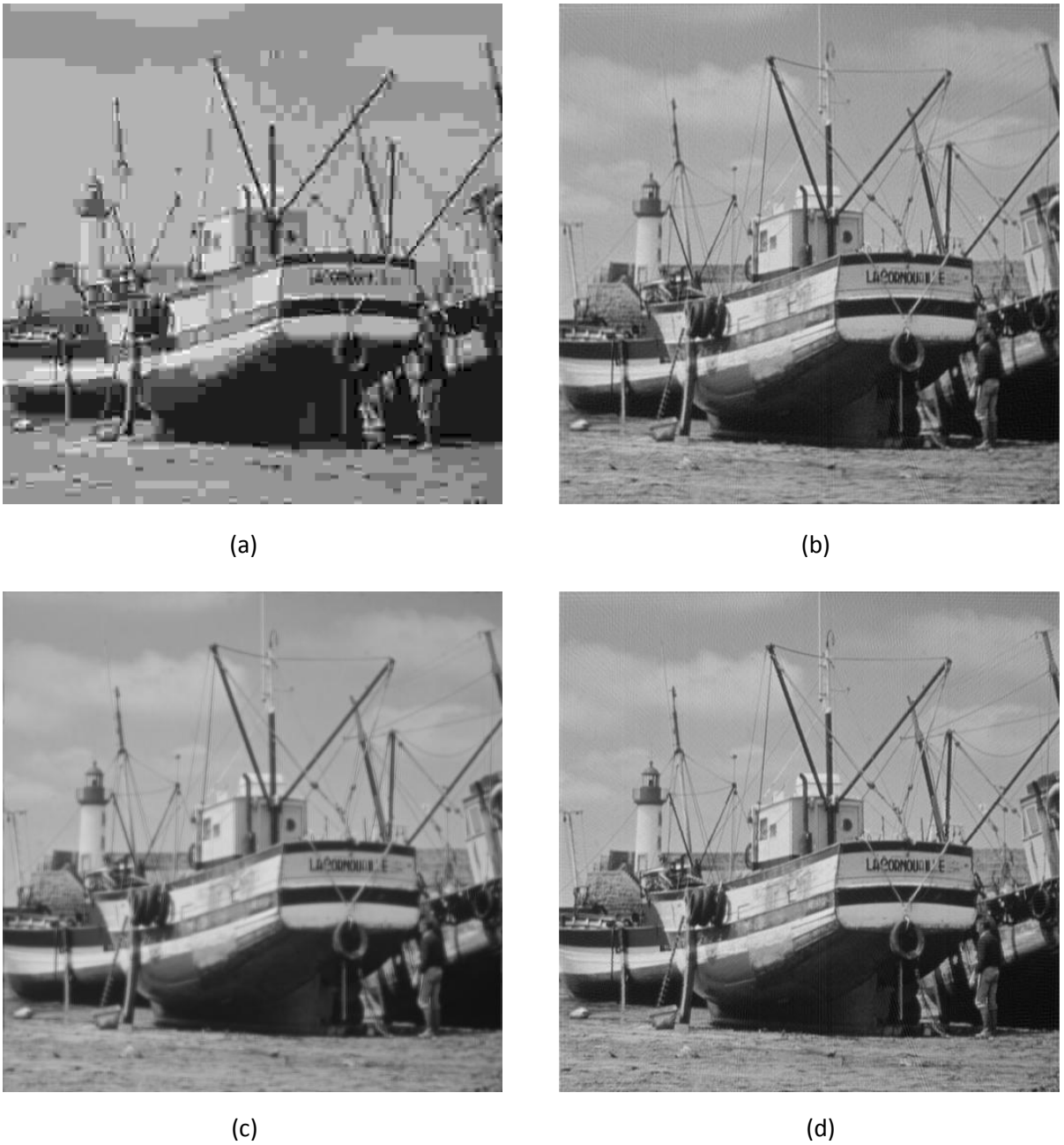
## 4.4. Tests et résultats

Nous avons comparé notre méthode avec trois dont deux basées sur la DCT et une sur les ondelettes bi-orthogonales. Le but de ce test est de vérifier la résistibilité de la transformée de Radon au bruit de quantification par le biais de la RAD-DCT et de la DCT et de positionner notre méthode par rapport à ces méthodes.

### 4.4.1. Résultats obtenus sur des images tests

La figure 4.4 représente la reconstruction de l'image test Boat pour les quatre méthodes. Le choix d'un niveau de quantification égal à 15, qui est élevé est justifié pour montrer l'efficacité de la méthode RAD-DCT par rapport à la méthode DCT basique. D'après la figure 4.4, nous remarquons que le bruit de quantification est imperceptible pour la méthode RAD-DCT (image (b)) contrairement à la méthode DCT basique (image (a)) où l'effet de pixellisation et remarquable. Cela est dû, comme nous l'avons indiqué au deuxième chapitre à la résistibilité de la transformée de Radon aux bruits car les fluctuations d'intensité dues aux bruits sont atténuées par le processus d'intégration **[Murphy 1986]**. La méthode bi-orthogonale DWT permet de reconstruire une image (image (c) avec un effet de flou. Nous remarquons que l'image reconstruite par notre méthode RAD-LPC (image (d)) est de qualité meilleure par rapport aux images reconstruites par les autres méthodes.

Les différents résultats des critères de mesure sont listés dans le tableau 4.2.



(a) (b)

(c) (d)

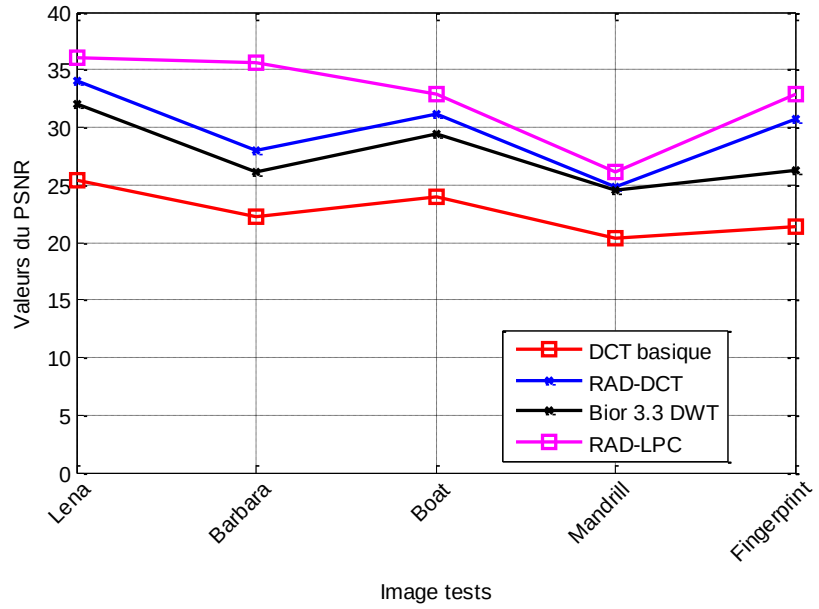
Figure 4.4. Images boat reconstruites.

