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 :

Mr ATMIMOU Omar

THEME :

Optimisation de la compression d'images par fractales en utilisant les algorithmes génétiques : Application aux images MSG

Devant le jury d'examen composé de :

AMEUR Soltane	Professeur à l'UMMTO	**Président**
LAHDIR Mourad	Maître de conférences A à l'UMMTO	**Rapporteur**
AMEUR Zohra née MAZOUZI	Professeur à l'UMMTO	**Examinatrice**
HADDAB Salah	Maître de conférences A à l'UMMTO	**Examinateur**
LAZRI Mourad	Maître de conférences B à l'UMMTO	**Examinateur**

Soutenu le : 12 /06 /2014

Remerciements

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

En premier lieu, je tiens à exprimer ma reconnaissance à monsieur LAHDIR Mourad mon directeur du mémoire, pour m'avoir guidé et encouragé tout au long de l'accomplissement de ce travail, et pour son entière confiance, ses conseils, critiques et sa patience.

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

J'exprime mes sincères remerciements à Madame MAZOUZI Zohra épouse AMEUR, Professeur à l'UMMTO et Directrice du laboratoire LAMPA, pour s'être portée volontaire pour faire partie des membres du jury.

J'exprime mes plus vifs remerciements à Monsieur HADDAB Salah, Maître de conférences à l'UMMTO, pour avoir accepté de participer à ce jury.

Mes remerciements vont aussi à Monsieur LAZRI Mourad, Maître de conférences à l'UMMTO, pour l'intérêt qu'il a porté à ce travail et d'avoir accepté de participer à ce Jury.

Mes remerciements vont également à tous les membres du laboratoire LAMPA pour leur soutien et suggestions. Je n'oublierai pas non plus les membres d'ADIAGTO, en particulier à Madame FEKID Fetta, Docteur NAHI Leila et Monsieur BELHOCINE Mehdi qui m'ont apporté une aide précieuse, Ainsi que mes amis(es) tels que MOUZARINE Ouahiba, DJENDER Samy, Samir DJANI, Samira GHEROUFELLA.

Je dédie ce mémoire à ma tendre grand-mère, mes chers parents, mes frères et sœurs qui n'ont jamais cessé de m'encourager tout au long de mon cursus, et de m'avoir apporté soutien et confort pondant les moments difficiles.

SOMMAIRE

CHAPITRE III : COMPRESSION D'IMAGES PAR DES FRACTALES OPTIMISÉE PAR LES ALGORITHMES GÉNÉTIQUES

CHAPITRE IV : RÉSULTATS ET DISCUSSIONS

INTRODUCTION

L'observation et l'étude des phénomènes atmosphériques exigent énormément de données. Ces données peuvent être acquises sur terre, ou bien dans l'espace grâce aux satellites météorologiques. Ces données sous formes d'images permettent d'obtenir une vue de plus en plus détaillée sur l'atmosphère. Dans notre travail, on s'intéresse 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. Il nous permet de suivre l'évolution détaillée des phénomènes atmosphériques grâce aux 12 canaux embarqués [1,2]. Chaque canal prend des images sur une fréquence bien spécifique du phénomène d'étude. Le rafraîchissement de ces images nous amène à un volume très important de données qui dépasse les 19 Goctets durant 24 heures. 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 transmission et de stockage. 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 [3]. 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.

Plusieurs techniques de compression ont été développées pour remédier aux contraintes de transmission et de stockage des images météorologiques. Ces techniques peuvent être classées en deux grandes classes, à savoir, les techniques sans pertes et les techniques avec pertes.

Les méthodes de compression sans perte sont basées sur le dénombrement statistique des données. Elles permettent de reconstruire l'image sans dégradation, mais avec un faible taux de compression (de l'ordre de 10%). Parmi ces méthodes, nous pouvons citer le codage de Huffman [4], le codage Run Length Encoding (RLE) [5] et le codage Lempel Ziv (LZ) [5].

Les méthodes de compression avec pertes qui se distinguent des méthodes sans pertes, introduisent une dégradation irréversible sur les pixels de l'image originale tout en augmentant les taux de compression. Parmi ces méthodes, nous pouvons citer la quantification vectorielle [5,6], le JPEG2000 [8,9,10] et les méthodes basées sur les fractales [11,12,13]. Ces dernières méthodes sont basées sur les travaux de Barnsley, résumés par Jacquin dans un algorithme établi en 1992 [14]. Cette technique repose sur le fait que toute image peut être considérée comme le résultat de la répétition des mêmes motifs à différentes échelles [15]. Notons néanmoins que malgré les performances des méthodes de compression par des fractales (bonne qualité de restitution des images avec des taux de compression élevés), elles restent lentes en terme de temps de codage comparées à d'autres techniques [16]. Afin de pallier cet inconvénient, nous avons utilisé les Algorithmes Génétiques (AGs). Ces derniers qui ont été développés par Holland, sont inspirés du mécanisme de sélection naturelle (génétique) [17,18]. Ils permettent de résoudre les problèmes d'optimisation, notamment quand l'espace de recherche est grand [19,20,21].

Dans le cadre de mon travail, nous présentons une nouvelle méthode de compression d'images basée sur des fractales en utilisant les algorithmes génétiques. L'objectif visé par notre approche est d'accélérer le temps de compression tout en réalisant un bon compromis entre le taux de compression et la qualité de restitution.

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

Dans le premier chapitre, nous introduisons la problématique de la quantité des données très importante fournie par le satellite Météosat Second Génération « MSG ». Le satellite MSG transmit à la terre des images prises dans l'intervalle allant de l'infrarouge jusqu'au domaine du visible. Nous introduisons aussi dans ce chapitre, des notions essentielles sur la compression, la structure générale des algorithmes utilisés ainsi que les paramètres permettant d'évaluer leurs performances

Dans le second chapitre, nous exposons les principales méthodes de compression d'images publiées dans la littérature. De plus, des méthodes de compression basées sur les fractales seront exposés et les résultats obtenus seront présentées.

Dans le troisième chapitre, nous présentons l'algorithme de compression que nous avons élaborée. Il est basé sur l'algorithme de Jacquin que nous avons modifié en incorporant les algorithmes génétiques pour générer les paramètres fractales à sauvegarder.

Les résultats obtenus par notre algorithme sur des images test et des images météorologiques du satellite MSG prises dans les canaux visible et infrarouge sont présentés dans le quatrième chapitre.

En conclusion, nous présentons les points importants de notre travail, en donnant les avantages et les inconvénients tout en proposant les perspectives nécessaires pour améliorer nos résultats.

Chapitre I :

Généralités

I.1- Préambule :

L’étude et l’observation des phénomènes atmosphériques par des satellites géostationnaires, ont énormément progressé au cours de ces dernières années. Les données fournies par ces satellites nous permettent une étude globale et détaillée sur ces phénomènes. L’étude et le suivie de l’évolution de ces phénomènes atmosphériques, nous impose une quantité de donnée importante à récolter par le satellite Météosat Seconde Génération «MSG». Le satellite MSG observe la terre au niveau de l’équateur via douze canaux de différentes longueurs d’ondes, collectés toute les quinze minutes. Dans ce chapitre, nous décrivons le satellite Météosat et les différents paramètres dont on a besoin pour répondre à la problématique du taux de données important des images Météosat.

I.2-Le satellite Météosat Seconde Génération (MSG) :

I.2.1- Prise de vues des images :

Le satellite Météosat Seconde Génération est équipé d’un radiomètre à balayage. C'est la rotation du satellite autour de son axe principal d'inertie qui est utilisée pour réaliser l'acquisition des images. Le satellite tourne à 100 tours par minute autour d'un axe parallèle à l'axe Nord-Sud de la

Terre (voir Fig. 1). Le télescope du radiomètre de Météosat vise la Terre par l'intermédiaire d'un miroir et balaie à chaque révolution du satellite une étroite bande de la surface de la Terre. L'angle de balayage correspondant à 18° est décrit en 30 ms. Pendant les 570 ms suivantes, le télescope vise l'espace et cette durée est mise à profit pour modifier l'orientation du miroir, de façon qu'au tour suivant, il balaie au sol la bande suivante, mais plus au nord [1,2].

L'acquisition complète de 3712 lignes pour chaque image à l'aide des 3 détecteurs que contiennent les 12 canaux exige environ 1250 révolutions du satellite, accomplies en 12 minutes 30 sec. Les 2 minutes 30 secs suivantes sont consacrées au retour du miroir à sa position initiale. La phase de non acquisition de données est mise à profit pour calibrer les détecteurs. L'envoi des données vers la terre se fait en temps réel.

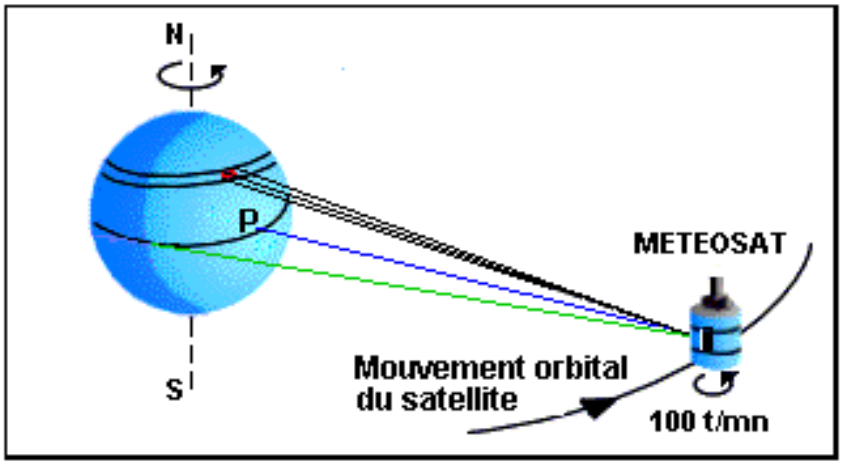


Fig. 1. Acquisition des images Météosat.

I.2.2- Caractéristiques du capteur SEVIRI :

L'imageur SEVIRI (Spinning Enhanced Visible & InfraRed Imager) est capable de fournir tous les quarts d'heure (au lieu d'1/2 heure avec Météosat) une image observée par le satellite, dans 12 bandes de fréquence différentes du spectre visible et infrarouge, soit 4 fois plus que Météosat. De plus, en réduisant de 30 à 15 minutes le rafraîchissement des données, le satellite MSG permet aux prévisionnistes de déceler plus facilement le déclenchement des phénomènes météorologiques à évolution rapide, comme les orages ou les tempêtes de neige. La résolution des canaux infrarouges passe de 5 km à 3 km, tandis que l'un des nouveaux canaux visibles fournit des images de 1 km de

résolution au lieu des 2,5 km de la première génération. Ce satellite emporte également une charge utile pour la collecte et la retransmission, quasiment en temps réel, d'observations recueillies par des stations automatiques au sol. Le tableau 1 décrit le domaine d'application de chaque image prise par le satellite MSG [22,23].

Le choix des bandes spectrales résulte de l'objectif de la mission Météosat. Le satellite Météosat de Seconde Génération a donc été équipé d'un capteur à douze canaux opérant dans les domaines du visible, de l'infrarouge moyen et de l'infrarouge thermique. Une scène SEVIRI comporte donc douze fichiers-images.

La figure 2 illustre les bandes spectrales du capteur SEVIRI en fonction de la transmission atmosphérique. Dans le domaine de l'infrarouge moyen (5,7 à 7,1 mm), l'atmosphère est opaque, en raison de l'absorption des rayonnements de ce domaine de longueur d'onde par les molécules d'eau présentes. Les données acquises dans les canaux vapeur d'eau WV donnent des informations sur la teneur en eau de l'atmosphère.

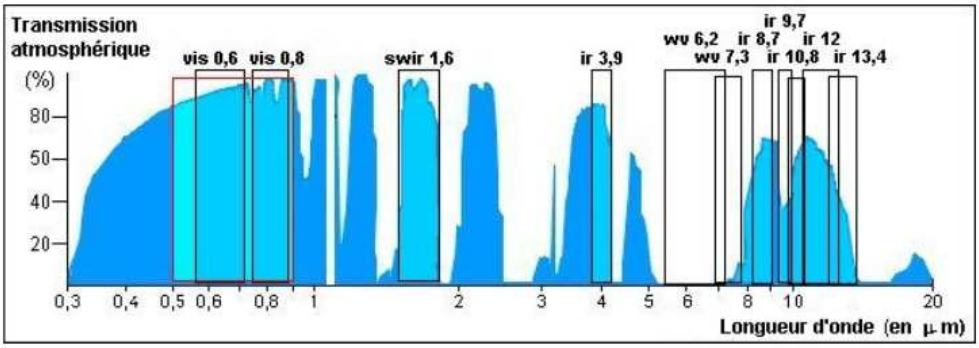


Fig. 2. Les bandes spectrales du capteur SEVIRI.

Canal	Utilisation
HRV, 0.6 et 0.8 µm	Détection, identification et évolution des nuages, observation des aérosols, suivis de la végétation.
1.6 µm	Différenciation entre neige et nuage, nuages de glace et d'eau liquide, information sur les aérosols.
3.9 µm	Détection des nuages bas de nuit, des feux de jour.
6.2 et 7.3 µm	Vapeur d'eau de la moyenne et haute troposphère, suivi de la dynamique atmosphérique, hauteur des nuages semi-transparents.
8.7 µm	Informations quantitatives sur les cirrus fins, distinction entre les nuages de glace et d'eau liquide.
9.7 µm	Radiance de l'ozone pour assimilation en prévision numérique, évolution du champ total d'ozone.
10.8 et 12 µm	Mesure de la température de surface de la terre et de la mer, détection des cirrus et déduction des quantités d'eau précipitable au-dessus de la mer
13.4 µm	Amélioration de la détermination du facteur de transmission des cirrus, information sur la température de la basse troposphère dépourvue de nuages pour les évaluations d'instabilité.

Tableau .1. Domaine d'application des images du satellite MSG.

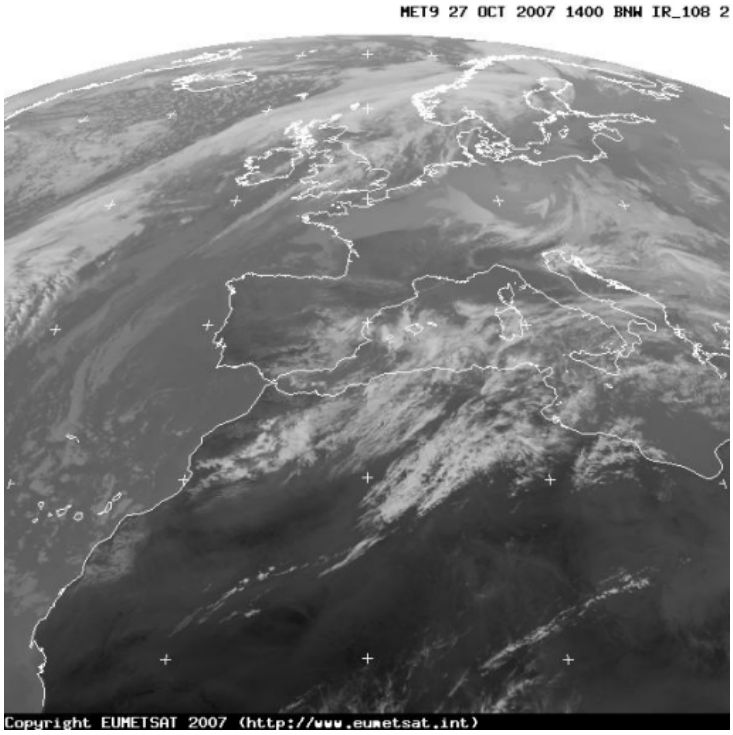
I.2.3-Images Météosat Seconde Génération :

Le satellite MSG a pour mission de transmettre vers la terre en temps réel une scène enregistrée dans 12 canaux du visible, de l'infrarouge moyen et de l'infrarouge thermique toutes les 15 minutes. Il permet d'avoir une vue de plus en plus détaillée sur l'atmosphère de l'Europe et d'Afrique. Les images fournies par le satellite MSG sont de bonne qualité. Ce sont des images en niveaux de gris codées sur 8 bits. Le tableau 2 décris les propriétés de ces images. Il affiche le nombre de pixels par ligne et le nombre de lignes par image, le champ total d'observation ainsi que la résolution spatiale au point sous-satellite [1].

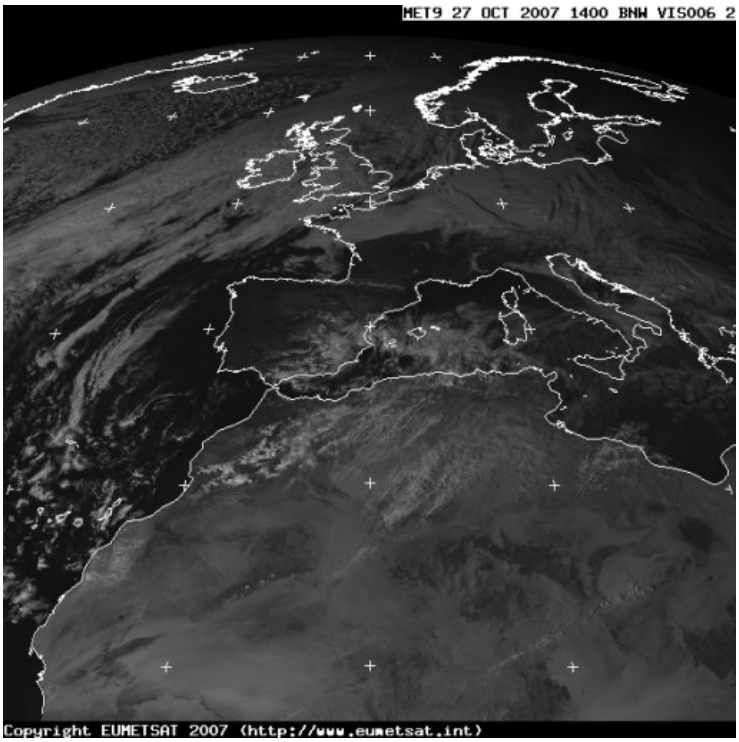
paramètres	Bandes spectrales (longueur d'onde exprimée en µm)											
nom du canal	HRV	VIS 0,6	VIS 0,8	SWIR	IR 3,9	WV 6,2	WV 7,3	IR 8,7	IR 9,7	IR 10,8	IR 12	IR 13,4
domaine de longueur d'onde	VIS 0,5-0,9	VIS 0,6-0,7	VIS 0,7-0,9	IR 1,5-1,8	IR 3,5-4,4	IR 5,3-7,1	IR 6,8-7,9	IR 8,3-9,1	IR 9,4-9,9	IR 9,8-11,8	IR 11-13	IR 12,4-14
résolution spatiale au point sous-satellite	1 km	3km	3km	3km	3km	3km	3km	3km	3km	3km	3km	3km
nombre de lignes par image	11136	3712	3712	3712	3712	3712	3712	3712	3712	3712	3712	3712
nombre de pixels par ligne	5568	3712	3712	3712	3712	3712	3712	3712	3712	3712	3712	3712
champ total d'observation	11200 km	11200 km	11200 km	11200 km	11200 km	11200 km	11200 km	1120 0 km	11200 km	11200 km	11200 km	11200 km

Tableau .2. Les principales caractéristiques des images MSG.