(a) Méthode DCT basique : $T_c = 98.35$ %, $PSNR = 23.93\ db$, $MAE = 12.09$, $RMSE = 16.26$.
(b) Méthode RAD-DCT : $T_c = 44.40$ %, $PSNR = 31.09\ db$, $MAE = 5.73$, $RMSE = 7.46$.
(c) Méthode bi-orthogonale DWT : $T_c = 27$ %, $PSNR = 29.45\ db\ MAE = 5.35$, $RMSE = 8.62$.
(d) Méthode RAD-LPC :$T_c = 99.90$ %, $PSNR = 32.88\ db\ MAE = 4.29$, $RMSE = 5.80$.

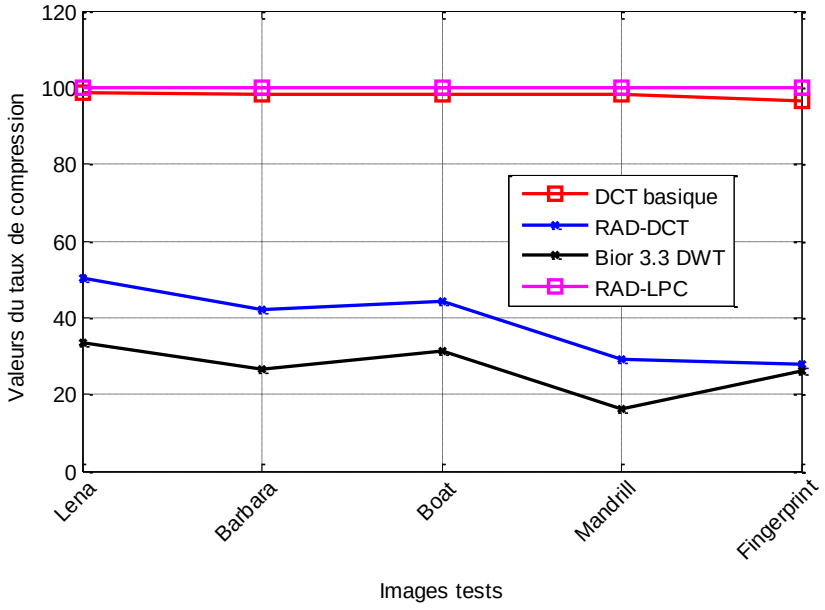
Tableau 4.2. Les différents résultats pour les quatre méthodes.

| Méthode | Image | $T_c(\%)$ | $PSNR$ (db) | $MAE$ | $RMSE$ |
|---|---|---|---|---|---|
| DCT basique | Lena | 98.48 | 25.45 | 10.42 | 13.67 |
| | Barbara | 98.31 | 22.23 | 14.45 | 19.81 |
| | Boat | 98.35 | 23.93 | 12.09 | 16.26 |
| | Mandrill | 98.10 | 20.41 | 18.21 | 24.42 |
| | Fingerprint | 96.37 | 21.31 | 17.41 | 22.00 |
| RAD-DCT | Lena | 50.27 | 34.02 | 3.55 | 5.09 |
| | Barbara | 42.05 | 27.96 | 7.08 | 10.23 |
| | Boat | 44.40 | 31.09 | 5.73 | 7.46 |
| | Mandrill | 28.93 | 24.75 | 11.50 | 14.80 |
| | Fingerprint | 27.76 | 30.79 | 5.66 | 7.38 |
| Bi-orthogonale DWT | Lena | 33.46 | 32.02 | 3.78 | 6.41 |
| | Barbara | 26.54 | 26.12 | 7.68 | 12.65 |
| | Boat | 31.37 | 29.45 | 5.35 | 8.62 |
| | Mandrill | 16.11 | 24.47 | 10.79 | 15.30 |
| | Fingerprint | 26.25 | 26.23 | 10.26 | 12.49 |
| RAD-LPC | Lena | 99.90 | 36.04 | 3.10 | 4.23 |
| | Barbara | 99.90 | 35.54 | 6.23 | 9.04 |
| | Boat | 99.90 | 32.88 | 4.29 | 5.80 |
| | Mandrill | 99.90 | 26.17 | 7.57 | 10.25 |
| | Fingerprint | 99.90 | 32.90 | 4.30 | 5.79 |

Nous présentons par la suite les résultats obtenus lors du test des quatre méthodes sous forme de courbes du taux de compression et du $PSNR$ en fonction de différentes images comme le montre la figure 4.5.




(a)




(b)

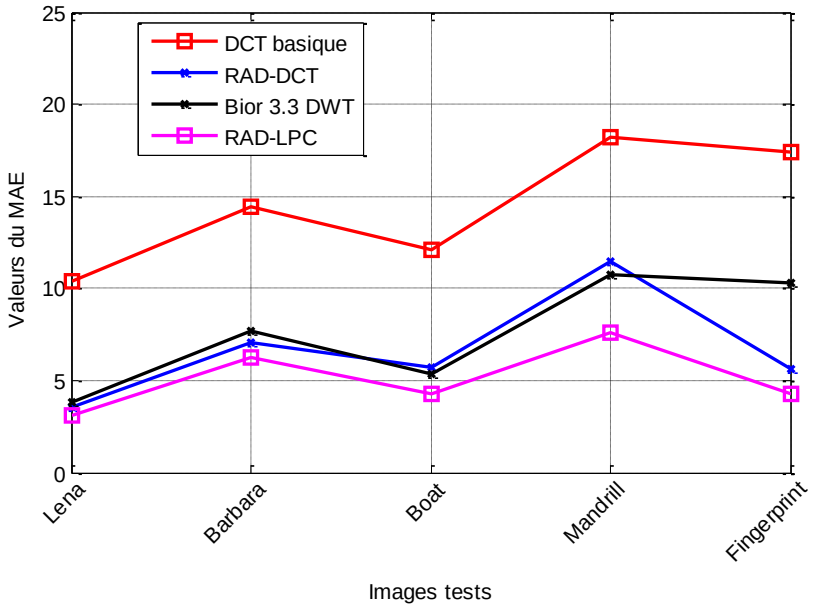
Figure 4.5. Variations du *PSNR* et du taux de compression $T_c$.

(a) Variations du *PSNR* en fonction des images tests pour les méthodes : DCT basique, RAD-DCT, bi-orthogonale et RAD-LPC.

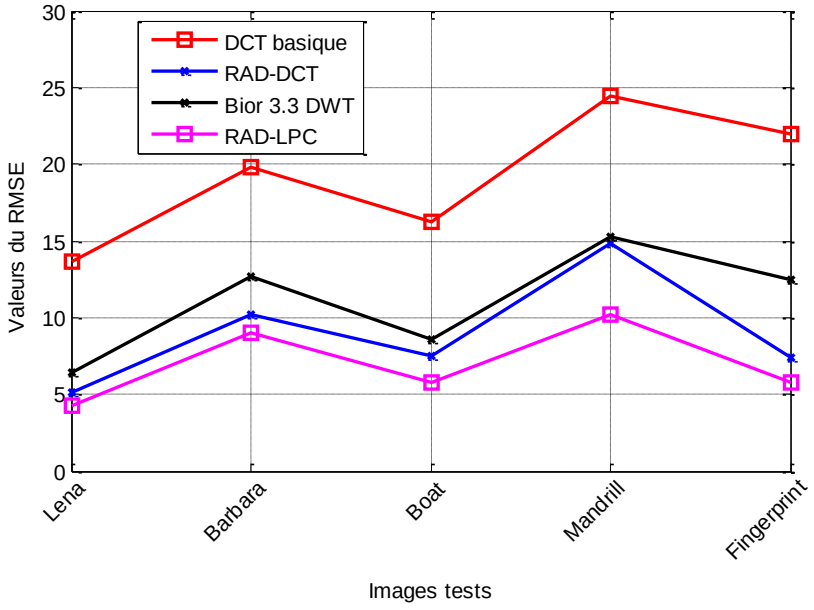
(b) Variations du taux de compression en fonction des images tests pour les méthodes : DCT basique, RAD-DCT, bi-orthogonale et RAD-LPC.

Nous remarquons d'après la figure 4.5, que la méthode RAD-LPC assure un bon compromis entre le taux de compression et le $PSNR$. Elle permet aussi d'obtenir un taux de compression très élevé et fixe tout en préservant la qualité de l'image reconstruite. Ce résultat est assuré par le LPC, car il permet de décorréler au maximum les points Radon, donc les pixels et ne garder que l'information utile. La bi-orthogonale DWT, comme la RAD-DCT permet d'obtenir un faible taux de compression mais elle permet de reconstruire des images moins dégradées tandis que la méthode DCT basique assure un taux de compression très élevée mais dégrade de plus les images. Cela est dû à la forte quantification qui engendre un même codage pour l'ensemble des blocs (peu de coefficients non nuls pour les représenter).

Pour confirmer ces résultats, nous faisons appel aux critères de déviation comme le $RMSE$ et $MAE$ comme le montre la figure 4.6. Comparons les résultats des différentes courbes, le calcul de déviation montre clairement que notre méthode RAD-LPC génère après décompression des images plus proches de celles introduites car les valeurs du $MAE$ et du $RMSE$ sont plus faibles et proches du zéro par rapport à celles des autres méthodes.




(a)

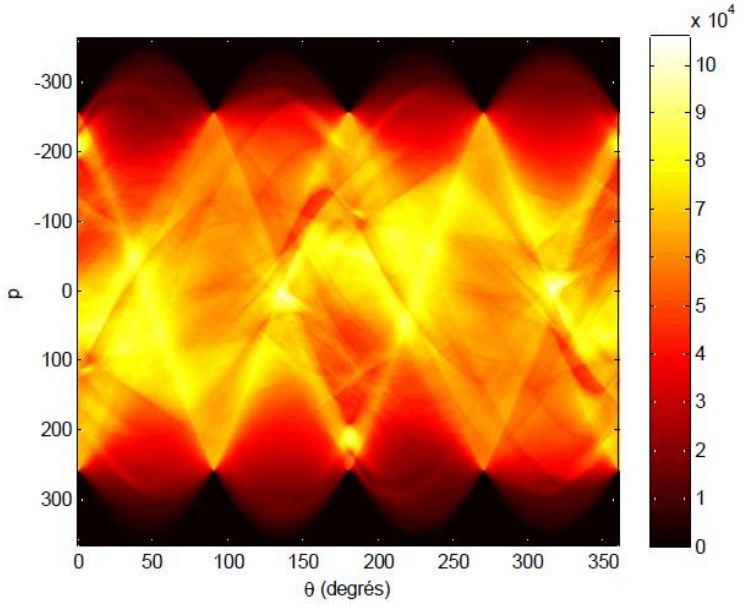



(b)

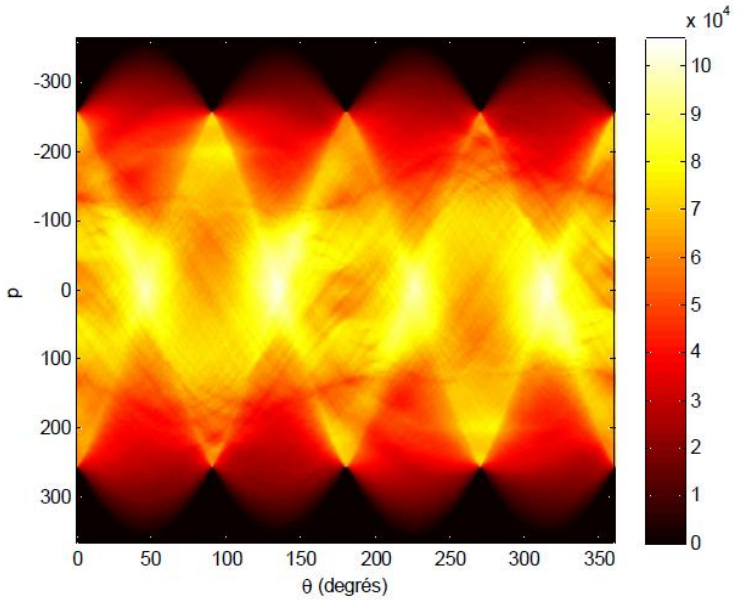
Figure 4.6. Variations du $MAE$ et du $RMSE$.

(a) Variations du MAE en fonction des images tests pour les méthodes : DCT basique, RAD-DCT, bi-orthogonale DWT et RAD-LPC.
(b) Variations du RMSE en fonction des images tests pour les méthodes : DCT basique, RAD-DCT, bi-orthogonale DWT et RAD-LPC.

Nous remarquons aussi pour les quatre méthodes que les valeurs du $PSNR, MAE$ et $RMSE$ sont maximales pour l'image Lena, mais elles sont minimales pour l'image Mandrill. Cela est expliqué, comme nous le remarquons d'après leurs sinogrammes représentés par la figure 4.7 par la complexité de la texture de l'image Mandrill par rapport à celle de Lena.




(a)




(b)

Figure 4.7. Sinogramme des images Lena et Mandrill.