La figure 3 illustre les images prises par le satellite MSG dans le domaine de l'infrarouge (canal 10.8) et dans le domaine du visible (canal 0.6), le 27octobre 2007 à 14 h.



a)



b)

Fig. 3. Exemples d'images du satellite MSG.

a) Image infrarouge du canal 10.8 b) Image du visible du canal 0.6

I.2.4-La taille des images MSG :

Pour connaître la taille (en octets) d'une image, il est nécessaire de compter le nombre de pixels que contient l'image, cela revient à calculer la hauteur de celle-ci que multiplie sa largeur. La taille d'un fichier image Ti en octet est donné par l'équation suivante :

$$Ti\ (Octet) = nombre\ de\ lignes\ par\ image * nombre\ de\ pixels\ par\ lignes \quad (1.1)$$

A titre d'exemple, la taille des données fournies par le MSG durant un quart d'heur selon les données du tableau 2 :

- Pour le canal HRV : Ti (octet)= 11136 * 5568 octet = 62 005 248 octet, soit 59.1328 Mo.
- Pour les 11 canaux restant : Ti (octet) = (3712 * 3712) * 11 = 151 568 384 octet, soit 144.5469 Mo.

La taille totale des données récoltées est alors de :

- 203.6797 Mo durant 15 minutes.
- 814.7188 Mo durant une heure.
- 19.09 Go durant 24 heures.

I.2.5- Temps d'envoi d'un fichier :

La taille du fichier influe de manière proportionnelle sur la durée de transmission ou de réception sur des réseaux caractérisés par un faible débit. Plus le fichier est volumineux, plus la durée est importante. On suppose que le fichier est envoyé de façon continu en un seul message.

- La durée d'envoi T_{upload} (en secondes) pour transmettre un fichier de taille Ti bits (taille totale des données initiale), avec une vitesse d'envoi (upload) exprimé en bits/s est calculé en utilisant l'équation (1.2) [24] :

$$T_{upload} = \frac{Ti}{upload} \quad (s) \quad (1.2)$$

Sur un réseau de transmission avec un débit de 512 Kbits/s, la durée nécessaire pour l'envoi des données images de taille 59.1328 Mo est :

T_{upload} = (8*1024*1024*59.1328) / (512*1024) = 946.1248 s (environ **15 min 46 s** pour uploader un fichier image du canal HRV de **59.1328 Mo).**

I.3- Compression des données Météosat :

L'utilisation des données images Météosat avec leur taille réelle, nous confronte à des contraintes de transmission sur des réseaux à faible débit, et de stockage qui exige un espace important malgré l'ascension de la technologie utilisée. La puissance des processeurs augmente plus vite que les capacités de stockage, et énormément plus vite que la bande passante des réseaux, car cela demande d'énormes changements dans les infrastructures de télécommunications [3].

Ainsi, pour pallier ce manque, il est important de réduire la taille de ces données en exploitant la puissance des processeurs plutôt qu'en augmentant les capacités de stockage et de transmission des données. Pour cela, il existe plusieurs méthodes de réduction, tout en prenant en considération leurs avantages et leurs inconvénients. Chaque méthode de compression dépend du type de données à compresser et s'impose quel que soit l'évolution du matériel utilisé [25].

I.3.1- Possibilité de compression :

A cette question essentielle, il existe deux niveaux de réponses qui seront exploités conjointement dans les systèmes de compression des images Météosat :

- Redondance est l'existence de dépendances statistiques entre les échantillons d'une même image, conduit à une répétition de l'information sur des pixels ou bien des morceaux d'image voisins.
- Manque de pertinence de l'information liée aux propriétés de la perception humaine qui est imparfaite. Par exemple, la perception visuelle humaine possède une résolution limitée.

Ainsi, une méthode de compression tend à extraire l'information pertinente et à supprimer les redondances statistiques.

La compression des données du satellite MSG est plus que nécessaire afin de réduire la taille des données. Cette réduction nous permet, d'une part, de transmettre le fichier compressé en un temps très court sur un réseau caractérisé par un faible débit de transmission et d'autre part, gagné en place occupée dans les supports informatiques de stockages.

I.3.2- Structure générale d'un schéma de compression des images :

Le processus de compression d'une image nécessite trois étapes principales, à savoir, l'étape de transformation ou de décorrélation, l'étape de quantification et l'étape de codage entropique [5]. La structure générale d'un schéma de compression d'images fixes est donnée par la figure 4.

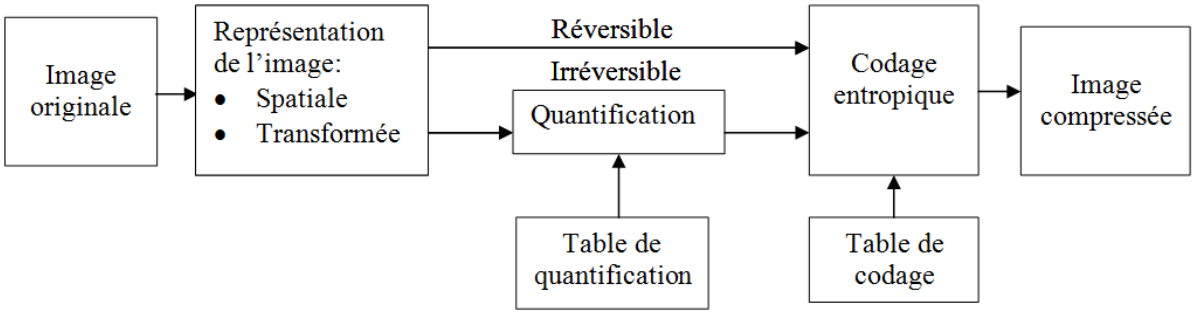


Fig.4. Schéma synoptique de compression réversible/irréversible des images fixes.

- **Etape de transformation ou de décorrélation :**

Cette étape a pour objectif de réduire la redondance d'informations contenues dans l'image, afin d'obtenir une représentation de l'image qui se prête à la quantification et au codage. De plus, elle nous offre la possibilité d'ajuster les erreurs de quantification selon la sensibilité du système visuel humain. Cette transformation est obtenue en exploitant les fortes corrélations spatiales entre les pixels de l'image. C'est une opération réversible qui a pour effet de redistribuer l'énergie de l'image originale en un nombre restreint de coefficients transformés [5].

- **Etape de quantification :**

Cette étape permet de réduire le nombre de bits nécessaires à la représentation des coefficients issus de l'étape de transformation. Les valeurs de ces coefficients sont approximées par un ensemble

fini d'éléments (scalaire ou vecteur) qui forme le dictionnaire de quantification. Ainsi, un élément quelconque d'entrée est remplacé par l'élément du dictionnaire le plus approprié (voir Fig. 2) [5].

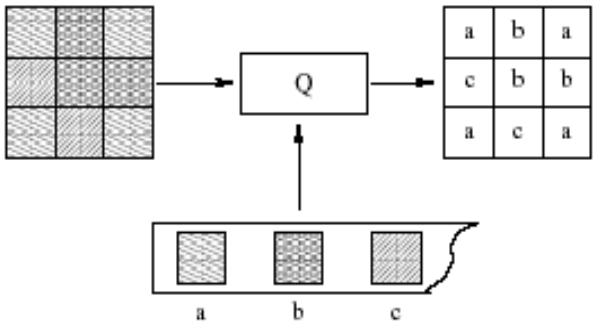


Fig. 5. Principe de la quantification.

- **Etape de codage entropique :**

C'est la phase finale du processus de compression, qui permet de générer le flot binaire à stocker ou à transmettre. Notons que le codage entropique est un codage sans pertes.

1.3.3-Classification des méthodes de compression d'images :

Les méthodes de compression d'images sont divisées en deux classes, à savoir, les méthodes sans pertes et celles avec pertes. Le choix d'une méthode de compression dépend des images à traiter et de leurs entropies. Rappelons que l'entropie est une grandeur qui caractérise la quantité d'information que contient une image. Pour une image en niveaux de gris dont les pixels sont codés sur *R* bits, l'entropie est donnée par l'équation (1.3).

$$H = -\sum_{i=0}^{i=N-1} P_i \log_2 P_i \qquad \text{(bits)}. \qquad (1.3)$$

où P_i est la probabilité de présence d'un niveau de gris i quelconque dans une image quantifié sur N niveaux.

A titre d'exemple, une image dont tous les pixels ont la même valeur (forte redondance) contient peu d'informations, son entropie est donc faible. Dans de tels cas, on utilise les méthodes sans pertes. A l'inverse, si la valeur de l'entropie est grande, on utilise les méthodes avec pertes.

Notons que dans certains cas, on utilise des méthodes hybrides qui consistent à coder différemment des zones déterminées de l'image en fonction de leurs intérêts. Il s'agit alors de coder les zones d'intérêts par des techniques sans pertes, et le reste de l'image par des techniques avec pertes.

Comme indiqué sur la figure 4, la seule étape qui différencie les méthodes de compression d'images avec pertes de celles sans pertes est l'étape de quantification qui n'intervient pas dans ces dernières. En effet, c'est le processus de quantification qui introduit les distorsions qui apparaissent sur les images décompressées [5].

- **Les méthodes sans pertes :**

Ces méthodes permettent après le processus de compression / décompression de reconstituer une image identique à l'image originale. Par exemple, pour les applications médicales : aucune erreur de diagnostic ne peut être tolérée. L'avantage de ce type de chaîne est d'avoir une image reconstruite identique, mais l'inconvénient réside dans le faible taux de compression que l'on peut atteindre. Les méthodes sans pertes peuvent être employées directement dans une chaîne de compression. Cependant, certaines d'entre elles sont bien souvent utilisées après la phase de quantification, lors de la transmission ou du stockage des index. Nous pouvons distinguer :

a. Les méthodes prédictives : celles-ci exploitent la redondance spatiale qui existe entre la valeur courante et les valeurs précédentes ou suivantes.
b. Les codeurs entropiques : ceux-ci tentent de s'approcher le plus possible de l'entropie de la séquence de valeurs à coder, en affectant un nombre de bits le plus faible possible aux valeurs les plus probables et vice versa. Le codage de Huffman et le codage arithmétique sont les principaux codeurs entropiques utilisés dans le domaine de la compression d'images.

- **Les méthodes avec pertes :**

Lors de la phase de quantification, des modifications sont apportées aux valeurs de l'image initiale. L'avantage de ce type d'approche est qu'il est possible d'atteindre des taux de compression importants, mais au détriment de la qualité de l'image reconstruite. Cependant, la majorité des applications grand public s'est orientée vers ce type de compression : appareil photo numérique,

images naturelles, transmission d'images sur les différents réseaux, stockage d'images, etc… Les méthodes de compression d'images avec pertes constituent la majorité des travaux de recherches.

I.4- Paramètres de performance des algorithmes de compression:

I.4.1-Taux de compression :

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

$$Rc = \frac{Nbre\ de\ bits\ de\ l'image\ originale}{Nbre\ de\ bits\ de\ l'image\ compressée} \tag{1.4}$$

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

$$Tc = (1 - \frac{1}{Rc}) \times 100 \tag{1.5}$$

I.4.2-Mesure de distorsion :

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

- L'erreur quadratique moyenne (MSE) :

$$MSE = \frac{1}{N} \sum_{i=1}^{N} (\hat{n}_i - n_i)^2 \tag{1.6}$$

Avec : n_i : le niveau de gris du i^{eme} point de l'image originale, $\hat{n}_i$: le niveau de gris du i^{eme} point de l'image transformée (reconstruite), N : le nombre total de points constituant chacune des images.

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

$$PSNR = 10 \log_{10} \frac{N_{dG\max}^2}{MSE} \tag{1.7}$$

Avec : $N_{dG\max}$: le niveau de gris maximum et l'exemple couramment utilisé en télévision numérique est $N_{dG\max}$ =255 donc :

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

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

$$I_{Diff} = 2(n_i - \hat{n}_i) + 128 \qquad (1.8)$$

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

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

Donc, pas de distorsion. Ce qui peut être vérifié sur le graphe de la figure 6, qui lie la distorsion au nombre de bits nécessaires au codage de l'image et qui est appelé « la fonction débit / distorsion » :

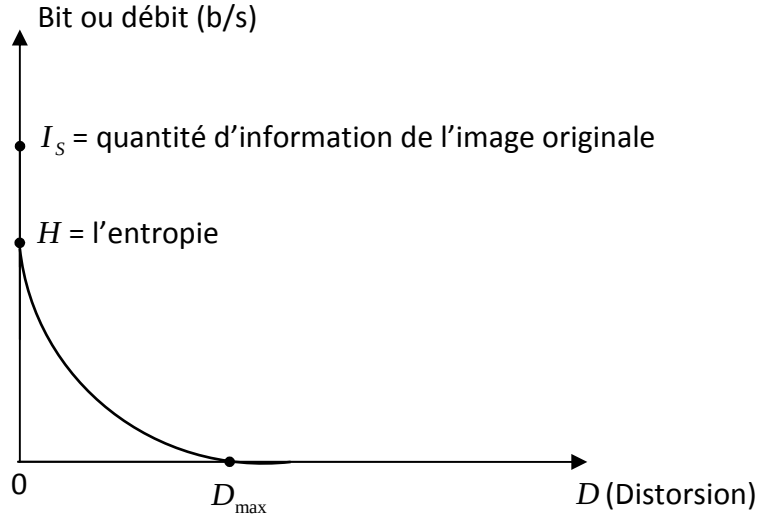


Fig.6. La fonction débit/distorsion.

De la figure 6, nous constatons que plus on réduit le nombre de bits par pixel, plus on introduit des distorsions ; donc D augmente. Par conséquent, on peut distinguer deux zones sur le graphe :

- La zone comprise entre I_S et H , $D = 0$; c'est-à-dire pas de distorsions,
- La zone comprise entre H et 0, $D \neq 0$; ce qui veut dire qu'il y a des distorsions.

I.4.3-Le temps d'exécution :

Le temps d'exécution d'une méthode de compression fait partie des paramètres des performances de celle-ci. La rapidité d'une méthode dépend du volume des données à compresser, de la complexité de l'algorithme de compression ainsi que de la puissance du processeur utilisé [5].

I.4.4-Le paramètre de performance :

Pour évaluer les performances des méthodes de compression d'images, un paramètre appelé paramètre de performance (Q) a été pris comme référence dans plusieurs articles de compression des images Météosat. Ce facteur (Q) est obtenu en effectuant le produit du taux de compression TC et du PSNR convertis en rapports sans dimensions [5]. Il est définit par l'équation (1.9).

$$Q = TC * e^{(PSNR/10)} \tag{1.9}$$

I.5- Discussions :

Dans ce premier chapitre, nous avons soulevé la problématique du taux de données très important récoltés par le satellite MSG. Ce taux de données nous confronte à la contrainte de stockage sur des supports informatiques et nous impose une durée de transmission très importante sur des réseaux à faible débit. Pour pallier à ces contraintes, nous proposons d'exploiter la puissance des microprocesseurs et les méthodes de compression. La compression permet donc de réduire le volume très important des images du satellite MSG avec le minimum de dégradation et le maximum de netteté possible.

Chapitre II :

Méthodes de compression d'images

II.1- Préambule :

Le satellite MSG fournit une quantité de données très importantes au file du temps sous forme d'images qui doivent être conservées, traitées et transmises. L'augmentation croissante et continue des capacités de stockage apporte une réponse partielle à ce problème, mais demeure insuffisante. Par ailleurs, la transmission nécessite des réseaux à débit très élevé. Nous présentons dans ce chapitre, les principales méthodes de compression d'images fixes qui ont été publiées dans la littérature, à savoir, les méthodes sans pertes et celles avec pertes. Nous donnerons pour chacune d'elle, le principe de fonctionnement ainsi que leurs avantages et leurs inconvénients. Nous décrirons, ensuite, quelques versions améliorées des méthodes hybrides utilisant les fractales.

II.2- Méthodes de compression d'images :

Comme précisé dans le chapitre un, l'objectif principal de la compression est de réduire la quantité de mémoire nécessaire pour le stockage d'une image et de réduire le temps de transmission de celle-ci. Cette compression peut soit conserver l'image intacte, on parle alors de compression sans perte, soit autoriser une dégradation de l'image pour diminuer encore l'empreinte mémoire, on parle de la compression avec perte.

II.2.1-Méthodes sans pertes (réversibles) :

Appelées aussi codeurs entropiques. Elles permettent de retrouver exactement les pixels de l'image originale. Le principe est d'associer à chaque pixel de l'image, un mot de code dont la longueur dépend de la probabilité d'apparition du niveau de gris correspondant. Pour obtenir un codage efficace, il suffit d'associer les mots de code les plus courts aux niveaux de gris ayant les plus fortes probabilités d'apparition, et inversement pour les niveaux

présentant une faible probabilité. Parmi les codeurs les plus utilisées, on peut citer le codage de Huffman [4,13], le codage Run Length Encoding (RLE) [5] et le codage Lempel Ziv (LZ) [5].

- **Codage de Huffman :**

Le codage de Huffman est certainement le plus utilisé en raison de sa gratuité, de sa simplicité et de son efficacité. Il est notamment utilisé pour le codage d'images binaires, telles que les fax, ainsi que dans toutes les modalités du standard JPEG. Le principe de ce codage est de créer des codes de longueurs variables [4]. Pour ce faire, les pixels sont d'abord triés et classés en fonction de leur fréquence (occurrence). Un graphe est alors construit de la manière suivante :

- A partir des deux pixels présentant la fréquence la plus faible, un nœud est créé. Il lui est affecté un poids égal à la somme des fréquences des deux symboles. Le nœud créé remplace désormais les deux symboles dans la suite du processus. A ces derniers sont affectés respectivement les chiffres binaires 0 pour le plus fréquent et 1 pour le plus rare ou l'inverse.
- La même démarche est reprise en considérant les deux symboles ou nœuds de poids le plus faible.
- Cette procédure est renouvelée tant qu'il reste plus d'un nœud libre.

- **Codeurs par longueur de plages :**

Le principe du codage RLE (Run Length Encoding) est de regrouper dans une image les pixels voisins ayant le même niveau de gris. Chaque groupement définit de façon unique un couple de valeurs $P_i(p, n)$, où p est le nombre de pixels voisins ayant un même niveau de gris n, et i le numéro de code dans la séquence [5]. L'exemple ci-dessous illustre un exemple de codage par RLE pour une séquence de niveaux de gris représentant une ligne d'une image.

….	19	13	13	13	13	109	131	131	162	162	162	162	162	88	…..

...; $P_i = (4, 13)$; $P_{i+1} = (1, 109)$; $P_{i+2} = (2, 131)$; $P_{i+3} = (5, 162)$; ...

L'efficacité de ce type de codeur dépend du nombre de transitions des niveaux de gris dans l'image à coder. Le codage est d'autant meilleur que le nombre de niveaux de gris est faible. Le taux de compression dépend de la complexité de l'image. Notons que ce type de codage est surtout utilisé dans des schémas intégrants les étapes de décorrélation et de quantification, tel que le standard JPEG.

- **Codeurs à base de dictionnaires :**

Le principal codeur à base de dictionnaire est le codeur « Lempel Ziv » (LZ). Il permet de coder des chaînes de longueurs variables par des mots de longueurs fixes. Ainsi, dans le cas d'une image, une suite de pixels est codée à l'aide d'un dictionnaire (ou d'une table de codage) construit au fur et à mesure de la lecture des données. Ce codeur est optimal pour des images de faible entropie telles que les images de synthèse, mais reste inefficace dans le cas des images réelles [5]. Son principal inconvénient réside dans le temps de calcul requis pour la comparaison des données à coder avec les mots du dictionnaire. Notons que LZ est à la base de nombreux algorithmes de compression de données tels que LZSS, LZ78 et LZW [5].

II.2.2-Méthodes avec pertes (irréversibles) :

La compression avec pertes, par opposition à la compression sans pertes, permet d'éliminer quelques informations pour avoir le meilleur taux de compression possible, tout en gardant un résultat qui soit le plus proche possible des données originales. Etant donné que ce type de compression supprime des informations contenues dans les données à compresser, on parle généralement de méthodes de compression irréversibles.

II.2.2.1- Méthodes spatiales :

Ces méthodes agissent directement sur les pixels de l'image originale. Elles exploitent la redondance entre un pixel et son voisinage, ou entre certaines régions de l'image. Dans les prochains paragraphes, nous décrirons brièvement quelques unes de ces méthodes, à savoir, les méthodes par quantification scalaire et vectorielle, les méthodes utilisant une décomposition en quadtree et les méthodes basées sur les fractales [11].