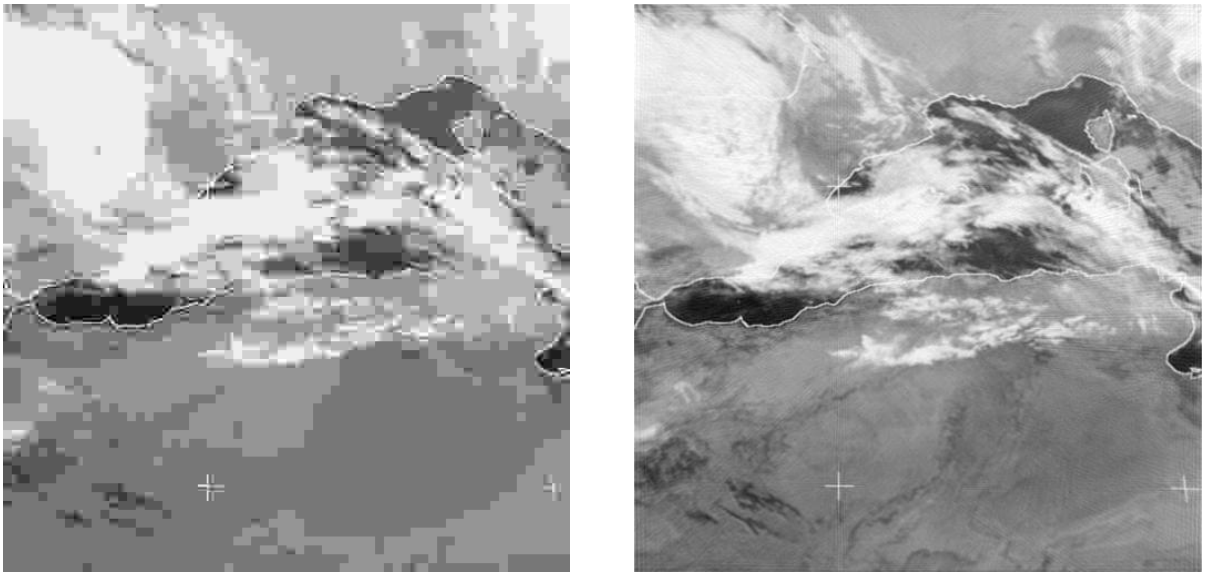
(a) La transformée de Radon de l'image Lena.
(b) La transformée de Radon de l'image Mandrill.

### 4.4.2. Résultats obtenus sur des images MSG

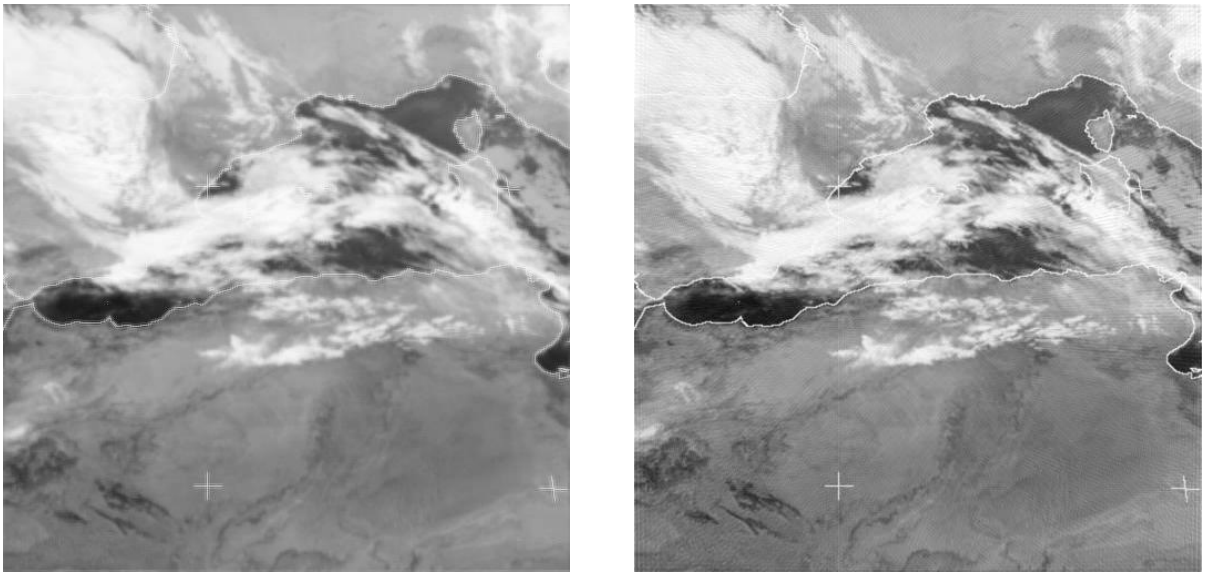
Nous utilisons pour les mêmes méthodes cinq images MSG. Nous avons choisi celles des canaux $VIS$ 0.6, $NIR$ 1.6, $IR$ 3.9, $WV$ 6.2, et $IR$ 12. Nous gardons le même niveau de quantification, qui est 15 pour les méthodes basées sur la DCT et le même filtre $bior3.3$ pour la bi-orthogonale DWT. La figure 4.8 nous donne la reconstruction de l'image MSG du canal $IR$ 12 pour les quatre méthodes.

La figure 4.8 montre une fois encore que notre méthode RAD-LPC permet d'obtenir la meilleure qualité d'image après décompression et elle préserve mieux les détails importants comme les nuages (image (d)). Comme dans le cas des images tests, la méthode DCT engendre des images de qualité inférieures (image(a)) dont l'effet de pixellisation et remarquable. Cela est dû à la forte quantification. Les deux méthodes bi-orthogonale DWT (image(c)) et la RAD-DCT (image(b)) permettent d'obtenir après décompression des images de qualité moyennes, inférieure à celles de notre méthode RAD-LPC avec un effet de flou pour la méthode bi-orthogonale DWT ce qui rend la détection des nuages difficile. La méthode RAD-DCT montre une fois encore la robustesse de la transformée de Radon vis-à-vis au bruit de quantification.

Les différents résultats des critères de mesure pour les images MSG sont représentés dans le tableau 4.3.



(a) (b)



(c) (d)

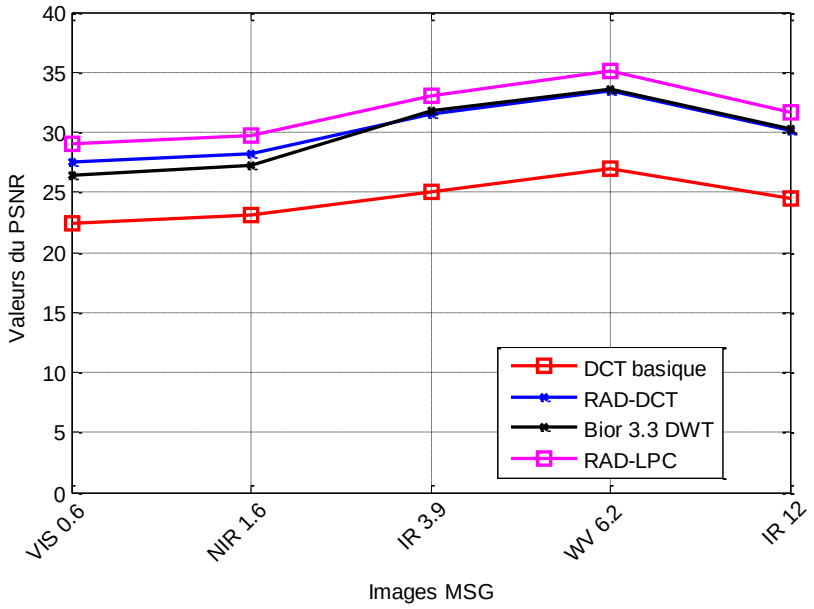
Figure 4.8. Image IR 12.0 reconstruite.

(a) Méthode DCT basique : $T_c = 98.61$ %, $PSNR = 24.44\ db$, $MAE = 10.87$,$RMSE = 15.35$.
(b) Méthode RAD-DCT : $T_c = 44.57$ %, $PSNR = 30.11\ db$, $MAE = 4.56$,$RMSE = 7.98$.
(c) Méthode bi-orthogonale DWT : $T_c = 37.88$ %, $PSNR = 30.22\ db$, $MAE = 4.29$, $RMSE = 7.89$.
(d) Méthode RAD-LPC : $T_c = 99.90$ %, $PSNR = 31.69\ db$, $MAE = 4.01$,$RMSE = 6.66$.

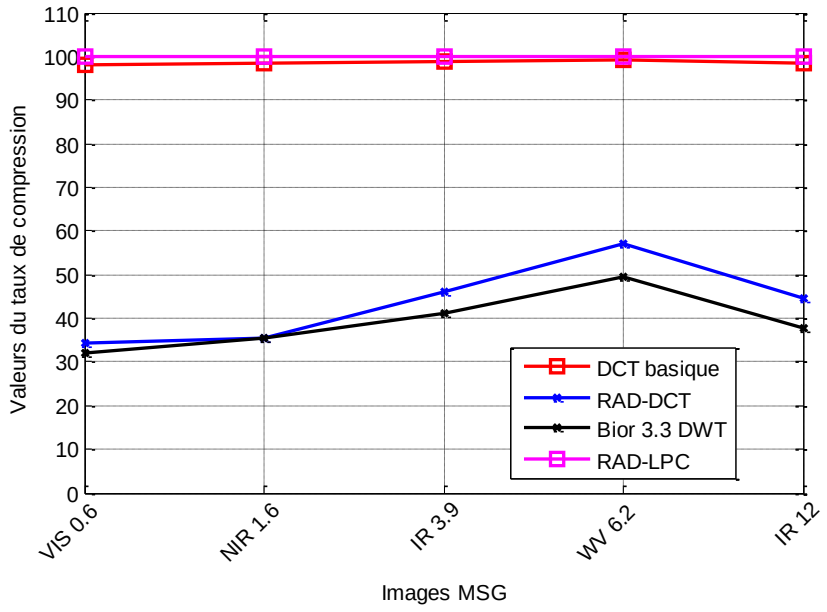
Tableau 4.3. Les différents résultats pour les quatre méthodes concernant les images MSG.

| Méthode | Image | $T_c(\%)$ | $PSNR$ (db) | $MAE$ | $RMSE$ |
|---|---|---|---|---|---|
| DCT basique | VIS 0.6 | 98.07 | 22.48 | 13.23 | 19.23 |
| | NIR 1.6 | 98.33 | 23.17 | 12.20 | 17.77 |
| | IR 3.9 | 98.89 | 25.05 | 10.68 | 14.31 |
| | WV 6.2 | 99.17 | 26.94 | 8.03 | 11.52 |
| | IR 12 | 98.61 | 24.44 | 10.87 | 15.35 |
| RAD-DCT | VIS 0.6 | 34.38 | 27.49 | 6.30 | 10.78 |
| | NIR 1.6 | 35.47 | 28.24 | 5.77 | 9.90 |
| | IR 3.9 | 46.24 | 31.57 | 3.98 | 6.75 |
| | WV 6.2 | 57.05 | 33.40 | 2.84 | 5.47 |
| | IR 12 | 44.57 | 30.11 | 4.56 | 7.98 |
| Bi-orthogonale DWT | VIS 0.6 | 31.94 | 26.45 | 5.99 | 12.18 |
| | NIR 1.6 | 35.57 | 27.24 | 5.48 | 11.12 |
| | IR 3.9 | 40.97 | 31.74 | 3.79 | 6.62 |
| | WV 6.2 | 49.39 | 33.63 | 2.76 | 5.33 |
| | IR 12 | 37.88 | 30.22 | 4.29 | 7.89 |
| RAD-LPC | VIS 0.6 | 99.90 | 29.02 | 5.74 | 9.06 |
| | NIR 1.6 | 99.90 | 29.80 | 5.25 | 8.28 |
| | IR 3.9 | 99.90 | 33.04 | 3.54 | 5.70 |
| | WV 6.2 | 99.90 | 35.05 | 2.32 | 4.52 |
| | IR 12 | 99.90 | 31.69 | 4.01 | 6.66 |

En analysant les résultats du tableau 4.3 et de la figure 4.9 des variations du $PSNR$ et du taux de compression $T_c$ en fonction des images MSG. Nous remarquons, en général que les résultats obtenus avec les images MSG sont inférieurs à ceux obtenus avec les images tests. Cela est expliqué par la complexité des images MSG.




(a)




(b)

Figure 4.9. Variations du $PSNR$ et du taux de compression $T_c$.

(a) Variations du PSNR en fonction des images MSG pour les méthodes : DCT basique, RAD-DCT, bi-orthogonale DWT et RAD-LPC.