- **Méthodes utilisant une quantification scalaire (QS) :**

Parmi les quantificateurs scalaires, le plus simple est le quantificateur uniforme. Pour mettre en œuvre ce quantificateur, l'ensemble d'entrée représentant les pixels de l'image à compresser, est partitionné en *N* intervalles distincts de même longueur et chaque intervalle est représenté par sa valeur centrale. Chaque pixel de l'image est alors codé par la valeur centrale de l'intervalle auquel il appartient. Notons que la performance d'un quantificateur est mesurée à l'aide de l'erreur quadratique moyenne [5].

- **Méthodes utilisant une Quantification vectorielle (QV) :**

La quantification vectorielle (QV) est une généralisation de la quantification scalaire. Shannon a montré qu'il était toujours possible d'améliorer la compression de données en codant des vecteurs plutôt que des scalaires [26]. La figure 7 illustre le processus de codage par quantification vectorielle. L'image à coder est divisée en blocs disjoints qui couvrent toute l'image. Chaque bloc est comparé à un ensemble de blocs (mots de code) formant un dictionnaire. La compression consiste alors à calculer une distance entre le bloc à coder et le mot de code. Le codage s'effectue en gardant uniquement l'indice du mot de code le plus proche (partition). La distance utilisée est en général la distance euclidienne. A partir des mots de code correspondant aux indices reçus (partition), l'image est décompressée en utilisant le même dictionnaire.

De nombreuses méthodes ont été proposées pour la création du dictionnaire. On peut citer entre autres, la quantification vectorielle Algébrique QVA (ou en anglais LVQ Lattice Vector Quantization) [5], la quantification vectorielle à classification [6]. D'autres approches ont été présentées dans la littérature et conduisent à des résultats similaires [5].

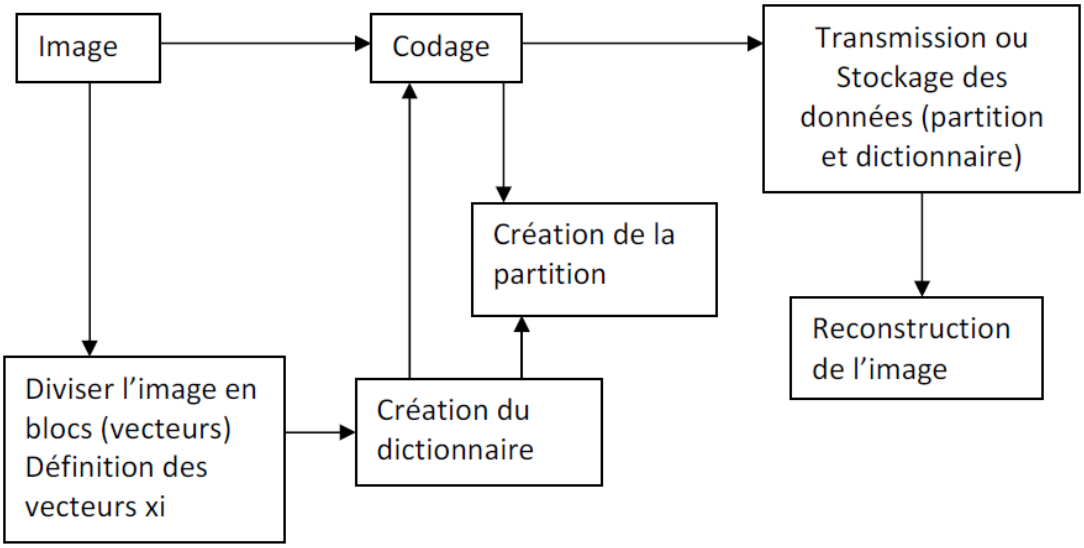
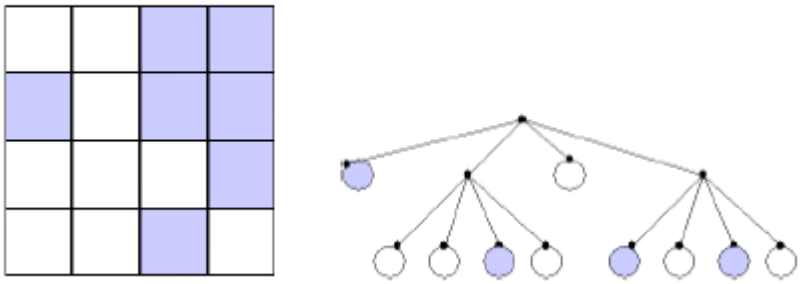


Fig. 7. Processus de codage d'images par quantification vectorielle.

- **Méthodes utilisant une décomposition en Quadtree :**

Ces méthodes sont basées sur une approche récursive. Le codage consiste à partitionner l'image suivant une structure arborescente [27,28]. Cette décomposition est réalisée comme suit :

Pour une image de taille $2^n \times 2^n$, le processus consiste à la diviser en quatre sous régions, de taille $2^{n-1} \times 2^{n-1}$. Les régions ne remplissant pas un critère d'homogénéité choisi au préalable sont divisées à leur tour en quatre sous régions. Le processus est réitéré jusqu'à ce que chacune des régions soit déclarée homogène (voir Fig. 8). Un code est alors affecté à chacune de ces régions.



a) b)

Fig. 8. Décomposition Quadtree.

a) Image originale b) Arbre de décomposition

II.2.2.2- Méthodes par transformation :

Ces méthodes restent celles qui sont les plus utilisées en compression d'images. En effet, elles permettent d'obtenir des taux de compression élevés tout en conservant une bonne qualité de l'image originale. Le principe de la compression est de décomposer les pixels fortement corrélés de l'image originale en un ensemble de coefficients spectraux partiellement décorrélés, dont l'énergie est concentrée en un nombre réduit de coefficients [29]. Les coefficients obtenus sont alors quantifiés et codés en vue de leurs transmissions ou de leurs stockages. La méthode JPEG reste la plus connue parmi les méthodes par transformation.

- **JPEG2000 :**

Ce standard a pour objectif d'offrir de nouvelles fonctionnalités permettant de répondre à une demande croissante, à savoir : Obtenir des performances de compression supérieures à son prédécesseur JPEG, notamment pour des débits très faibles, permettre d'organiser le fichier compressé de plusieurs manières, notamment en fonction de la résolution désirée ou de la qualité de reconstruction. Avoir un mode de compression performant sans perte, fournir la possibilité de coder des parties d'une image avec une qualité supérieure à d'autres parties. Le codage d'une image au format JPEG2000 est effectué en utilisant quatre étapes principales (voir Fig. 9) [8,9,10].

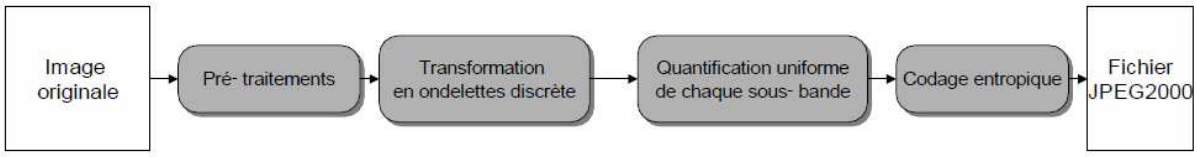


Fig. 9. Diagramme de la chaîne de codage de l'algorithme JPEG2000.

- **Prétraitements :**

Dans le contexte de la norme JPEG2000, chaque image est découpée en pavage. Le but de cette opération est de réduire la complexité de l'algorithme concernant les images de très grande taille, mais aussi de faciliter la navigation à l'intérieur de telles images. Ces

pavages sont de tailles carrées (64 x 64) ou (128 x 128). L'algorithme JPEG2000 code les coefficients DC de chaque bloc de manière différentielle, et enlève la valeur moyenne de l'image avant d'effectuer la transformation de l'image. Les coefficients basse-fréquence pourront ainsi être codés sur un nombre de bits moins importants. Cette étape comporte également une transformation de l'espace des couleurs RGB en l'espace luminance/chrominance. Cette transformation permet également de coder les couleurs avec un nombre de bits inferieur.

- **La transformation en ondelettes discrètes :**

Comme son nom l'indique, le but de cette transformation est de décomposer un signal [30]. On procède ainsi à une décorrélation de l'information qu'il contient. Les basses résolutions représentent la forme grossière du signal, tandis que les hautes résolutions encodent les détails du signal. Dans une approche du traitement du signal, les hautes résolutions représentent les hautes fréquences et les basses résolutions représentent les basses fréquences. Pour des signaux 2D comme les images, des propriétés topologiques (orientations, agencement du contenu) sont ainsi conservées après la transformation multirésolution. Le codeur qui est basé sur une telle transformation, peut ainsi prendre en compte la redondance spatiale et fréquentielle de l'image et ainsi être plus efficace. En compression d'images, l'étape qui permet la décomposition d'une image en sous-bandes est appelée analyse, elle s'effectue en utilisant deux filtres à réponse impulsionnelle finie, l'un passe-haut $H_A(n)$, l'autre passe-bas $L_A(n)$. La reconstruction de l'image, aussi appelée synthèse, nécessite également l'utilisation de deux filtres ($H_A(n)$ $L_A(n)$) et qui peut être construit à partir des filtres $H_A(n)$ et $L_A(n)$. Les étapes de décomposition (analyse) et de reconstruction (synthèse) d'une résolution à la résolution supérieure ou bien inferieure sont illustrées sur les figures 10 et 11.

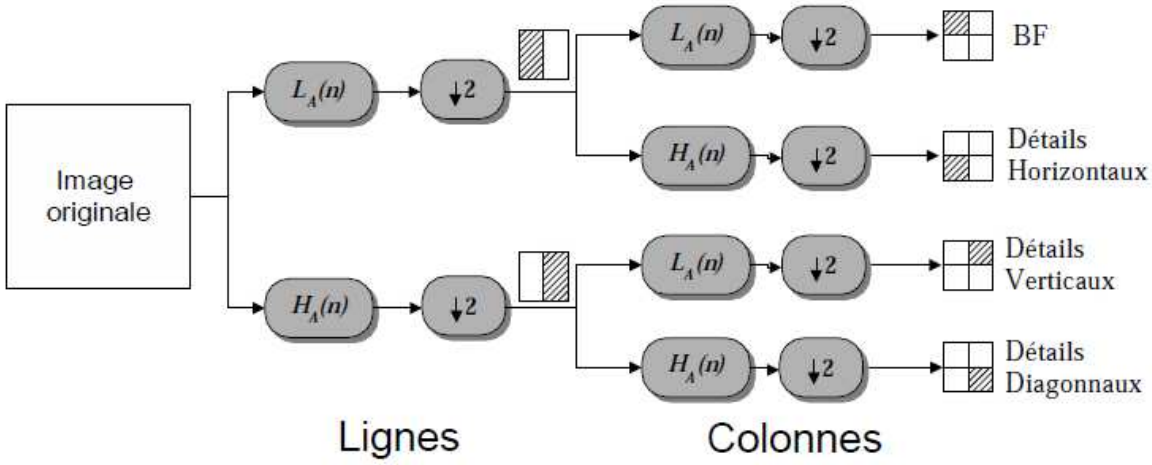


Fig. 10. Décomposition multirésolution d'une image.

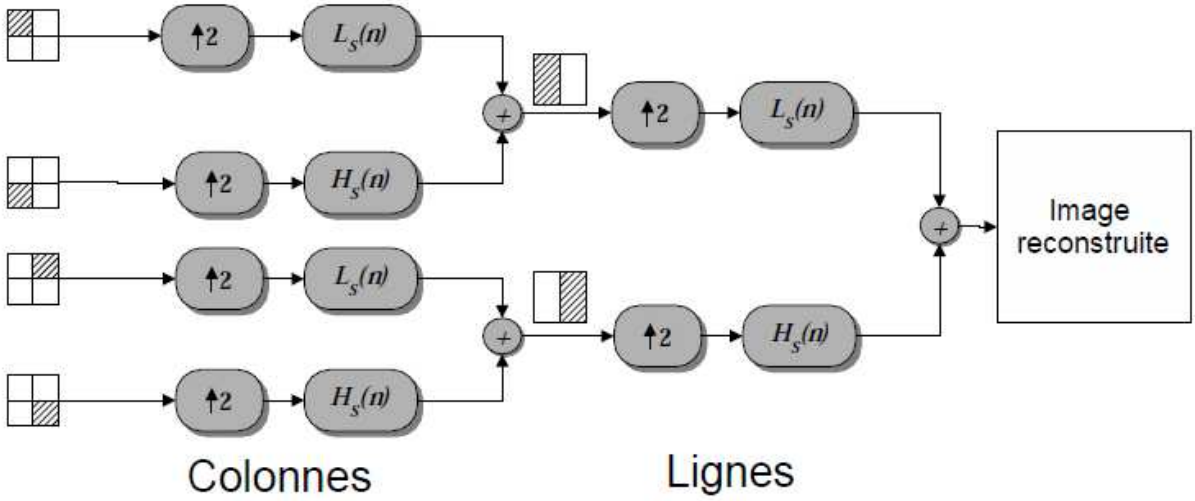


Fig. 11. Reconstruction d'une image.

- **Quantification uniforme de chaque sous-bande :**

Quand on code avec pertes, la précision sur les coefficients d'ondelettes obtenus à l'étape précédente est réduite par ce qu'on appelle une quantification scalaire uniforme. Disons qu'on ne conserve qu'un ordre de grandeur plus ou moins précis des coefficients. (Précisons que la norme permet d'affecter un pas de quantification différent à chaque sous-bande). Si l'on code sans perte, le pas de quantification est égal à 1.

- **Codage entropique :**

Il s'agit maintenant d'appliquer un codage sans perte aux données obtenues. On peut dire que JPEG2000 utilise ce qui se fait de mieux dans le genre. Les principaux types de codage entropique sont le codage de Huffman et le codage arithmétique

Notons au passage que le JPEG2000 demande 2 à 6 fois plus de calculs que le JPEG. Ceci peut paraître un inconvénient majeur, mais, étant donné la loi de Moore, la puissance de nos ordinateurs a été multipliée environ par 100 depuis la création du JPEG.

II.3- Méthodes de compression par des fractales :

Le principe de la compression fractale est basé sur une propriété intéressante des images naturelles : les auto-similarités (self-similarités). En effet, dans toute image, on peut trouver des motifs qui se répètent, avec différentes orientations, différentes échelles, et différents contrastes. On utilisera cette redondance à différentes échelles : les images fractales sont caractérisées par le fait que les motifs se répètent indéfiniment quand on zoom sur les frontières. Le fichier compressé comporte les transformations et les indices des blocs repérés [13].

La compression d'une image par des fractales repose sur une transformation, appelée transformation fractale. Cette dernière consiste à transformer l'image afin que son aspect visuel reste quasiment identique. Dans la méthode proposée par Jacquin en 1989 [11], l'image est divisée en N blocs $r_{n,}$ appelés blocs destination ou « cible ». Chacun de ces blocs est mis en correspondance avec un autre bloc transformé $\omega_n(d_{\alpha(n)})$ qui lui ressemble au sens d'une mesure d'erreur sur les niveaux de gris (voir Fig. 12). Le bloc $d_{\alpha(n)}$ est recherché dans une librairie composée de Q blocs appartenant à l'image. Ces Q blocs ne composent pas nécessairement une partition complète de l'image mais sont représentatifs de toute l'image. La transformée du bloc source $d_{\alpha(n)}$ par ω_n est $\hat{r}_n$, l'approximation du bloc destination r_n. Cette opération est appelée opération de collage.

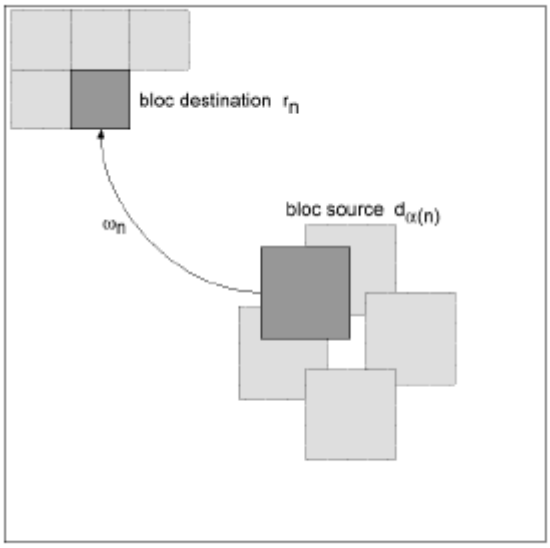


Fig. 12. Partitionnement de l'image.

Afin d'obtenir un taux de compression élevé, une bonne qualité de l'image restituée et un très court temps d'exécution lors du processus de compression et de décompression, on utilise les méthodes hybrides. Elles sont une combinaison de plusieurs techniques de compression dans le but de jumeler les différentes performances de celles-ci. Nous présentons quatre méthodes de compression d'images basées sur ce principe. Deux ont été élaborées au sein de notre laboratoire LAMPA (Laboratoire d'Analyse et de Modélisation des Phénomènes Aléatoires), les deux autres méthodes ont été publiées dans des revues ou communiquées dans des rencontres scientifiques internationales. Les résultats de ces quatre méthodes sur une base d'images tests et Météosat seront donnés dans le chapitre 4 afin de les comparer avec les résultats obtenus par notre approche.

II.3.1- Algorithme non itératif, basé sur les ondelettes biorthogonales et les fractales, pour la compression d'images satellitaires :

L'image à traiter est compressée puis décompressée en utilisant des ondelettes biorthogonales et une quantification vectorielle. L'image d'ondelettes qui en résulte est soustraite de l'image originale, pour donner une image résiduelle. Cette dernière est codée par des fractales [12]. La figure 12 illustre le schéma bloc de l'algorithme de compression. L'application de cette méthode sur l'image test Lena a permis d'obtenir un temps

d'exécution de 02 s, un PSNR de 32,90 dB pour un taux de compression de 93,75%. La méthode permet d'obtenir un facteur de performance de 2.0132 $x10^3$.

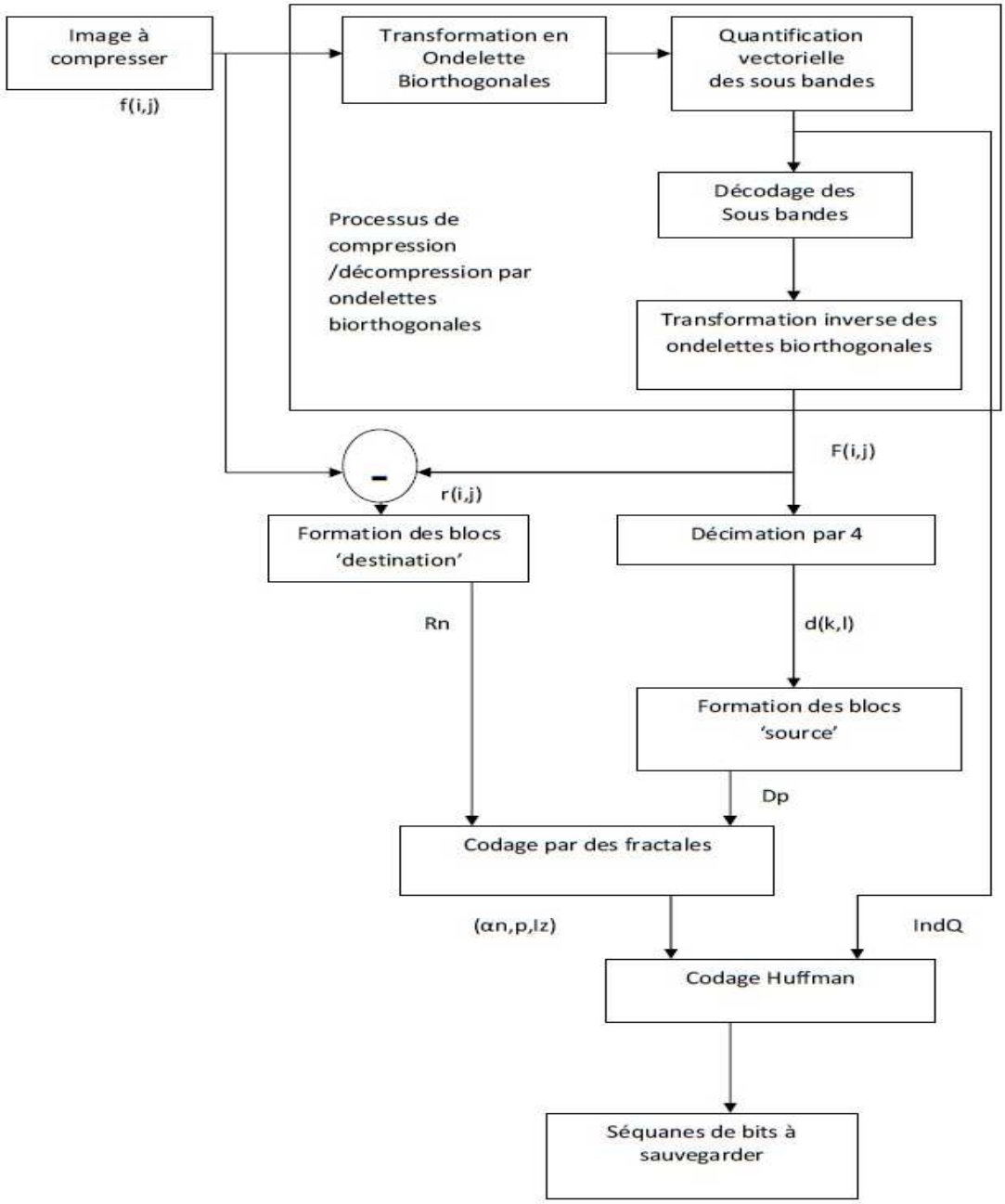


Fig. 12. Processus de compression.

II.3.2- Compression d'images fixes par des fractales en utilisant les algorithmes génétiques :

L'image à traiter est dans une première étape, compressée puis décompressée en utilisant des fractales. Cette étape consiste à partitionner l'image en blocs source et en blocs destination. Ces derniers sont classifiés à leur tour en blocs homogènes et blocs hétérogènes

selon la variance de leurs niveaux de gris. Les blocs hétérogènes sont codés par un système de fonctions itérées (IFS), en recherchant pour chacun d'eux le bloc source le plus similaire en utilisant un algorithme génétique. Les paramètres fractals générés ainsi que la moyenne des blocs homogènes sont sauvegardés. Une décompression est réalisée en recréant les partitions source sur une image de même taille que l'image originale et en leurs appliquant les transformations sauvegardées. Dans une seconde étape, l'image décompressée précédemment est soustraite de l'image originale pour former une image « différence » ou « résidu ». Cette dernière est codée selon la norme JPEG. Les paramètres fractals, la moyenne des blocs homogènes et les coefficients issus de JPEG représentent l'image compressée. Pour restituer l'image originale, il suffit de sommer l'image restituée de JPEG et celle du décodage fractal.
L'application de cette méthode sur l'image test Lena a permis d'obtenir un temps d'exécution de 06 s, un PSNR de 30,74 dB pour un taux de compression de 96,89%. La méthode permet d'obtenir un facteur de performance de 2.0956×10^3. [31]

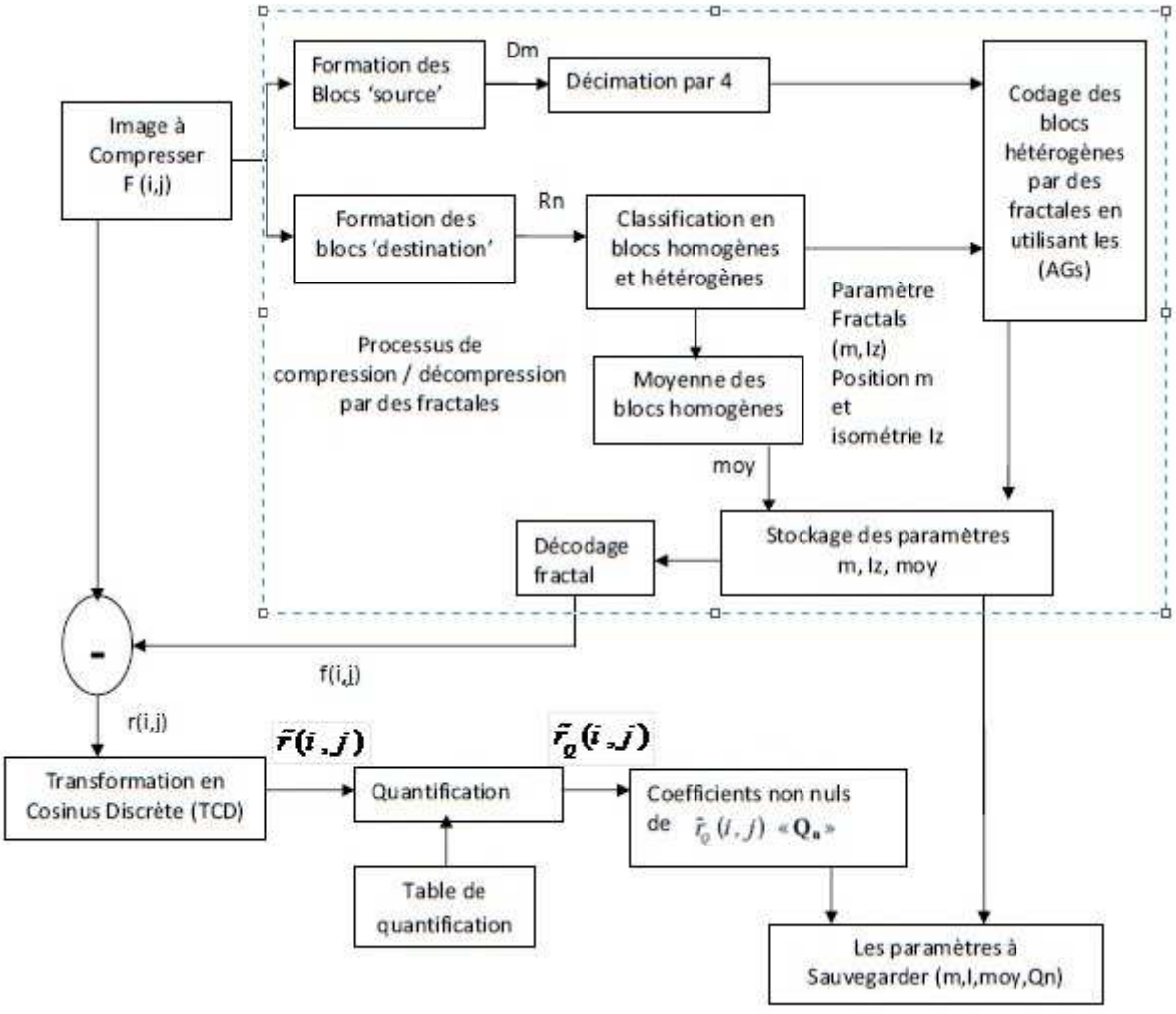


Fig. 13. Schéma du processus de compression.

II.3.3- Algorithme génétique appliqué à la compression d'image fractale :

La figure 14 illustre le schéma bloc de l'algorithme de codage d'image fractale. L'image à traiter est dans une première étape, compressée puis décompressée en utilisant des fractales. Cette étape consiste à partitionner l'image en cellules de domaine et en cellules gamme. Pour chaque cellule de domaine, une transformation correspondante et qui mieux à la cellule gamme est identifié. Les transformations sont généralement des transformations telles que le contraste et la luminosité. Le code fractal de l'image codée est une liste constituée d'information pour chaque emplacement des cellules [16]. L'application de cette méthode sur l'image test Lena a permis d'obtenir un temps d'exécution de 2370 s, un PSNR de 26.22 dB pour un taux de compression de 85.14 % .La méthode permet d'obtenir un facteur de performance de 1.1718×10^3.

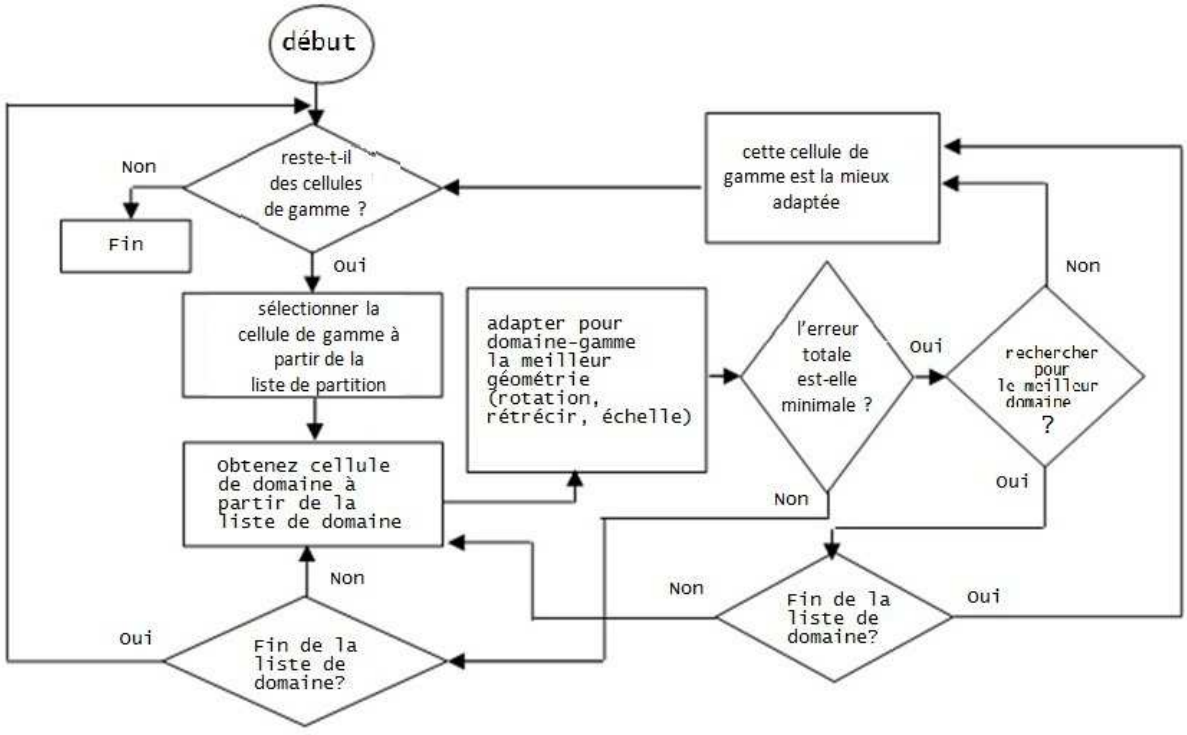


Fig. 14. Organigramme de codage d'image fractale.

II.3.4- Compression d'images fractales en estimant le plus proche voisin en utilisant la théorie du schéma :

L'algorithme tente de trouver un code pour l'image fractale sur l'ensemble des pixels adjacents de l'image. L'image à traiter est dans une première étape, compressée puis décompressée en utilisant des fractales. Cette étape consiste à partitionner l'image en blocs gamme et en blocs domaine [32]. L'objectif est de trouver à proximité d'une plage des blocs de gamme qui peut être remplacé par des blocs de domaine. De cette manière, au lieu d'utiliser les pixels de chaque bloc gamme, l'adresse du bloc domaine similaire est introduite, qui se traduisent par la compression de l'image. On définie un ensemble de 16 pixels adjacents (4 x 4) qui ont la forme carrée appelé bloc de domaine. Le bloc de domaine inclus une certaine similaire de bloc plus petite taille (2 x 2) appelé bloc de gamme (voir Fig. 15). Les blocs de l'image qui sont exactement adjacent à un bloc spécial sont appelés trame voisine de la première couche, Figure 16, les trames qui sont adjacentes avec distance d'un bloc, sont appelés trame voisine deuxième couche, des troisième et quatrième couches sont définies de la même façon. L'application de cette méthode sur l'image test Lena a permis d'obtenir un temps d'exécution de 70 s, un PSNR de 31.24 dB pour un taux de compression de 93.75 % . La méthode permet d'obtenir un facteur de performance de 2.1316×10^3.

Fig. 15. Image similaires avec différentes tailles.

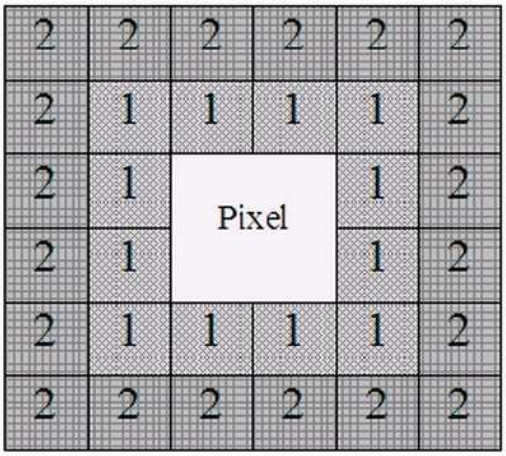


Fig. 16. Contiguïté des couches 1 et 2 pour le pixel.

II.4- Discussions :

La compression des données est appelée à prendre un rôle encore plus important, en raison du développement des besoins de l'utilisateur. Son importance est surtout due au décalage qui existe entre les possibilités matérielles des dispositifs que nous utilisons (débits sur internet, capacité des mémoires,...) et les besoins qu'expriment les utilisateurs (quantités d'information toujours plus importantes dans des délais toujours plus brefs). Quand ce décalage n'existe pas, ce qui est rare, la compression permet des économies. Les quelques méthodes de codage présentées (Huffman, RLE, LZ, JPEG2000) sont loin d'être exhaustives. Les méthodes de compression par des fractales citées dans ce chapitre font l'objet de comparaison de leurs résultats dans le chapitre quatre. Chaque méthode a son propre algorithme pour traiter l'image. Mais nous sommes loin d'avoir épuisé toutes les pistes de la compression par des fractales. Dans le chapitre trois, on s'intéressera à la méthode de Jacquin. Son principe de compression est basé sur les fractals. On a apporté des améliorations par les algorithmes génétiques dans le fonctionnement de la méthode, pour l'utiliser dans la compression des images du satellite MSG.

Chapitre III :

Compression d’images par des fractales optimisée par les algorithmes génétiques

III.1-Préambule :

Dans ce chapitre, nous présentons notre technique de compression choisie ainsi que les différentes étapes qui ont contribué à l’élaboration de l’algorithme de codage des images Météosat par les fractales. Notre choix s'est porté sur la méthode de Jacquin qui a été développée en 1989 [11]. Elle répond au besoin de compression des images satellitaire. Cette méthode permet d’atteindre un taux de compression très élevé avec un minimum de dégradation pour l’image reconstruite. Son principe de compression consiste à trouver des portions d’image qui se ressemble afin d’éliminer la redondance des groupes de pixel [33,34]. Cette méthode présente un inconvénient majeur qui est le grand nombre de comparaisons effectué, qui se répercute sur le temps de compression. Afin d’optimiser et d’accélérer la convergence de notre méthode de compression, nous l’avons associé aux algorithmes génétiques.

III.2- Méthode de compression :

III.2.1- Principe :

Le schéma de compression que nous avons adopté est représenté sur la figure 17. Dans un premier temps, l'image à compresser F(x,y) est partitionnée en blocs carrés selon la méthode de Jacquin afin de créer les blocs sources et les blocs cibles [14]. Pour chaque bloc source de taille 2T x 2T, on lui applique la transformation fractale W qui permet de l'approximer pour trouver le bloc cible de taille T x T le plus similaire tout en minimisant l'erreur de distorsion. L'algorithme génétique permet d'améliorer dans l'algorithme de Jacquin le temps de recherche et de trouver le bloc le plus similaire. Le code fractal à sauvegarder est composé de l'adresse du bloc source, de l'adresse du bloc cible qui lui correspond ainsi que de la transformation fractale W.

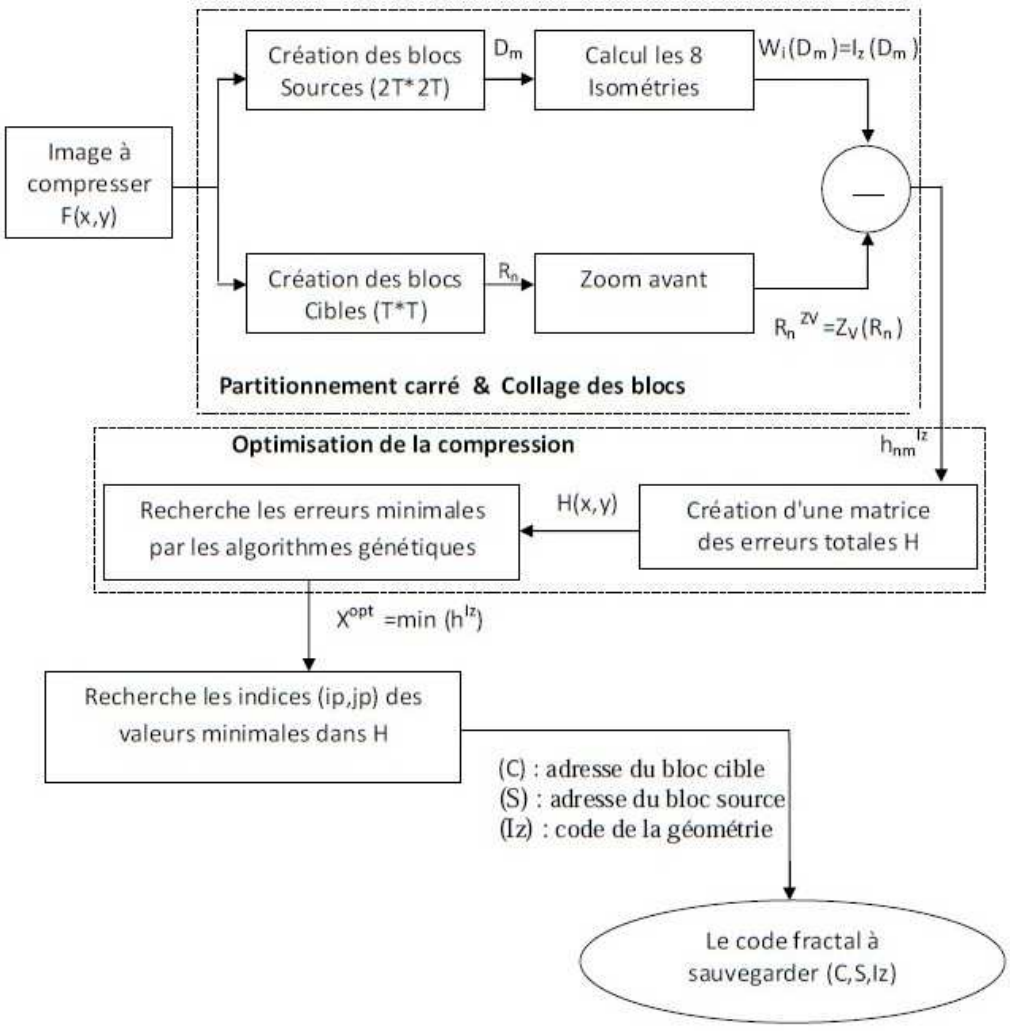


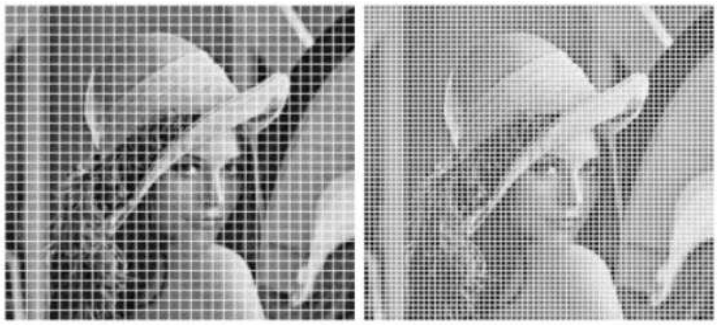
Fig. 17. Schéma bloc de compression.