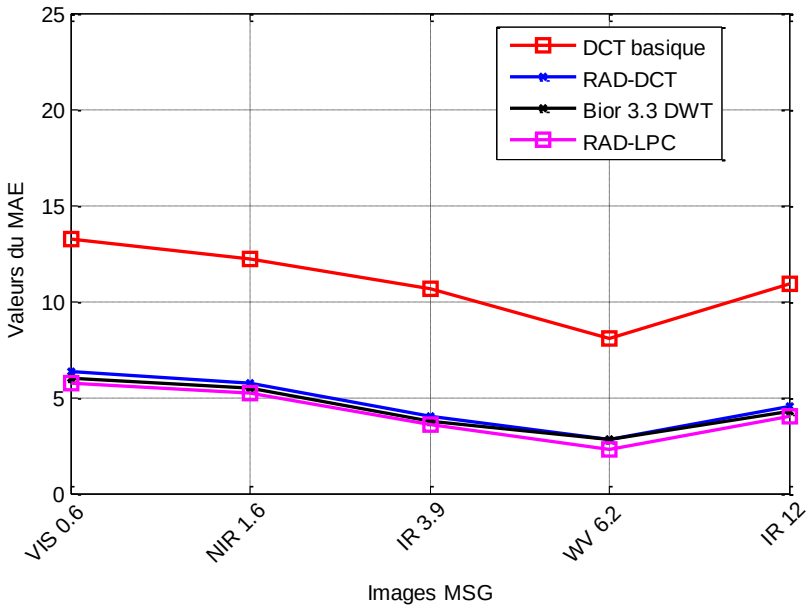
(b) Variations du taux de compression en fonction des images MSG pour les méthodes : DCT basique, RAD-DCT, bi-orthogonale DWT et RAD-LPC.

Comme nous l'avons indiqué lors du chapitre précédent et comme le montre la figure 4.9, notre méthode RAD-LPC assure un taux de compression très élevé et fixe quel que soit la nature des images compressées tout en préservant la qualité des images qui est meilleure par rapport à celle reconstruite par les autres méthodes. Cela est dû essentiellement à la forte décorrélation des points Radon par le codage prédictif LPC et le codage RLC des longues séquences des mots-code identiques ce qui permet de créer un fichier réduit par rapport à celui créer lors de l'utilisation du codage RLC pour la séquence binaire.

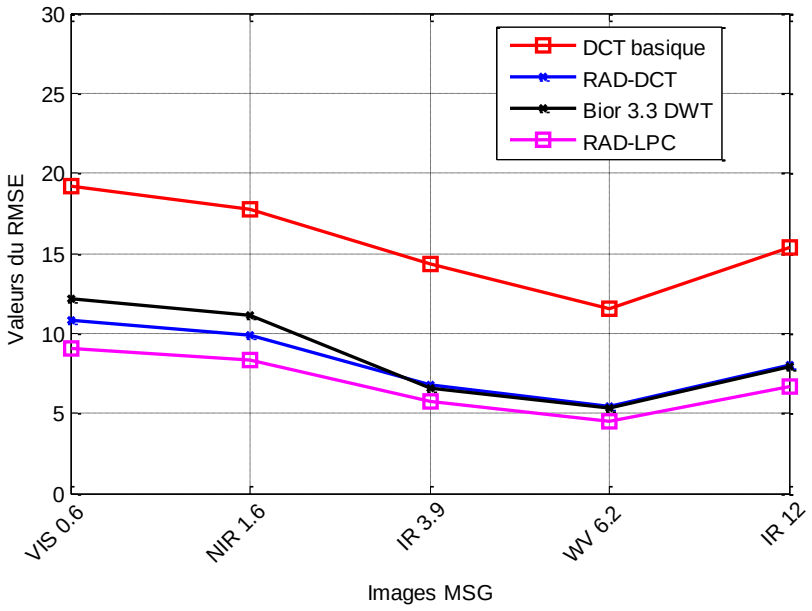
Les deux méthodes RAD-DCT et la bi-orthogonale DWT sont proches malgré la forte quantification pour la DCT. La stabilité de la méthode RAD-DCT est assurée par la transformée de Radon qui atténue les bruits par le processus d'intégration **[Murphy 1986]** dont chaque projection est filtrée par le filtre Ram-Lak

La méthode DCT basique permet d'obtenir un taux de compression élevé mais elle dégrade de plus la qualité des images MSG décompressées à cause de la forte quantification.

Les résultats du $PSNR$ et du taux de compression sont soutenus par ceux des critères de déviation comme le $RMSE$ et le $MAE$. La figure 4.10 représente leurs variations en fonction des images MSG utilisées et comme le montre, les images MSG reconstruites lors de la décompression par la méthode RAD-LPC sont les plus proches des images initiales car elles assurent des mesures de prédictions les plus faibles.




(a)

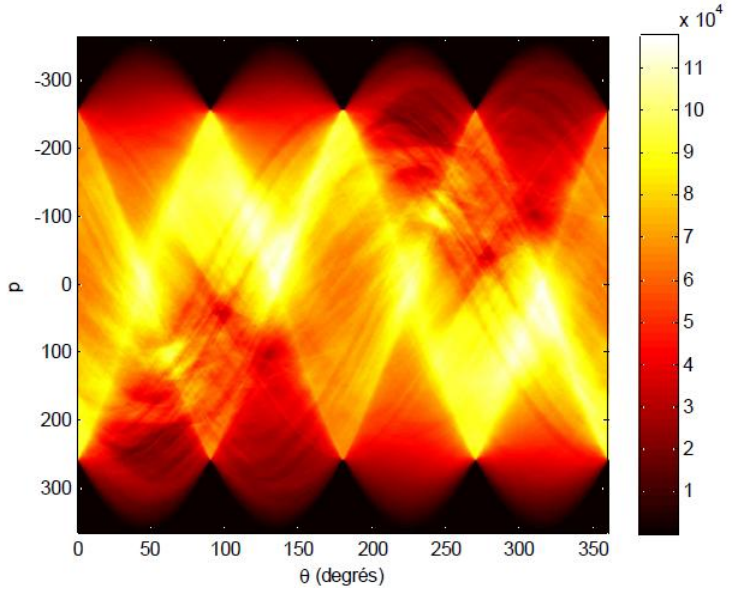



(b)

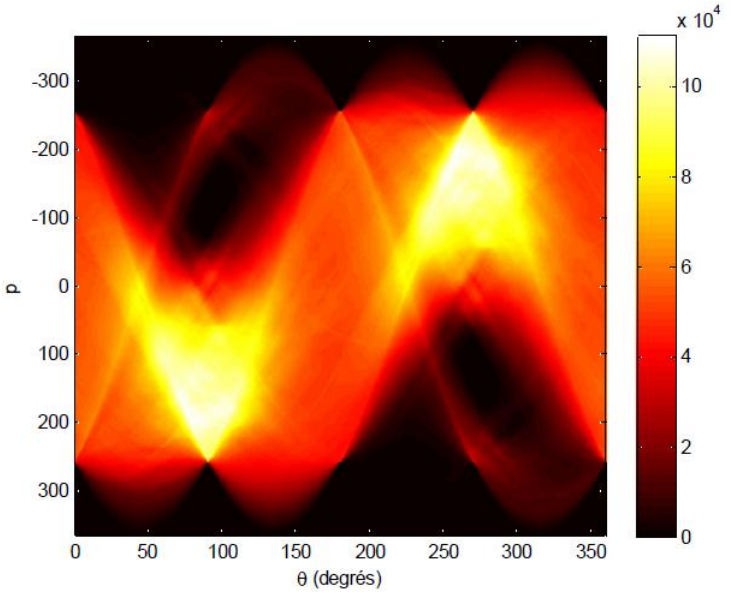
Figure 4.10. Variations du $MAE$ et du $RMSE$.