III.2.2- Partitionnement de l'image :

L'image initiale F(x,y) est partitionnée en blocs carrés (de puissance de deux) pour former les blocs sources et les blocs cibles. Les blocs sources D_m son de taille 2T x 2T. Et les blocs cibles R_n son de taille T x T (voir Fig. 18), selon les deux équations :

$$F(x,y) = \bigcup_{n=1}^{N} R_n \tag{3.1}$$

$$F(x,y) = \bigcup_{m=1}^{M} D_m \tag{3.2}$$



a) b)

Fig. 18. Partitionnement de l'image Lena.
a) - Blocs sources de taille 2T x 2T.
b) - Blocs cibles de tailles T x T.

Un point essentiel dans les partitionnements des blocs est que le partitionnement cible doit être plus petit que le partitionnement source. En effet, dans le cas contraire, nous serions amenés à faire un agrandissement (et non une réduction, et l'image ne pourra pas alors converger) lors de la transposition des figures sources vers les figures cibles [35].

III.2.3-La transformation d'isométrie :

En pratique, la transformation w_i que nous avons utilisée permet de modifier la forme des blocs 'source'. Une telle opération est réalisée en faisant varier le facteur d'isométrie I_z. D'où :

$$w_i(D_m) = I_z(D_m) \tag{3.3}$$

La valeur de I_z peut prendre 8 valeurs différentes selon l'orientation considérée (voir le tableau 3).

Valeur de I_z	**Opérations d'isométrie**	**Application à un bloc D_m**
$I_1=1$	Identité	$I_1(D_m(i,j))=D_m(i,j)$
$I_2=2$	Rotation de 90°	$I_2\{D_m(i,j)\}=D_m(j,T-1-i)$
$I_3=3$	Rotation de 180°	$I_3\{D_m(i,j)\}=D_m(T-1-i,T-1-j)$
$I_4=4$	Rotation de 270°	$I_4\{D_m(i,j)\}=D_m(T-1-j,i)$
$I_5=5$	Réflexion par rapport à l'axe vertical	$I_5(D_m(i,j))=D_m(i,T-1-j)$
$I_6=6$	Réflexion par rapport à l'axe horizontal	$I_6\{D_m(i,j)\}=D_m(T-1-i,j)$
$I_7=7$	Réflexion par rapport à la première diagonale	$I_7\{D_m(i,j)\}=D_m(j,i)$
$I_8=8$	Réflexion par rapport à la seconde diagonale	$I_8\{D_m(i,j)\}=D_m(T-1-j,T-1-i)$

Tableau 3. Les huit isométries utilisées.

III .2.4-Transformation géométrique « Zoom avant » :

Cette transformation ramène le bloc cible R_n de taille T x T en un bloc R_n^{ZV} de taille 2T x 2T. Le bloc ainsi transformé R_n^{ZV} est obtenu par répétition des pixels du bloc cible R_n illustré par l'équation 3.4. cette transformation agit sur la valeur des pixels de position *(x ,y)* du bloc cible R_n.

$$R_n^{ZV}(2x,2y)=R_n^{ZV}(2x,2y+1)=R_n^{ZV}(2x+1,2y)=R_n^{ZV}(2x+1,2y+1)=R_n(x,y) \quad (3.4)$$

Dont x,y sont les coordonnées des pixels des blocs R_n et R_n^{ZV}.

Exemple :

Soit un bloc de taille T x T représenter par la matrice A (de taille 3 x 3), la répétition des éléments de A selon l'équation (3.4) qui fait un zoom avant forme la matrice I (de taille 6 x 6). (Voir Fig. 19)

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \qquad I = \begin{pmatrix} a_{11} & a_{11} & a_{12} & a_{12} & a_{13} & a_{13} \\ a_{11} & a_{11} & a_{12} & a_{12} & a_{13} & a_{13} \\ a_{21} & a_{21} & a_{22} & a_{22} & a_{23} & a_{23} \\ a_{21} & a_{21} & a_{22} & a_{22} & a_{23} & a_{23} \\ a_{31} & a_{31} & a_{32} & a_{32} & a_{33} & a_{33} \\ a_{31} & a_{31} & a_{32} & a_{32} & a_{33} & a_{33} \end{pmatrix}$$

Fig. 19. Zoom avant.

III.2.5-Collage d'un bloc source sur un bloc cible :

Le codage par des fractales consiste à déterminer une transformée affine et contractante w_i qui en l'appliquant aux blocs sources D_m, donne la meilleure approximation possible des blocs cibles R_n (voir Fig. 20).

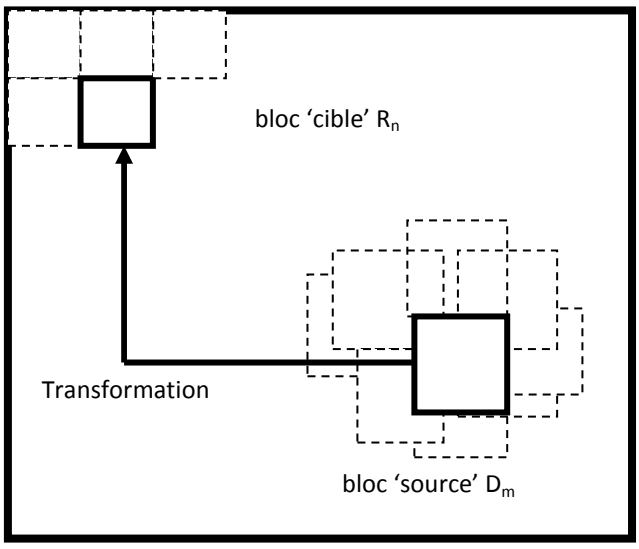


Fig. 20. Collage d'un bloc source sur un bloc cible.

III.2.6-La matrice des erreurs totales H :

Pour trouver le bloc D_m le plus proche du bloc R_n, nous cherchons pour chacun des blocs cibles R_n après un zoom avant, le bloc source D_m qui, après transformation w, permet de minimiser l'erreur quadratique moyenne h obtenue. Cette erreur est calculée par :

$$h^{Iz}(m,n) = \left(\frac{1}{p}\sum_{k=1}^{p}\sum_{l=1}^{p}(R_n^{ZV}(k,l) - I_Z(D_m(k,l)))\right)^2 \qquad (3.5)$$

I_z : représente l'une des huit isométries utilisées, avec $I_z = I_1 ; I_2 ; \ldots, I_8$.

p : représente la taille des blocs.

$R^{ZV}{}_n$: représente le bloc cible numéro n qui a subi un zoom avant.

D_m : représente le bloc source.

La matrice des erreurs totales H contient toutes les erreurs quadratiques moyennes obtenue entre $R^{ZV}{}_n$ et le bloc transformé D_m. Elle a le rôle d'organiser les différents résultats des erreurs dans un ordre bien déterminé (voir Fig. 21).

$$H = \begin{pmatrix} \text{Les erreurs } h_{nm}^1 \text{ Par rapport à l'identité} & \text{Les erreurs } h_{nm}^2 \text{ Par rapport à la rotation de 90°} & \text{Les erreurs } h_{nm}^3 \text{ Par rapport à la rotation de 180°} & \text{Les Erreurs } h_{nm}^4 \text{ Par rapport à la rotation de 270°} & \text{Les erreurs } h_{nm}^5 \text{ Par rapport à l'axe vertical} & \text{Les erreurs } h_{nm}^6 \text{ Par rapport à l'axe horizontal} & \text{Les erreurs } h_{nm}^7 \text{ Par rapport à la première diagonale} & \text{Les erreurs } h_{nm}^8 \text{ Par rapport à la seconde diagonale} \end{pmatrix}$$

Fig. 21. Matrice des erreurs totales H.

h_{nm}^{Iz} : représente l'erreur quadratique moyenne obtenue entre le bloc cible R_n après un zoom avant et le bloc source D_m qui a subi l'une des huit isométries.

III.3- Optimisation de la compression :

III.3.1-Les algorithmes génétiques :

La résolution d'un problème d'optimisation consiste à explorer un espace de recherche afin de maximiser (ou minimiser) une fonction donnée. L'espace de recherche et de la fonction à maximiser conduit à utiliser des méthodes de résolutions radicalement différentes. En première approximation, on peut dire qu'une méthode déterministe est adaptée à un espace de recherche petit et complexe et qu'un espace de recherche grand nécessite plutôt une méthode de recherche stochastique. L'usage d'un algorithme génétique est adapté à une exploration rapide et globale d'un espace de recherche de taille importante et est capable de fournir un résultat très satisfaisant [17, 18, 19,20].

III.3.2-Principe:

Les algorithmes génétiques adoptent le principe de l'évolution des populations biologiques à une population composée de N individus, codés en chaînes de bits, et initialisée aléatoirement au début de l'algorithme. La population évolue de la génération k à la génération k+1. Dans un premier temps, la population est reproduite par sélection où les bons individus se reproduisent mieux que les mauvais. Ensuite, on applique un croisement aux paires d'individus (les parents) d'une certaine proportion de la population (probabilité Pc, généralement autour de 0.6) pour en produire des nouveaux (les enfants). Un opérateur de mutation est également appliqué à une certaine proportion de la population (probabilité Pm, généralement très inférieure à Pc). Les nouveaux individus sont ensuite évalués et intégrés à la population de la génération suivante. Plusieurs critères d'arrêt de l'algorithme sont possibles : le nombre de générations peut être fixé a priori (temps constant) ou l'algorithme peut être arrêté lorsque la population n'évolue plus suffisamment rapidement.

Pour utiliser un algorithme génétique sur un problème d'optimisation, on doit donc disposer d'un principe de codage des individus, d'un mécanisme de génération de la population initiale et d'opérateurs permettant de diversifier la population au cours des générations et d'explorer l'espace de recherche [17]. La figure 22 donne un aperçu général sur le principe des algorithmes génétiques.

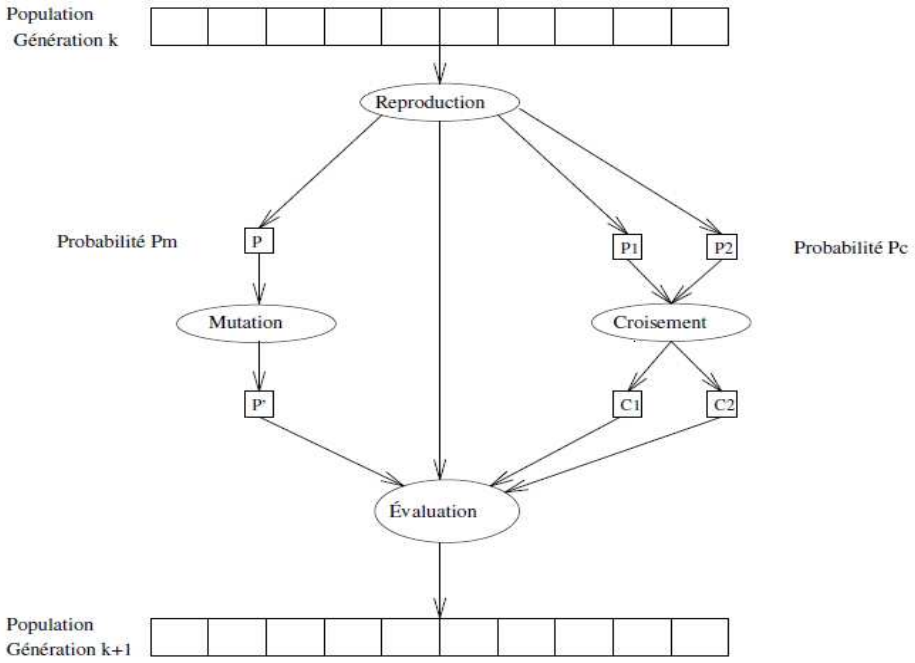


Fig. 22. Principe général des algorithmes génétiques.

III.3.3-Paramètres de l'algorithme génétique :

III.3.3.1-Codage des chromosomes :

Dans un algorithme génétique, un individu représente une solution du problème à traiter. Dans notre cas, une solution est une chaîne binaire de longueur L, représentant les paramètres fractals à savoir le numéro du bloc cible et le numéro d'isométrie. La valeur de L dépend de la taille de l'image et de la taille des blocs [19].

Exemple :

Pour une image de taille 256*256 et une taille des blocs sources de 4*4 on a :

- Pour chaque bloc cible on a 32 768 blocs sources ((256/4)* (256/4))*8
- Les numéros des blocs sources sont codés sur 16 bits.

Donc, un individu « X » est représenté sur 16 bits. X représente le numéro d'un bloc source parmi les 32 768 et une isométrie parmi les 8 existante.

III.3.3.2-Fonction d'évaluation (fitness) :

L'objectif de l'algorithme génétique est de trouver pour chaque bloc cible R_n , le bloc source D_n et l'isométrie I_z minimisant l'erreur quadratique moyenne h_i définie dans la relation (3.5). Cette dernière représente la fonction de fitness de l'algorithme génétique [19,20].

III.3.3.3-Population initiale :

La première étape de l'algorithme génétique est la construction de la population, c'est-à-dire, le choix d'un ensemble de solutions de départ, dans la matrice des erreurs totales H, que nous allons faire évoluer pour trouver la solution optimale du problème. La population initiale représente les individus de la première génération. Ses éléments « P » qui se trouvent dans la matrice des erreurs totales H, sont générés aléatoirement d'un ensemble dit espace de recherche P tel que :

$$P = [1, 2, ..., (M / T)] \tag{3.6}$$

M : nombre de lignes ou colonnes de l'image.

T : la taille des blocs cibles.

III.3.3.4-Sélection des individus :

La sélection des individus aptes à la reproduction est effectuée en utilisant la sélection par tournoi qui consiste à choisir aléatoirement un certain nombre d'individus et on sélectionne celui qui a la plus petite valeur de fitness [21].

III.3.3.5-Croisement :

Le croisement en un point est effectué pour tout couple d'individus sélectionnés avant. Le point de croisement est choisi aléatoirement.

III.3.3.6-Mutation :

La mutation consiste à changer ou à permuter des valeurs des gènes d'un chromosome. L'individu muté et le point de mutation sont choisis aléatoirement.

III.3.3.7-Critère d'arrêt :

Le critère d'arrêt est le nombre maximum de générations. C'est-à-dire que la solution est obtenue après un nombre de générations prédéfinies. Pour chaque bloc cible $R_{n,}$ on exécute l'algorithme génétique et on déduit l'isométrie I_z et le numéro de bloc source correspondants [20].

III.4-Formulation du code fractal :

III.4.1-Localisation des valeurs optimales :

Les valeurs optimales X^{opt} représentant l'erreur minimale entre le bloc source et le bloc cible sont localisés dans la matrice des erreurs totales H. Et nous utilisons les valeurs des indices « i_p , j_p » pour déduire le code fractal selon l'équation (3.7) :

$$(i_p , j_p) = \text{ind}\ (X^{opt}) \qquad (3.7)$$

i_p : représente l'indice ligne.

j_p : représente l'indice colonne.

X^{opt} : représente l'erreur minimale entre un bloc source et un bloc cible.

III.4.2-Extraction du code fractal :

Une fois que les indices « i_p , j_p » des valeurs optimales X^{opt} sont localisés dans H, nous les interprétons afin de formuler le code fractal. L'interprétation consiste à extraire la vraie adresse du bloc source S qui coïncide le mieux avec le bloc cible C, avec la valeur de l'isométrie I_z . Le tableau 4 résume la manière d'interpréter ces résultats selon le nombre totale M des blocs sources D_m , et la valeur de l'indice colonne j_p.

Localisation de j_p	**Valeur de I_z**	**Opérations d'isométrie**	**Numéro du bloc source S**
$1 \leq j_p \leq M$	$I_1=1$	Identité	$S = j_p$
$M+1 \leq j_p \leq 2M$	$I_2=2$	Rotation de 90°	$S = j_p - M$
$2M+1 \leq j_p \leq 3M$	$I_3=3$	Rotation de 180°	$S = j_p - 2M$
$3M+1 \leq j_p \leq 4M$	$I_4=4$	Rotation de 270°	$S = j_p - 3M$
$4M+1 \leq j_p \leq 5M$	$I_5=5$	Réflexion par rapport à l'axe vertical	$S = j_p - 4M$
$5M+1 \leq j_p \leq 6M$	$I_6=6$	Réflexion par rapport à l'axe horizontal	$S = j_p - 5M$
$6M+1 \leq j_p \leq 7M$	$I_7=7$	Réflexion par rapport à la première diagonale	$S = j_p - 6M$
$7M+1 \leq j_p \leq 8M$	$I_8=8$	Réflexion par rapport à la seconde diagonale	$S = j_p - 7M$

Tableau 4. Géométrie des blocs cibles.

Le code fractal est ainsi constitué de :

- C : l'adresse (ou numéro) du bloc cible R_n égal la valeur de i_p.
- S : l'adresse (ou numéro) du bloc source D_m selon la valeur de j_p (voir le tableau 4).
- I_z : le code de l'isométrie.

Exemple :

Soit M le nombre total des blocs sources (D_m) et M = 256 et N le nombre total des blocs cibles (R_n) avec N = 1024, formulons le code fractal selon la valeur de i_p et j_p.

a) $(i_p,j_p) = (1,96)$

$i_p = 1$. (c'est à dire le bloc cible R_n numéro 1).

$j_p = 96$. (c'est à dire le bloc source D_m numéro 96).

Iz = 1 (car $1 \leq 96 \leq 256$ donc le code de la géométrie est égal à 1).

donc le code fractal $(C,S,I_z) = (1,96,1)$.

b) $(i_p,j_p) = (3, 1888)$

$i_p = 3$. (c'est à dire le bloc cible R_n numéro 3).

$j_p = 96$ car $256 * 7 + 96 = 1888$. (c'est à dire le bloc source D_m numéro 96).

Iz = 8 (Réflexion par rapport à la seconde diagonale).

car $256*7+1 \leq 1888 \leq 256*8$.

donc le code fractal $(C,S,I_z) = (1,96,8)$.

III.5-Décompression :

Le schéma de décompression de notre méthode est représenté par la figure 22. A partir du code fractal sauvegardé, l'image est décompressée en effectuant les opérations inverses.

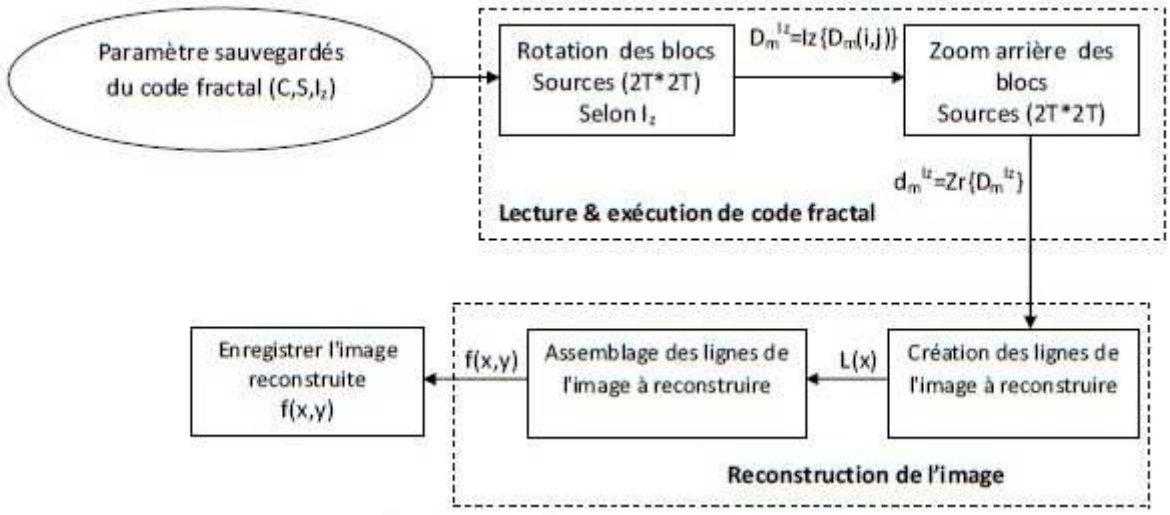


Fig. 23. Schéma synoptique de décompression d'images.

III.5.1-Rotation des blocs sources:

Pour chaque bloc source D_m (de taille 2T*2T), nous lui appliquons une transformation géométrique indiquée par I_z parmi les huit isométries définies dans le tableau 4 pour constituer le bloc D_m^{Iz}.

III.5.2-Zoom arrière des blocs sources :

Zoom arrière ou bien décimation consiste à réduire la taille des blocs sources D_m^{Iz} (qui sont de taille 2T * 2T) en blocs (de taille T * T) d_m^{Iz}. Cela est réalisé en remplaçant chaque sous bloc de taille 2×2 du bloc considéré par sa moyenne selon l'équation suivante :

$$d_m^{Iz}(i,j)=\frac{1}{4}\left[D_m^{Iz}(2i,2j)+D_m^{Iz}(2i,2j+1)+D_m^{Iz}(2i+1,2j)+D_m^{Iz}(2i+1,2j+1)\right] \quad (3.8)$$

$D_m(2i, 2j)$: est la valeur du pixel de la position (2i,2j) du bloc D_m^{Iz}. Notons que les sous blocs ne doivent pas se chevaucher.

III.5.3-Création des lignes de l'image:

La création des lignes de l'image à reconstruire L(x) se fait par un collage des blocs d_m^{Iz} dans leur position approprier indiqué par la valeur du C en utilisant l'équation suivante:

$$L(x) = \bigcup_{c=1}^{Tli} d_m^{Iz}(c) \tag{3.9}$$

L : représente la ligne de l'image à reconstruire.

x: représente le numéro de la ligne de l'image à reconstruire.

Tli : taille de la ligne de l'image originale.

C : représente l'emplacement où on doit coller le bloc d_m^{Iz}.

III.5.4-Formation de l'image décompressée :

L'image décompressée f(x,y) est reconstruite en utilisant l'équation suivante :

$$f(x, y) = \bigcup_{x=1}^{TCI} L(x) \tag{3.10}$$

L : représente la ligne de l'image à reconstruire.

x: représente le numéro de la ligne reconstruire.

TCI : taille de la colonne de l'image originale.

f(x,y) : représente l'image à reconstruire.

La figure 24-a représente une image test « Léna » inachevée. La figure24-b représente l'image reconstruite qui va être sauvegardée dans un fichier image.

a)

b)

Fig. 24. Image décompressée.

a) image inachevée.

b) image complète.

III.6- Discussions :

L'algorithme de Jacquin amélioré que nous avons conçu pour la compression des images satellitaires en associant les algorithmes génétiques pour son optimisation a été programmé sous matlab. Notre algorithme part du principe que toute image peut être considérée comme résultant de la répétition des mêmes motifs à différentes échelles. La recherche de ces motifs qui se répètent dans la même image nécessite de former des pavages sur l'image en utilisant différentes tailles de partitionnement. L'application de notre algorithme de compression sur un ensemble d'images tests et météorologiques seront présentées dans le chapitre quatre.

Chapitre IV :

Résultats et Discussions

IV.1-Préambule :

Pour évaluer les performances de notre méthode de compression, nous avons utilisé deux types d'images. A savoir, l'image test Lena et un ensemble d'images météorologiques prises dans les canaux visibles et infrarouge par le satellite MSG durant l'année 2007. Les résultats obtenus seront comparées aux méthodes basées sur les fractales décrites dans le chapitre deux.

IV.2-Matériels utilisé :

Pour implémenter notre méthode de compression, nous avons utilisé un Micro-ordinateur qui présente les caractéristiques systèmes suivantes :

- Système d'exploitation : Microsoft Windows 7 professionnel
- Type du système : Système d'exploitation 32 bits
- Processeur : Intel® Core (TM) i3-2312M CPU @2.10GHz, 2100MHz, 2coeur(s), 4processeur(s) logique(s).
- Mémoire physique (RAM) installée : 4.00 Go

La programmation des différentes procédures de notre application a été mise en œuvre en utilisant le langage Matlab R2013a.

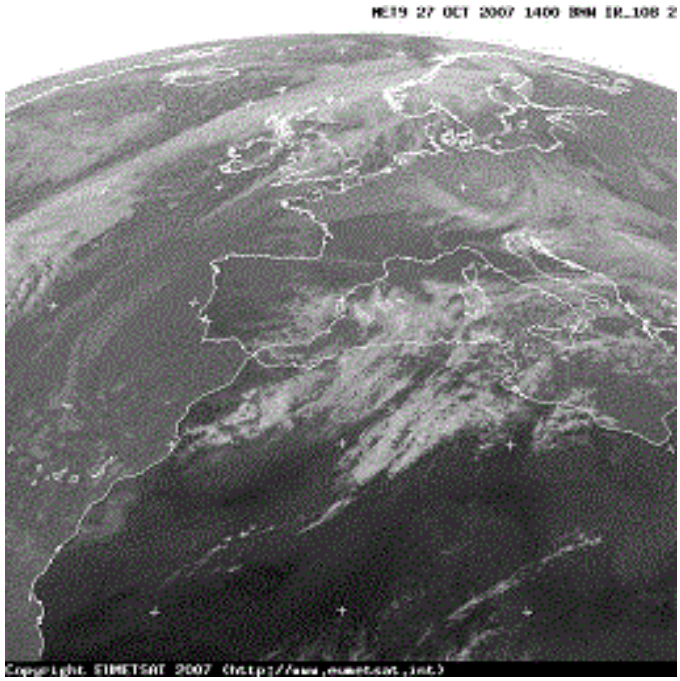
IV.3- Base de données utilisée:

Pour valider la méthode que nous avons mise en œuvre et montrer son efficacité, nous l'avons appliquée à une série d'images tests et une base d'images Météorologique :

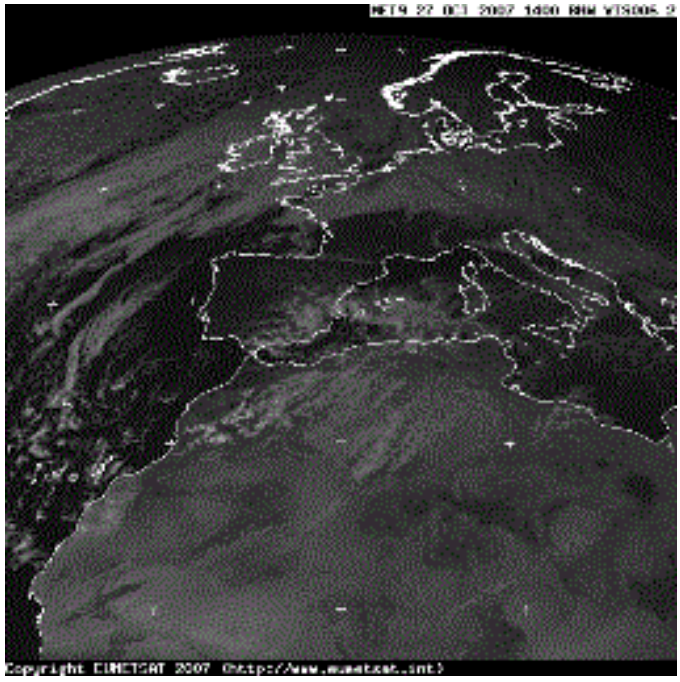
- Image test Lena qui représente un portrait de référence très utilisé pour l'évaluation des techniques de compression d'images. Elle est de résolution 256 * 256 codée sur 8 bits (voir Fig. 25).

Fig. 25. Image test 'Lena'.

- De plus, nous avons appliqué notre algorithme de compression sur un ensemble d'images Météorologiques du satellite MSG prises le 27 octobre 2007 à 14h dans les canaux visible 0.6 et le canal infrarouge 10.8. Ces images sont illustrées sur la figure 26.



a)



b)

Fig. 26. Images Météorologiques du satellite MSG.
a) Image du canal 10.8.
b) Image du canal 0.6.

IV.4-Application et Résultats :

IV.4.1-Partie optimisation :

L'application de l'algorithme de compression sur l'image Lena permet d'obtenir les résultats du tableau 5. Le temps de compression est calculé en variant la taille des blocs cibles et sources. Deux algorithmes d'optimisation ont été utilisés pour la recherche de l'erreur de l'étape d'apprentissage : la procédure min et l'algorithme génétique. Nous constatons que le temps d'exécution varie en fonction de l'algorithme d'optimisation et de la taille des blocs cibles-sources.

Recherche l'erreur la plus petite par	La taille des blocs (cible-source)	Temps (s)
Algorithme génétique	**2-4**	**410.62**
	4-8	**56.78**
	8-16	**12.21**
	16-32	**12.32**
Procédure min	**2-4**	**438.3**
	4-8	**64.78**
	8-16	**13.9**
	16-32	**13.4**

Tableau 5. Résultats obtenus en variant la taille des blocs cibles-sources.

La figure 27 illustre la variation du temps de compression/décompression en fonction de la taille des blocs. La plus grande valeur 438 s est obtenue en utilisant la procédure min avec la taille des blocs 2-4 (cible-source), tandis qu'avec les algorithmes génétiques, le temps maximal est de 410 s. Cette valeur diminue jusqu'à atteindre 12 s avec la taille 16-32.

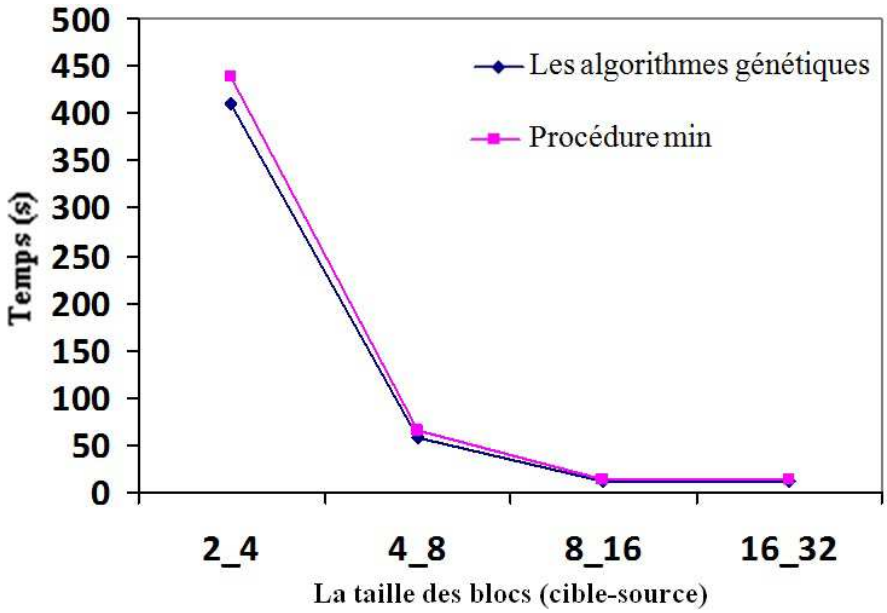


Fig. 27. Variation du temps d'exécution en fonction de la taille des blocs cibles-sources.

Dans le tableau 6, nous donnons le nombre total d'opérations N à effectuer pour la recherche des blocs cibles-sources les mieux adaptés.

Notons que le nombre d'opérations « N » est obtenu par l'équation suivante :

$$\mathbf{N = BS * BC * 8} \qquad (4.1)$$

Avec :

$$BC=\left(\frac{\text{taille de l'image}}{2*\text{taille des blocs}}\right)^2$$

$$BS = \left(\frac{\text{taille de l'image}}{\text{taille des blocs}}\right)^2$$

BS : nombre des blocs sources de taille 2T.

BC : nombre des blocs cibles de taille T.

Taille des blocs (cible - source)	BC (T x T)	BS (2T x 2T)	N
2-4	**16384**	**4096**	**536 870 912**
4-8	**4096**	**1024**	**33 554 432**
8-16	**1024**	**256**	**2 097 152**
16-32	**256**	**64**	**131 072**

Tableau 6. Nombre d'opérations effectuées selon la taille des blocs cibles-sources.

La figure 28 représente l'évolution de la 1er ligne de la matrice des erreurs totale H en fonction du nombre des blocs sources. Ce sont les erreurs entre le 1er bloc cible et les blocs sources qui ont subi les huit orientations. Nous constatons qu'il existe huit regroupements dans le graphe, dû aux huit isométries subies par les blocs sources. Chaque regroupement présente des valeurs minimales des erreurs. Avec les algorithmes génétiques, on élimine le risque de convergence dans un minimum local, c'est-à-dire les algorithmes génétiques trouvent le minimum globale, autrement dit, le bloc source le plus similaire au bloc cible.

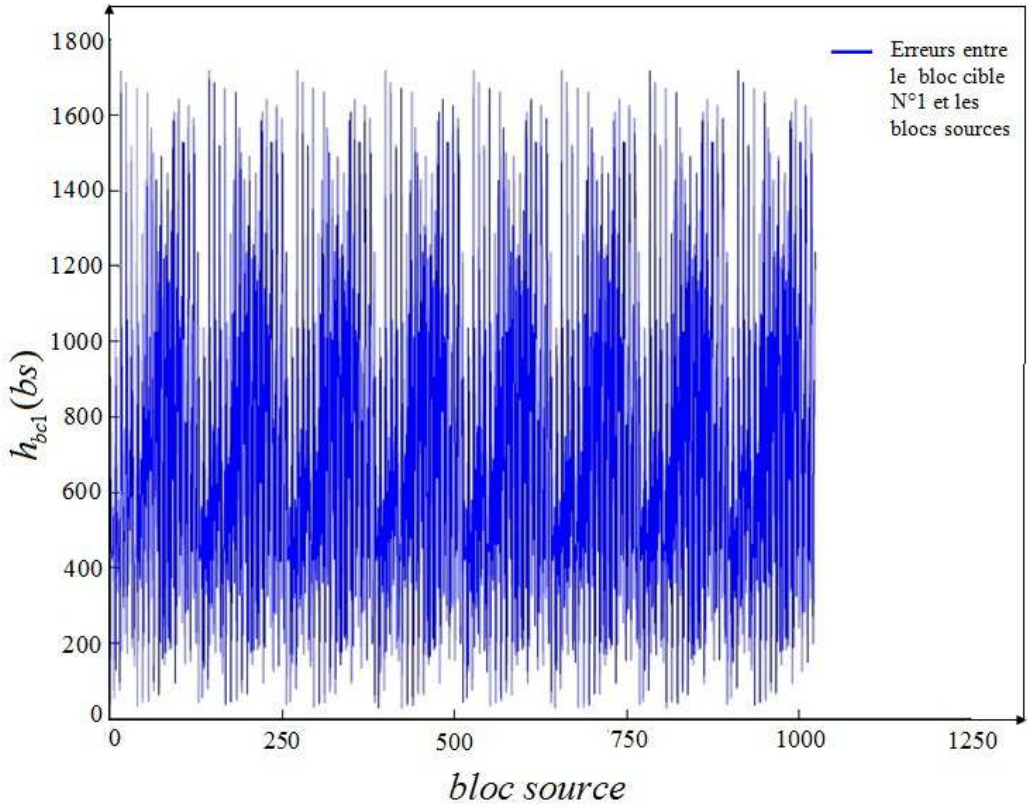


Fig. 28. Représentation graphique des erreurs de la 1er ligne de H.

IV.4.2- Variation de la taille des blocs (cible-source) :

Pour fixer la taille des blocs, nous avons varié la taille des blocs cibles-sources tout en choisissant le type de l'algorithme d'optimisation. Le tableau 7 donne les résultats obtenus. Nous constatons que le PSNR et le taux de compression ne varie pas en fonction de l'algorithme d'optimisation.

Recherche l'erreur la plus petite par	La taille des blocs (cible-source)	PSNR (dB)	TC (%)	Temps (s)
l'algorithme génétique	2-4	33.02	86.42	410.62
	4-8	22.02	81.64	56.78
	8-16	16.92	79.68	12.21
	16-32	1.41	78.1	12.32
Procédure min	2-4	33.02	86.42	438.3
	4-8	22.02	81.64	64.78
	8-16	16.92	79.68	13.9
	16-32	1.41	78.12	13.4

Tableau 7. Résultats du PSNR et taux de compression selon la taille des blocs.

La figure 29 représente la variation du PSNR en fonction de la taille des blocs à traiter. Plus nous augmentons la taille des blocs à traiter, plus le PSNR diminue. Sa valeur maximale est de 33.02 dB pour une taille des blocs 2-4 pixels. Plus la taille d'un bloc est petite, ce qui engendre une erreur petite l'or du remplacement d'un bloc par l'autre, ce que se traduit sur l'image reconstruite une dégradation minimale par rapport à l'image originale.

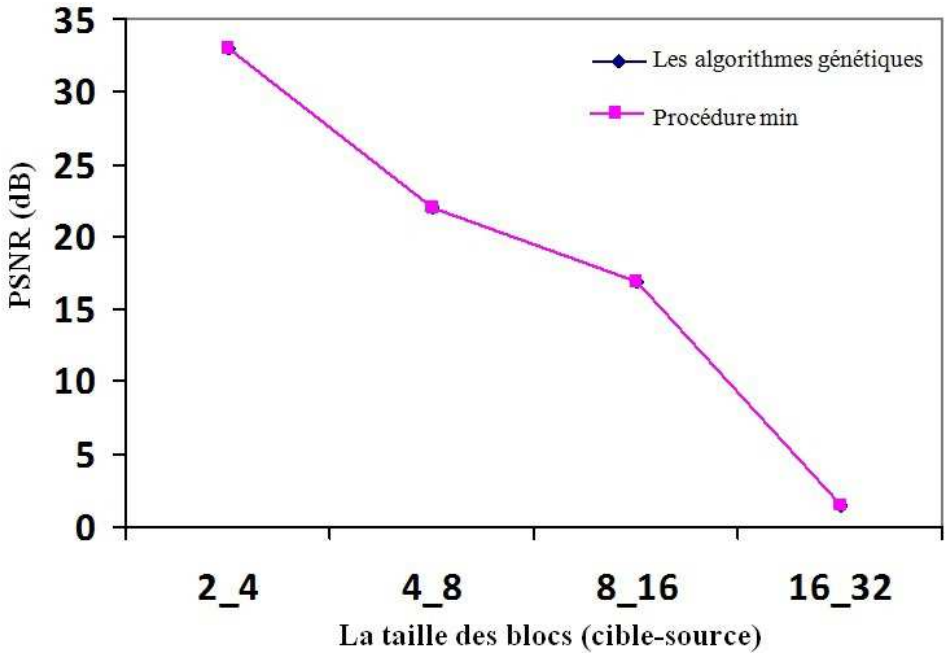


Fig. 29. Variation de PSNR en fonction de la taille des blocs.

Dans la figure 30, nous remarquons que le taux de compression est inversement proportionnel à la taille des blocs à traiter. Plus nous augmentons la taille de ces blocs, plus le taux de compression diminue. Le meilleur taux de compression 86.42% est obtenu en fixant la taille à 2-4. La plus faible valeur 78.10 % est obtenue en fixant la taille à 16-32. Plus la taille d'un bloc est petite, la probabilité de trouver un bloc qui lui ressemble (au pris d'une erreur) augmente. L'utilisation des petits blocs permet d'augmenter le taux de compression. C'est-a-dire un petit bloc va représenter autant de petit bloc lors de la reconstruction d'image avec une dégradation minimale par rapport a l'image originale.