(a) Variations du MAE en fonction des images MSG pour les méthodes : DCT basique, RAD-DCT, bi-orthogonale DWT et RAD-LPC.
(b) Variations du RMSE en fonction des images MSG pour les méthodes : DCT basique, RAD-DCT, bi-orthogonale DWT et RAD-LPC.

Nous remarquons aussi pour les quatre méthodes de compression d'images que les valeurs du $PSNR$, $MAE$ et $RMSE$ sont maximales pour les images vapeur d'eau $WV$ 6.2, par contre, elles sont minimales pour les images visibles $VIS$ 0.6. Cela est dû, comme le montre la figure 4.11 à la forte texture des images $VIS$ 0.6 par rapport à celle des images MSG des autres canaux. Ainsi, la perte des détails fins sera élevée par rapport aux autres images lors de la compression.




(a)




(b)

Figure 4.11. Sinogramme des images MSG $VIS$ 0.6 et $WV$ 6.2.
(a) La transformée de Radon de l'image MSG $VIS$ 0.6.
(b) La transformée de Radon de l'image MSG $WV$ 6.2.

## 4.5. Discussion

Ce chapitre a été consacré à l'évaluation de la qualité associée à la compression d'images. L'application de notre méthode de compression d'images sur une base de données constituée des images tests et Météosat a donné de bons résultats. Elle permet d'obtenir un taux de compression fixe et élevé quel que soit l'image introduite. Un taux de compression qui vaut 99.90 % avec un $PSNR$ égal à 36.04 $db$ pour l'image Lena et de 35.05 $db$ pour l'image MSG $WV$ 6.2. Ces résultats sont soutenus par ceux des critères de déviation comme $RMSE$ et $MAE$. Ces derniers présentent des valeurs les plus faibles pour les images reconstruites après décompression par notre méthode. Les résultats de tous critères confondus montrent également que notre méthode de compression d'images assure un bon compromis débit-distorsion.

# Conclusion

Le satellite MSG permet de collecter douze fichiers images toutes les quinze minutes pour des résolutions différentes. Ce processus engendre des bases de données de volume important et crée des problèmes de transmission et de stockage en mémoire. Ces difficultés nous ont poussés à réaliser une méthode de compression d'images avec pertes.

L'état de l'art que nous avons abordé au premier chapitre nous a permis de construire le schéma de notre méthode de compression avec pertes basée sur la transformée de Radon et le codage prédictif. La transformée de Radon nous donne la possibilité de traiter seulement les sommes des pixels (points Radon) pour chaque angle. Sa puissance réside dans sa résistance aux différents bruits. Le codage prédictif LPC nous a assuré une forte décorrélation des points Radon, donc des pixels à travers un prédicteur qui a comme design les coefficients du filtre SNN pour mieux préserver les contours, c'est-à-dire les hautes fréquences. Le codage arithmétique qui forme le noyau du codage prédictif nous a permis de coder les prédictions résiduelles pour les représentées seulement par un seul mot-code. La séquence générée par le codage arithmétique contient des longues séquences de symboles décimaux identiques et donc un codage RLC était un choix adéquat pour augmenter le taux de compression. Ce dernier nous a donné d'excellents résultats : un taux de compression élevé et fixe quel que soit l'image introduite.

Pour montrer l'efficacité de notre méthode de compression d'images, de justifier le choix de transformée de Radon et d'exposer sa puissance envers les bruits comme le bruit de quantification, nous l'avons comparés avec d'autres méthodes de mêmes types qui sont : RAD-DCT, bi-orthogonale et DCT basique.

L'application de notre méthode sur une base de données constituée d'images tests très utilisées en compression d'images et d'images Météosat a permis d'atteindre un taux de compression de 99.90% tout en gardant la qualité des images après décompression. Ces résultats montrent clairement que notre méthode est qualifiée pour être un bon choix pour l'archivage et la transmission des images Météosat.

La taille de la matrice des projections 1-D augmente proportionnellement avec la taille des images et donc, des temps de compression élevés.

Afin d'améliorer nos résultats, nous donnons les perspectives suivantes :

- Application de la transformée Mojette pour reconstruire les images introduites à partir d'un nombre de projections fini et d'éviter les artefacts causés par le processus d'interpolation.
- Utilisation des langages de programmation puissants comme le C, C++, C# et Java pour réduire le temps d'exécution et exploiter cette méthode de compression d'images pour une transmission en temps réel sans dépendance de Matlab.

# Bibliographie

**A. Moreno, G. Miramontes, E. Garcia, C. Sifuentes, R. Magallanes and E. De la rosa,** « La transformada Mojette », DIFU100ci@ pp. 28 – 33 Vol. 5, No. 2, Mexico, septiembre-diciembre 2011.

**A. Kingston, F. Autrusseau,** « Lossless Image Compression via Predictive Coding of Discrete Radon Projections, » Signal Processing Image Communication, vol. 23, no. 4, pp. 313–324, Jun. 2008.

**A. Al Muhit, M. S Islam and M. Othman,** « Design and analysis of discrete wavelet transform (DWT) for image compression using VHDL », in Proc. of the International Conference on Parallel and Distributed Processing Techniques and Applications, PDPTA 2005, vol.1. CSREA Press 2005, pp. 157-160, Las Vegas, Nevada, USA, 2005, ISBN 1-932415-58-0.

**A. Said, W. A. Pearlman,** « Digital signal compression principles and practice » Cambridge university press, p. 440, 2011.

**A. Devi, M. G. Mini,** « Gray scale image compressed based on wavelet transform and linear prediction », The International Journal of Multimedia & Its Applications (IJMA) vol.4, no.1, pp. 47-62, February 2012.

**B. P. Tunstall,** « Synthesis of noiseless compression codes », thèse de Doctorat, Georgia Institute of Technology, USA, p.108, 1967.

**B. Isabelle, Cours,** « Reconstruction d'images de tomographie » Télécom ParisTech, département TSI, CNRS UMR 5141, 46 rue Barrault, 75634, p.28, France.

**C. J. Willmott, K. Matsuura,** « Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance », Center of climatic research, Department of Geography, vol. 30,no.1, pp.79–82, USA, 2005.

**C. Chapin,** « Differential quantization of communication signals », U.S. patent 2605361, filed June 29, 1950, issued July 29, 1952.

**C. Valade,** « Compression d'images complexes avec pertes : application à l'imagerie Radar », thèse de Doctorat en traitement d'images et des signaux, école nationale supérieure des télécommunications, Paris, p.221, France, 2006.