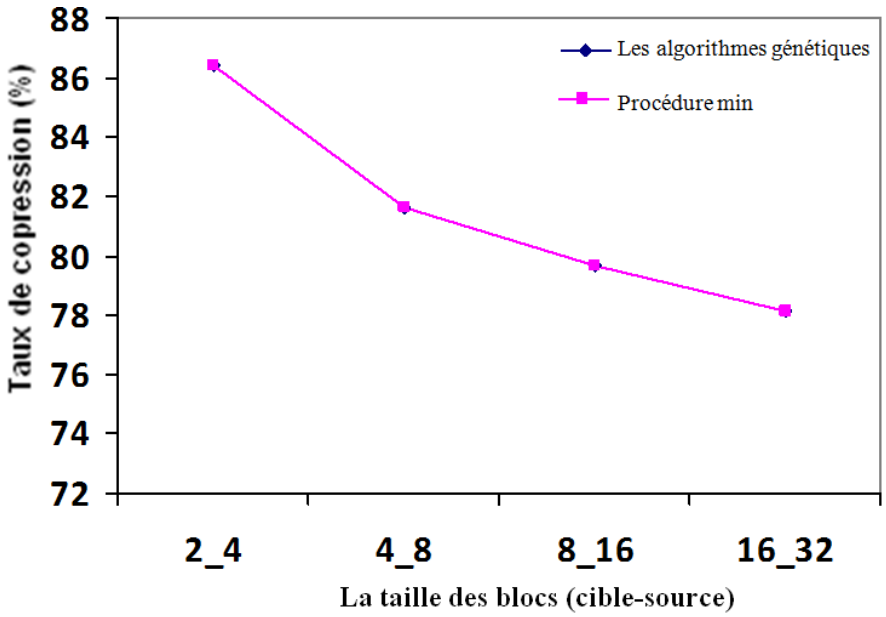


Fig. 30. Variation du taux de compression en fonction de la taille des blocs.

Le meilleur compromis possible entre le PSNR et le taux de compression est évalué en variant la taille des blocs. Dans la figure 31, nous illustrons la variation du PSNR en fonction du taux de compression (Tc). Le meilleur compromis est atteint pour un taux de compression de 86,42 % et un PSNR de 33.02 dB.

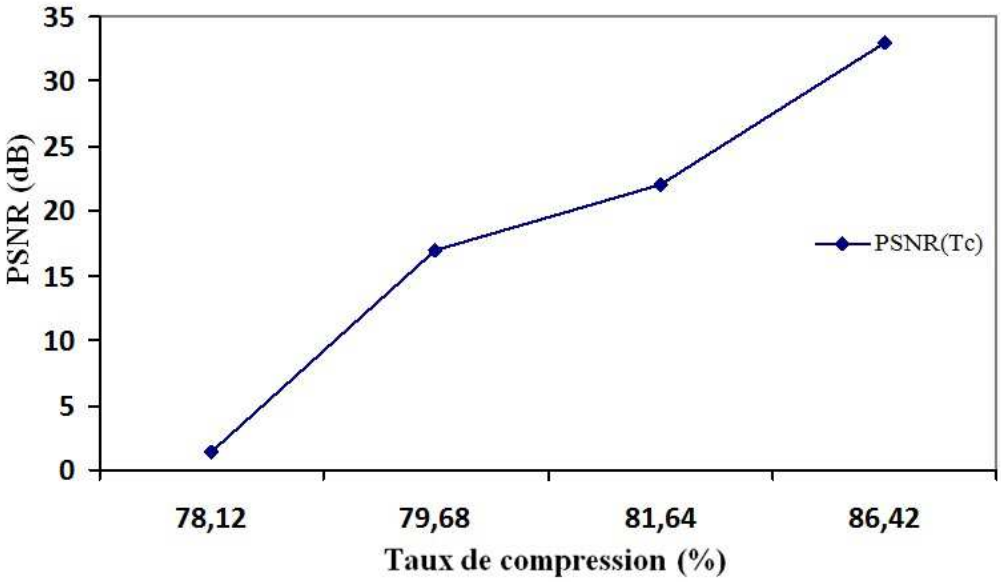
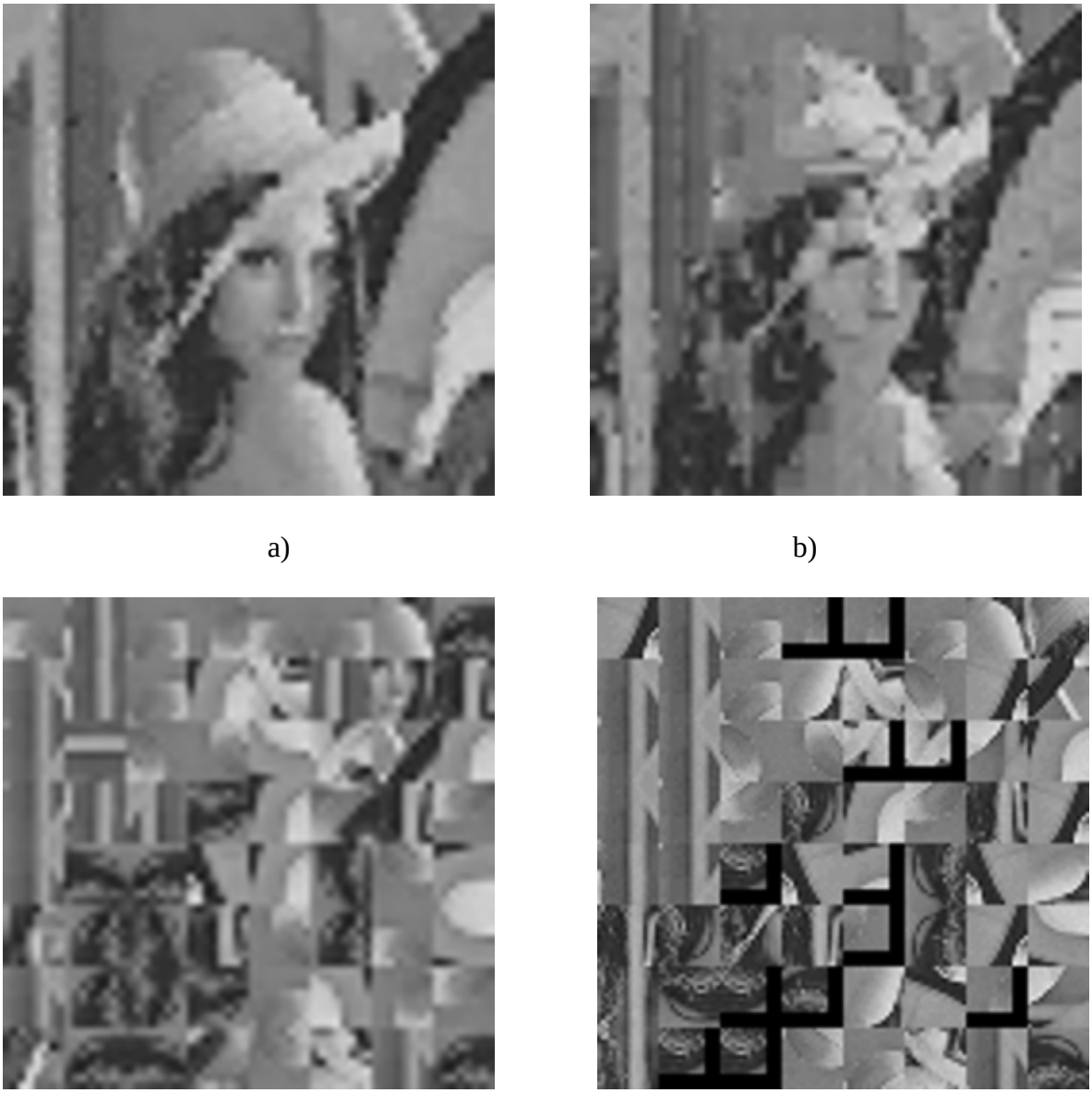


Fig. 31. Variation du PSNR en fonction du taux de compression Tc.

A partir de la figure 31, nous remarquons que le PSNR augmente en fonction du taux de compression.

La figure 32 représente les images Lena restituées avec différentes tailles des blocs. Nous remarquons que l'image obtenue en utilisant une taille de 2-4 pixels présente le meilleur aspect visuel. Par contre, en augmentant la taille des blocs, nous constatons une dégradation de l'image reconstruite. La figure 32-d représente la reconstitution de l'image Lena avec la taille des blocs 16-32. Elle est totalement dégradée et représente un mauvais aspect visuel.



a) b)

c) d)

Fig. 32. Les images test reconstruites selon la taille des blocs.

a) taille des blocs 2-4 b) taille des blocs 4-8
c) taille des blocs 8-16 d) taille des blocs 16-32

IV.5-Comparaison des résultats obtenus :

Le tableau 8 illustre le temps d'exécution des méthodes citées dans les chapitres précédents. Nous constatons que notre algorithme de compression permet d'obtenir un temps d'exécution comparable aux autres méthodes basées sur les fractales.

Méthode	Temps d'exécution en (s)
Méthode proposée	**410**
Algorithme non itératif, basé sur les ondelettes biorthogonales et les fractales, pour la compression d'images satellitaires. [12]	**02**
Algorithme génétique appliqué à la compression d'image fractale. [16]	**2370**
Compression d'images fractales en estimant le plus proche voisin en utilisant la théorie du schéma. [32]	**70**
Compression d'images fixes par des fractales en utilisant les algorithmes génétiques. [31]	**06**

Tableau 8. Temps d'exécution des méthodes basées sur les fractales.

A partir des résultats qui sont données dans le tableau 9, nous constatons que notre méthode présente un meilleur facteur de performance en obtenant un facteur de 2347.8.
La figure 33 donne les images reconstruites par les différentes méthodes représentées dans le tableau 9.

Méthode de compression	PSNR en (dB)	Le taux de compression en (%)	Facteur de performance Q
Méthode proposée	**33.02**	**86.42**	**2.3478×10^3**
Algorithme non itératif, basé sur les ondelettes biorthogonales et les fractales, pour la compression d'images satellitaires. [12]	**32,90**	**93,75**	**2.0132×10^3**
algorithme génétique appliqué à la compression d'image fractale. [16]	**26.22**	**85.14**	**1.1718×10^3**
Compression d'images fractales en estimant le plus proche voisin en utilisant la théorie du schéma. [32]	**31.24**	**93.75**	**2.1316×10^3**
Compression d'images fixes par des fractales en utilisant les algorithmes génétiques. [31]	**30.74**	**96.89**	**2.0956×10^3**

Tableau 9. Comparaison des méthodes.

a)

b)

c)

d)

Fig. 33. Images reconstruites.

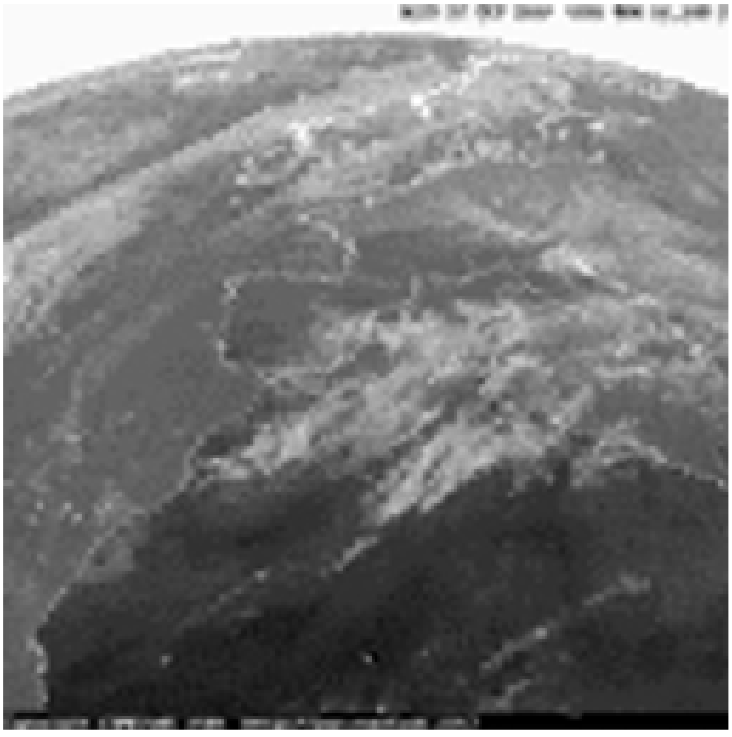
a) Algorithme non itératif, basé sur les ondelettes biorthogonales et les fractales, pour la compression d'images satellitaires.
b) algorithme génétique appliquée à la compression d'image fractale.
c) Compression d'images fractales en estimant le plus proche voisin en utilisant la théorie du schéma.
d) Compression d'images fixes par des fractales en utilisant les algorithmes génétiques.

IV.6- Application sur les images MSG :

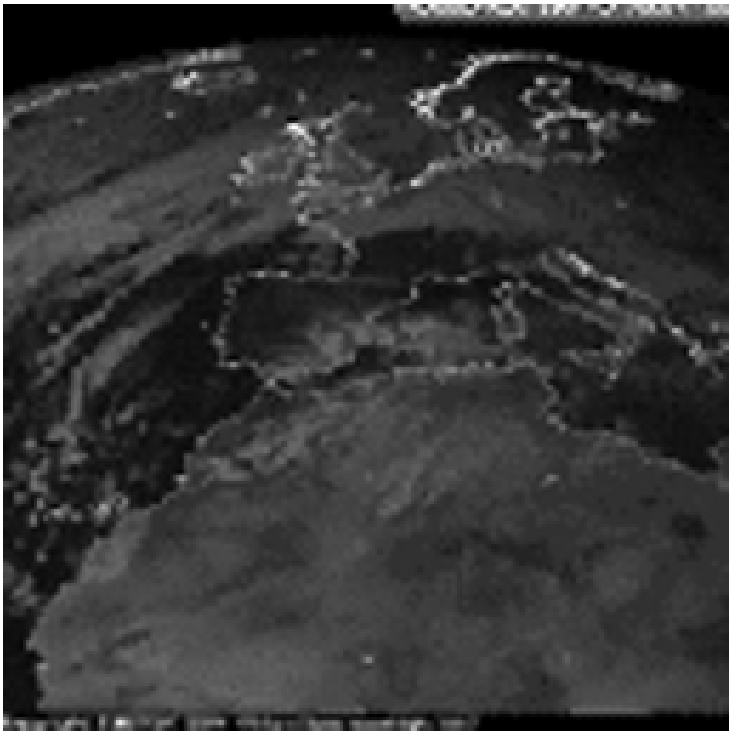
Une fois que nous avons fixé les paramètres de notre application et avoir choisi l'algorithme d'optimisation, à savoir, la taille des blocs cibles – sources qui est de 2-4 pixels et l'algorithme génétique pour la recherche des blocs les plus similaires. Nous avons appliqué notre technique sur un ensemble d'images météorologiques prises par le satellite MSG dans les canaux 10.8 et 0.6. Les résultats obtenus sont représentés dans le tableau 10 et les images reconstruites sont donnés par la figure 34.

Les images du MSG de 27 octobre 2007 à 14 h	PSNR en (dB)	Le taux de compression en (%)	Temps d'exécution en (s)
L'image infrarouge (canal 10.8)	**35.2384**	**90.9424**	**4763.2409**
L'image dans le domaine du visible (canal 0.6)	**29.4001**	**91.2109**	**4784.4883**

Tableau 10. Résultats de la compression des images MSG.



a)



b)

Fig. 34. Les images du MSG reconstruites.

a) Image du canal 10.8.

b) image du canal 0.6.

Les résultats de compression des images MSG que nous avons obtenus ont montré que l'image infrarouge du canal 10.8 présente un taux de compression de 90.94% et un PSNR de 35.23 dB. Par contre, l'image dans le domaine du visible du canal 0.6 a un taux de compression de 91.21% et un PSNR de 29.4 dB. La différence de netteté pour les deux images satellitaires revient que l'image dans le domaine du visible présente plus de détails que celle d'infrarouge. L'image infrarouge est utilisée pour suivre l'évolution des nuages, l'observation des aérosols et le suivi de la végétation. Par contre, l'image visible permet de mesurer la température de surface de la terre et de la mer, détection des cirrus et déduction des quantités d'eau précipitable au-dessus de la mer. Nous remarquons aussi, que le temps d'exécution sur les deux images a une différence de 20 secondes lors de la compression. A partir de ces résultats, nous déduisons que la compression par fractales prend en considération la quantité et la qualité d'information présente dans l'image à compresser.

IV.7- Discussion :

L'application de notre méthode de compression a permis de déduire que la partie d'optimisation représentée par les algorithmes génétiques, a permet de réduire le temps d'exécution et de trouver le bloc le plus similaire. La variation de la taille des blocs nous a permis de fixer une taille qui est 2-4 pour avoir une meilleure qualité d'image reconstruite. Nous avons appliqué notre algorithme sur limage test Lena, et nous avons obtenu un PSNR de 33.02 dB et un taux Tc de 86.42% (pour une taille des blocs 2-4). Les résultats obtenus par notre méthode confirment l'aptitude et l'efficacité de notre approche pour la réduction le volume des données satellitaires.

CONCLUSION

Dans ce travail, nous avons mis en œuvre une méthode de compression basée sur les fractales pour répondre à la problématique du taux de donnée important transmis par le satellite MSG, lors de l'observation des phénomènes atmosphériques. Notre méthode permet de faciliter l'archivage d'images satellitaires de grande taille avec de forts taux de compression et le minimum de distorsions. L'objectif visé est d'améliorer le temps de compression tout en réalisant un bon compromis entre le taux de compression et la qualité de restitution.

Dans un premier temps, nous avons procédé à une comparaison des principales techniques de compression d'images publiées dans la littérature. Nous avons ainsi montré que les méthodes avec pertes étaient les plus adaptées à la compression des images satellitaires. Parmi ces techniques, celles utilisant les fractals semblent convenir à cette exigence. Elles ont pour objectif de réduire la quantité de données image par élimination de la redondance due aux blocs qui se répète, en gardant un bon aspect visuel de l'image reconstruite. Notre méthode est basée sur le principe de la compression par fractale en utilisant les algorithmes génétiques pour l'optimisation de l'étape d'apprentissage pour générer le code fractal.

Dans l'étape d'apprentissage, le nombre d'opérations effectué pour la recherche des similitudes entre les blocs augmente avec le temps de compression. L'utilisation des algorithmes génétiques pour la recherche du code fractal a réduit considérablement l'espace de recherche ainsi que le temps de compression /décompression tout en réalisant un bon compromis entre le taux de compression et la qualité de restitution. L'application de notre méthode sur l'image test Lena a permis de fixer les meilleurs paramètres pour le codage. La variation de la taille des blocs nous a permis de faire varier le taux de compression. L'application sur l'image test Lena a permis d'atteindre un taux de compression de 86.42 % avec un PSNR de 33.02 dB.

L'application de la méthode élaborée à des images Météosat MSG dans les canaux visible et infrarouge a permis d'une part de réduire considérablement le volume des données satellitaires à sauvegarder et d'autre part d'obtenir un PSNR de 35,23 décibels. Elle représente

donc un outil intéressant pour l'archivage d'images météorologiques sur de longues périodes et par conséquent, l'étude des changements climatiques à long terme.

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

- Dans l'étape d'apprentissage par exemple, il serait intéressant de localiser les blocs texturés et les blocs uniformes et de traiter chaque région par une méthode appropriée.
- Nous pouvons aussi améliorer encore la vitesse de notre application, par l'utilisation d'un zoom arrière avant l'étape d'isométrie.
- Appliquer un traitement global sur l'image d'entrée au lieu de l'appliquer pour chaque pour chaque bloc.
- Utiliser un partitionnement triangulaire pour réduire les artefacts aux niveaux des contours.