**D. Harwood, M. Subbarao, H. Hakalahti and L. S. Davis,** « A new class of edge-preserving smoothing filters », Pattern Recognition Letters, Elsevier, Vol.6, Issue.5, pp. 155-162, USA, December 1987.

**D. A. Huffman,** « A method for the construction of minimum redundancy codes », Proceedings of the IRE, Massachusetts Institute of Technology, vol. 40, Issue.9, pp.1098-1101, USA, September 1952.

**Eduscol,** document officiel du capteur SEVIRI (Spinning Enhancer Visible and Infrared Imager), « http://eduscol.education.fr/orbito/system/meteosat/met23.htm ».

**E. Woods, S. Eddins and R. Gonzalez,** « Digital image processing using Matlab », 2nd Ed, Gatesmark publishing, p.827, 2009.

**Education Meteofrance,** Site éducatif dans le domaine météorologique, « http://education.meteofrance.fr/observer-et-mesurer/l-atmosphere/les-principaux-satellites-operationnels ».

**F. Dubois,** « RECONSTRUCTION des images tomographiques par rétroprojection filtrée», Revue de l'ACOMEN, vol.4, no.2, pp.92–99, FRANCE, 1998.

**G. Wolberg,** « Digital image warping », IEEE Computer Society Press, Los Alamitos, CA, USA, 1994.

**G. T. Herman,** « Two direct methods for reconstructing pictures from their projections: a comparative study ». Comput. Graphics Image Process, vol.2, no.1, pp.123–144, August 1972.

**G. N. Ramachandran, A. V. Lakshminarayanan** « Three-dimensional reconstruction from radiographs and electron micrographs: application of convolutions instead of Fourier transforms » Proceedings of the National Academy of Sciences, vol.68, no.9, pp.2236–2240, USA, September 1971.

**ISO/IEC 11172,** « International standard (MPEG-1), Information technology—coding of moving pictures and associated audio for digital storage media at up to 1.5 Mb/s », p.112, 1993.

**J. Mitchell, W. Pennebaker, C. Fogg and D. LeGall,** « MPEG video compressions standard », Kluwer Academic Publishers, p.509, October 1996.

**J. Radon,** « On the determination of functions from their integrals along certain manifolds », translation of Radon's 1917 paper by R. Lohner, « The Radon transform and some of its applications », Annexe A, pp.204-217,John Wiley & Sons, 1983.

**J. Fessler,** « Analytical tomographic image reconstruction methods », Chapter 3, College of Engineering, University of Michigan, p.47, November 2009.

**J. Guédon,** « Psychovisual image coding via an exact discrete Radon transform » Proc. Visual Communications & Image Processing (VCIP), Lance T. Wu editor, pp.562-572, Taipei, Taiwan ,May 1995.

**J. J. Rissanen and G. G. Langdon,** « Arithmetic coding », IBM J. Res. Dev., 23 (2), 149–162, 1979.

**K. S. Thyagarajan,** « Still image and video compression with Matlab », A John Wiley & Sons publication, p.442, 2011.

**K. Sayood,** « Introduction to data compression », 4th Edition, Morgan Kaufmann, p.743, 2012.

**L. M. Murphy,** « Linear feature detection and enhancement in noisy images via the Radon transform », Pattern Recognition Letters, vol.4, Issue.4, pp.279-284, 1986.

**M. B. Katz,** « Questions of uniqueness and resolution in reconstruction from Projections », Lecture notes in Biomathematics, Springer Verlag, p.175, 1977.

**M. Lahdir, S. Ameur, and L. Akrour,** « Codage d'images numériques par fractales dans le domaine de la DCT », 3rd International Conference : Sciences of Electronics, pp.131-135, Tunisia, 2005.

**M. Lahdir, S. Ameur, and A. Adane,** « algorithme non itératif, basé sur les ondelettes biorthogonales et les fractales, pour la compression d'images satellitaires », Télédétection, vol.6, no.4, pp.345-360, www.teledetection.net, ISSN 1028-7736, 2007.

**N. Ahmed, T. Natarajan and K. Rao,** « Discrete Cosine Transform », IEEE Computers Transactions, vol.23, no.1, pp.90-94, 1974.

**N. Normand, J. Guédon,** « Controlled redundancy for image coding and high-speed transmission », SPIE Digital Library, vol. 2727, pp.1070-1081, February 1996.

**O. Marques,** « Practical image and video processing using Matlab», John Wiley & Sons, publication and IEEE Press, p.696, 2011.

**P. Devaki, D. G. Raghavendra,** « Lossless reconstruction of secret image using threshold secret sharing and transformation », International Journal of Network Security & Its Applications (IJNSA), vol.4, no.3, pp.111-119, May 2012.

**P. Bourguet, Cours,** « Reconstruction tomographique », Biologie Médicale (Master1), p.53, université Rennes, 2007.

**P. Courmontagne,** « Transformée de Radon et filtrage Application à la détection de sillages de mobiles marins », GRETSI, Saint Martin d'Hères, vol.15, no.4, pp.297-307, France, 1998.

**R. Kumar, M. Rattan,** « Analysis of various quality metrics for medical image processing », International Journal of Advanced Research in Computer Science and Software Engineering vol.2, Issue.11, pp.137-144, November 2012.

**R. F. Rice, R. Plaunt,** « Adaptive variable-length coding for efficient compression of spacecraft television data », IEEE Transactions on Communications, vol.6, no.9, pp.889–897, December 1971.

**R. Tyagi, D. K. Sharma,** « Digital image compression comparisons using DPCM and DPCM with LMS algorithm », International Journal of Computer Applications & Information Technology, vol.1, Issue.2, (ISSN: 2278-7720), September 2012.

**R. Lohner,** « The Radon transform and some of its applications », Annexe A, John Wiley & Sons, pp.204-217, 1983.

**S. A. Pradeep, R. Manavalan,** « Image compression using Radon transform With DCT: Performance analysis», International Journal of Scientific Engineering and Technology, vol.2, no.8, pp.759-765, 2013.

**S. Ameur, A. Adan and M. Lahdir,** « Compression d'images Meteosat en sous-bandes par transformation en cosinus et quantification vectorielle », Télédétection, vol.2, no.4, pp.255-266, 2002.

**S. R. Deans,** « The Radon transform and some of its applications », John Wiley & Sons, p.300, 1983.

**W. B. Pennebaker,** « Motion vector search strategies for MPEG-1 », Technical Report ESC96–002, Encoding Science Concepts, Inc., February 1996.

**X. Delaunay,** « Compression d'images satellite par post-transformées dans le domaine des ondelettes », thèse de Doctorat en Télécommunications, Laboratoire des Télécommunications Spatiales et Aéronautiques - TéSA Institut National Polytechnique de Toulouse, France, 2008.