ANNEXE A

LES FRACTALES

A.1-Les fractales :

A.1.1-Théorie des IFS :

La théorie des IFS est fondée principalement sur la propriété d’invariance par changement d’échelle. Dans le cadre de la géométrie fractale, elle devient une méthode qui consiste à prendre un ensemble de transformations contractantes, qui présentent des affinités les unes envers les autres pour exprimer des relations, entre les différentes parties d'une même image.

A.1.2-Théorème du point fixe :

Ce que l'on conçoit visuellement sur l'exemple du triangle de Sierpinsky (voir fig.A.1) a un fondement mathématique: c'est une application du théorème du point fixe:

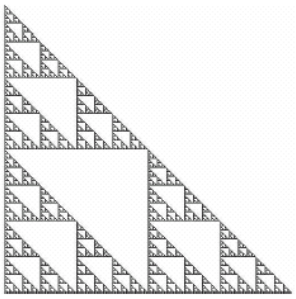


Fig.A.1 le triangle de Sierpinsky.

On pourrait choisir une résolution aussi grande que l'on veut, elle ne suffirait pas pour représenter cette image correctement. Les détails sont aussi petits que l'on veut, chaque triangle visible est en fait composé de 3 autres triangles (voir fig.A.1), eux même composés chacun de 3 triangles, et ainsi de suite...

Soit E un ensemble, muni d'une distance *d*, complet, *f* une application de E dans E contractante**:**

$$\forall a,b \in E,\ \ d(f(a),f(b)) < d(a,b)$$

Alors, il existe un unique *x* dans E tel que $f(x) = x$, et de plus, toute suite définie par :

$U_{n+1} = f(U_n)$ Converge vers *x* et *x* est le point fixe de *f.*

Soit donc E l'ensemble de $[0..255]^{n*m}$, muni de la norme euclidienne :

$$\|A\| = \sqrt{\sum_{i,j} A[i,j]^2}$$

Cet espace est complet car il est fermé, borné d'un espace vectoriel de dimension finie.

Le théorème du point fixe peut donc s'appliquer. L'espace que nous utiliserons pour représenter les images va être celui directement relié au format des images digitales : Un ensemble de pixels dont l'intensité est quantifiée (on se limitera aux images en niveau de gris).

La donnée de l'application contractante suffit à définir l'image, et celle-ci peut être représentée en appliquant itérativement la fonction à partir d'une image quelconque. Le code d'une image sera la description de la fonction dont elle est le point fixe.

A.1.3- Les applications affines dans K^n :

Ce genre de transformation est très simple en principe. Pourtant, ils sont à l'origine de la naissance de la théorie des IFS. Les applications affines dans *K* sont exactement les applications f : K → K de la forme :

$$f(x) = ax + b, \quad \forall x \in K$$

Avec a et b deux scalaires quelconques. On a alors: b = f(0).

De façon plus générale, une application affine $f : K^n \to K^m$ est une application de la forme :

$$f(X) = A \cdot X + B$$

Où A est une matrice de taille (m * n) et B une matrice de taille (m * 1). Alors,
B = f(O), O étant le vecteur nul de K^n.

A.1.4- Systèmes de Fonctions Itérées (IFS) :

Un IFS est un système de fonctions itérées qui, lorsqu'on l'applique à un objet géométrique, permet d'obtenir une fractale. Ces fonctions sont affines et contractantes. Le terme contractant signifie que lorsque l'on applique ces fonctions à une figure, les points de celle-ci ne s'éloignent pas, mais se rapprochent, il n'y a donc pas de divergence, on obtient une

figure contenue dans un espace fini. Dans le cas de figure de dimension 2, la figure fractale obtenue occupe donc une surface finie, qui n'est autre que la surface de l'objet initial.

L'IFS possède une propriété très utile. En effet, il peut converger vers un élément fixe indépendamment des contraintes initiales. On appellera cet élément l'attracteur de l'IFS, et il a été démontré aux années 80 qu'il est de type fractale.

A.2- La compression des images fractales :

Pour comprendre le fonctionnement de la compression fractale, il est important de comprendre sur quels faits l'algorithme se base. Dans toute image naturelle, ou d'images représentant des choses réelles, il existe des self –similarités.

L'image Lena illustrée par la figure A.2 représente des portions d'images qui se ressemblent (le chapeau et l'épaule), à quelques transformations près. Ces transformations peuvent être géométriques (rotation, mise à l'échelle) ou sur les pixels (luminosité, couleur, contraste).

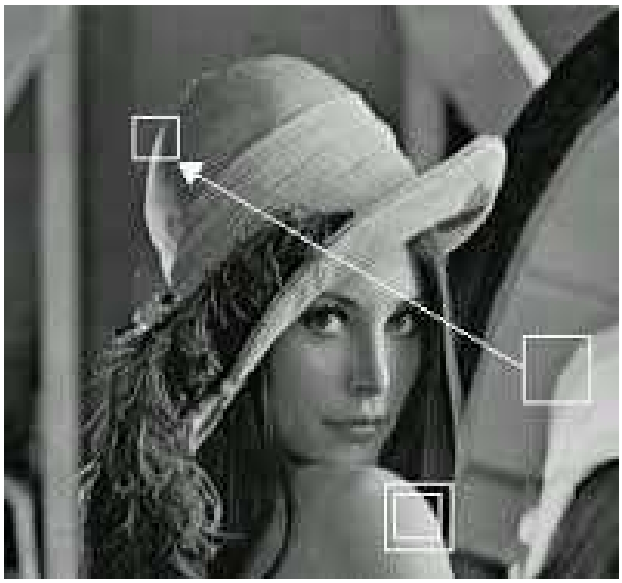


Fig. A.2 Image Lena qui représente des self –similarités.

Une fois qu'une image est représentée par des matrices, on peut se demander si des opérations sur leurs éléments influencent l'image correspondante. Pour manipuler une image

on a qu'a manipulé sa matrice qui la représente fidèlement .On a choisi deux types de manipulation : une manipulation sur la géométrie et une manipulation sur la taille.

A.2.1- Manipulation sur la géométrie :

Elle consiste à faire des rotations sur les matrices son aucun changement sur la valeur des pixels qu'elle la compose, ce qui correspond à des rotations pour les images qu'elle représente.

Prenons un exemple pour les illustrer, soit A une matrice quelconques de 3 x 3 et les a_{ij} les éléments de A. les matrices B,C,D,E,F,G et H sont des matrices qui montres les résultats de différentes rotations de A .

$$A=\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} B=\begin{pmatrix} a_{13} & a_{23} & a_{33} \\ a_{12} & a_{22} & a_{32} \\ a_{11} & a_{21} & a_{31} \end{pmatrix} C=\begin{pmatrix} a_{33} & a_{32} & a_{31} \\ a_{23} & a_{22} & a_{21} \\ a_{13} & a_{12} & a_{11} \end{pmatrix} D=\begin{pmatrix} a_{31} & a_{21} & a_{11} \\ a_{32} & a_{22} & a_{12} \\ a_{33} & a_{23} & a_{13} \end{pmatrix}$$

$$E=\begin{pmatrix} a_{13} & a_{12} & a_{11} \\ a_{23} & a_{22} & a_{21} \\ a_{33} & a_{32} & a_{31} \end{pmatrix} F=\begin{pmatrix} a_{31} & a_{32} & a_{33} \\ a_{21} & a_{22} & a_{23} \\ a_{11} & a_{12} & a_{13} \end{pmatrix} G=\begin{pmatrix} a_{11} & a_{21} & a_{31} \\ a_{12} & a_{22} & a_{32} \\ a_{13} & a_{23} & a_{33} \end{pmatrix} H=\begin{pmatrix} a_{33} & a_{23} & a_{13} \\ a_{32} & a_{22} & a_{12} \\ a_{31} & a_{21} & a_{11} \end{pmatrix}$$

- *La matrice B est le résultat de la rotation de A de 90° dans le sens antihoraire.*
- *La matrice C est le résultat de la rotation de A de 180° dans le sens antihoraire.*
- *La matrice D est le résultat de la rotation de A de 270° dans le sens antihoraire.*
- *La matrice E est le résultat de la rotation de A dans le sens vertical.*
- *La matrice F est le résultat de la rotation de A dans le sens horizontal.*
- *La matrice G est le résultat de la rotation de A par rapport à la 1^{er} diagonal.*
- *La matrice H est le résultat de la rotation de A par rapport à la 2^{eme} diagonale.*

La figure A.3.a représente l'image teste « Lena » avec une rotation dans le sens horizontal. Et la figure A.3.c montre une rotation par rapport à la deuxième diagonale .Ses

images avec des rotations différentes ont été obtenue par une simple rotation des éléments de la matrice qui défini l'image Lena .la figure A.3.b est l'image originale.

a -le sens horizontal b - originale c - la 2eme diagonale

Fig.A.3 manipulation géométrique de l'image Lena.

A.2.2- Manipulation de la taille des blocs (ou Transformation spatiale) :

La représentation des blocs par des matrices nous offre la possibilité de les manipuler facilement. La manipulation sur la taille de bloc consiste à augmenter ou à diminuer la taille de la matrice qui représentait une image ce qui veut dire qu'on va ajoute des éléments à la matrice originale ou bien de supprimer quelque élément, se qui ce traduit par un zoom avant ou un zoom arrière sur le bloc en considération.

La matrice A est la matrice originale de taille 3 x 3, la répétition des éléments de A forme la matrice I de taille 6 x 6.

$$I = \begin{pmatrix} a_{11} & a_{11} & a_{12} & a_{12} & a_{13} & a_{13} \\ a_{11} & a_{11} & a_{12} & a_{12} & a_{13} & a_{13} \\ a_{21} & a_{21} & a_{22} & a_{22} & a_{23} & a_{23} \\ a_{21} & a_{21} & a_{22} & a_{22} & a_{23} & a_{23} \\ a_{31} & a_{31} & a_{32} & a_{32} & a_{33} & a_{33} \\ a_{31} & a_{31} & a_{32} & a_{32} & a_{33} & a_{33} \end{pmatrix} \qquad A = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}$$

Fig.A.3 Transformation spatiale.

ANNEXE B
LES ALGORITHMES GENETIQUES

B.1-Définition :

Les algorithmes génétiques sont des procédures itératives qui s'inspirent des mécanismes de sélection naturelle et des phénomènes génétiques. Ce type d'algorithme utilise un vocabulaire similaire à celui de la génétique.

Un algorithme basé sur la génétique pour trouver les solutions les plus efficaces. Ainsi pour un problème d'optimisation donné, on parlera alors :

- chromosome = solution (bonne ou mauvaise).
- gêne.
- bassin de population.
- en favorisant les solutions les plus efficaces.

L'algorithme génère de façon itérative de nouvelles populations d'individus sur lesquelles on applique des principes de sélection, de croisement, de mutation telle qu'on peut en voir en génétique. L'idée directrice est d'assimiler l'application de l'algorithme au processus d'évolution naturelle.

Ainsi, le mécanisme consiste à faire évoluer à partir d'un tirage initial un ensemble de points de l'espace représentés à l'aide d'un code en général par une suite de nombre vers le ou les optima du problème d'optimisation. L'ensemble du processus s'effectue à la taille de population que l'on notera N, de sorte que les générations successives comportent toutes N individus.

B.2-Les fondements des algorithmes génétiques :

B.2.1-Les représentations d'une solution :

- solution : n'importe quel candidat pour une solution correcte pour F(x) = 4 - (x-3)2 = 0 ; x=2 , x=0 sont des solutions mais seule 1 et 5 sont des solutions correctes.
- solution ≈ chromosomes (bagage génétique)
- solutions encodées sous forme de chaînes de caractère, de graphe, de matrice, d'un programme
- chaque caractère est un gêne.

B.2.2-Le degré d'adaptation (fitness) d'une solution [fi ou f(Ai)] :

Une fonction qui permet de comparer des solutions afin de déterminer la meilleure selon :

- le profit, l'énergie, le temps.
- souvent un indice composé.
- lorsque la solution exacte ne peut être identifiée.

On regarde si la suite des meilleures valeurs devient asymptotique ou on fixe le nombre d'itérations.

B.2.3-L'algorithme :

B.2.3.1-Création d'un bassin de reproduction :

⟹ Début de l'algorithme

• évaluer le degré d'adaptation de chaque solution dans la population et leur probabilité d'être choisie :

- si la solution correcte est trouvée alors l'algorithme se termine.
- plus une solution est adaptée, plus sa probabilité d'être choisie est grande.

• choisir au hasard (en respectant la distribution des probabilités) des solutions.

B.2.3.2-Croisement et reproduction :

• former une paire de solutions au hasard.

• pour chaque paire, choisir au hasard s'il y a croisement :

- non alors conserver les deux solutions pour le prochain bassin de population.

- oui choisir au hasard le point de croisement puis effectuer le croisement.

• recommencer jusqu'à l'atteinte du nombre de solutions dans la population originale.

un site est un nombre *k* choisi dans [1, L-1], L= longueur de la chaîne ici *k* = 3

	avant	après
Solution 1	**+++----**	**+++vvvv**
Solution 2	**…vvvv**	**…----**

B.2.3.3-Mutation :

• choisir au hasard un ensemble de solution.

• modifier au hasard un caractère : l'idée est de créer une solution qui n'était pas accessible à partir de la population initiale.

Retour au début de l'algorithme ⟹

B.3-Scénario simple d'une application de l'algorithme :

B.3.1-Définition du problème :

3 machines, maximiser la production totale une solution est une chaîne de 3 bits (1 : en fonction ; 0 : arrêt) la solution est 111

Nombre de population n = 4

Population initiale
101
001
010
110

B.3.2-Création d'un bassin de reproduction :

i	$Ai_{base\ 2}$	fi_{base10}	pi=fi/total	n*pi
1	101	5	0.357	1.429
2	001	1	0.071	0.286
3	010	2	0.143	0.571
4	110	6	0.429	1.714
	total	14	1	4
	moyenne	3.5	0.25	1
	max	6	0.429	1.714

- méthode de choix aléatoire

i	Pi	nombre aléatoire
1	0.375	000-356
2	0.071	357-427
3	0.143	428-570
4	0.429	571-999

-choisir aléatoirement un nombre entre 0 et 999

- bassin de population créé

Population de reproduction
010
101
110
110

B.3.4-Croisement et reproduction :

• former une paire de solutions au hasard (101 et 110)

• choisir au hasard s'il y a croisement : oui

• choisir au hasard un site de croisement : k =2

avant	après
10 :1	10 :0
11 :0	11 :1

• former l'autre paire de solutions (010 et 110)

• choisir au hasard s'il y a croisement : oui

• choisir au hasard un site de croisement : k =1

avant	après
0 :10	0 :10
1 :10	1 :10

B.3.4-Mutation :

• non

B.3.5-Prochaine itération création d'un bassin de reproduction :

i	Ai	fi
1	010	2
2	100	4
3	111	7
4	110	6
	total	19

• La solution correcte est trouvée (fin de l'algorithme)

• C'était un problème d'optimisation qui cherche le maximum.

BIBLIOGRAPHIE

[1] Document officiel du capteur SEVIRI (Spinning Enhancer Visible and Infrared Imager) embarqués sur le MSG, http://eduscol.education.fr/orbito/system/meteosat/met23.htm

[2] EUMETSAT, "MSG Level 1.5 Image Data Format Description", EUM/MSG/ICD/105 v6, 23 Germany, February 2010.

[3] AM. Cohen and HL. Resnikoff , " Image Compression for Radiology and Telemedicine ", Proc. SPIE 2298, SPIE, Bellingham, Wash., pp. 304-315, 1994.

[4] D. A. Huffman," A method for the construction of minimum redundancy codes ", IRE Proc., Vol. 40, pp. 1098-1101, September 1952.

[5] M. Lahdir, "Nouvelle approche de compression d'images basée sur les ondelettes et les fractales : Application aux images Météosat ", thése de Doctorat en électronique : Option Télédétection, faculté de génie Electrique et Informatqiue, Université Mouloud Mammeri de Tizi-Ouzou UMMTO, p. 120, 2007.

[6] J. M. Moureaux, "Quantification vectorielle algébrique pour la compression d'images - Application à l'imagerie radar à synthèse d'ouverture (SAR) ", thèse de Doctorat en Sciences de l'ingénieur, Université de Nice- Sofia Antipolis, France, p. 130, 1994.

[7] B. Ramamurthi and A. Gersho, "Classified vector quantization of images ", IEEE Trans. Commun. 34, 11, pp. 1105-1115, 1986.

[8] M. D. Adams, "The JPEG-2000 Still Image Compression Standard", ISO/IEC JTC 1/SC 29/WG 1 N 2412, Dec. 2002.

[9] S. Renard ,"Première approche de JPEG 2000" , Conférence, pp 02-11, Corbeil-Essonnes**,** France, Sep 2003.

[10] T. Acharya and P.-S. Tsai, JPEG2000 Standard for Image Compression: Concepts, Algorithms, and VLSI Architectures, John Wiley & Sons, Inc., NJ, p. 296, 2005

[11] A. E. Jacquin, “Fractal image coding based on a theory of iterated contractive image transformations”, In Proceedings SPIE Visual Communications and Image Processing ’90, Vol. 1360, pp. 227–239, 1990.

[12] M. Lahdir, S. AMEUR et H. ADANE "Algorithme non itératif basé sur les ondelettes biorthogonales et les fractales pour la compression d’images satellitaires", Télédétection, Vol. 6, n° 4, p. 345-360, 2006,

[13] S. V.Veenadevi and A. G. Ananth, "fractal image compression using quadtree decomposition and huffman coding", Signal & Image Processing : An International Journal (SIPIJ) Vol.3, No.2, April 2012.

[14] A.E.Jacquin, "Image Coding Based on a Fractal Theory of Iterated Contractive Image Transformations", IEEE Transaction. Image processing, Vol. 1, n° 1, pp. 18–30 . Jan 1992.

[15] A.E.Jacquin, "Fractal Image Coding a Review", Proceedings of the IEEE, Vol. 81, n°. 10, pp. 1451 - 1465 , Oct 1993

[16] Y. Chakrapani and K. S. Rajan, " Genetic algorithm applied to fractal image compression" , ARPN Journal of Engineering and Applied Sciences ,Vol. 4, no 1, ISSN 1819-6608, 2009.

[17] N. Barnier et P. Brisset, "Optimisation par Algorithme Génétique sous Contraintes ", Technique et science informatiques, Vol 18, no 1, pp. 01-23, 1999

[18] Ming-Sheng Wu, "Spatial correlation genetic algorithm for fractal image compression", Chaos, Solitons and Fractals, Vol. 28, Issue 2, pp. 497–510, 2006.

[19] N. Srinivas et K. Deb "Multiobjective Optimization using Nondominated Sorting in Genetic Algorithms ". Evolutionary Computation, Vol. 2, n° 3, pp. 221-248 , 1994.

[20] Y. Frank, "Robust watermarking and compression for medical images based on genetic algorithms", Information Sciences, Vol. 175, Issue 3 , pp. 200–216, 2005.

[21] B.Sareni, "Algorithmes génétiques standards." Rapport interne R-97-01, Centre de génie Electrique de Lyon, Ecole centrale de Lyon, France , 1997.

[22] EUMETSAT," Météosat Seconde Génération ", l'Agence Spatiale Européenne (ESA), centre de Météorologie spatiale, France.

[23] D. M. A. Aminou, B. Jacquet and F. Pasternak, "Characteristics of the Meteosat Second Generation Radiometer/Imager :SEVIRI" , Proceedings of SPIE, Europto series, Vol. 3221, pp. 19-31.

[24] "Estimation du temps de transfert", http://www.neobe.com/fr/estimation-transfert

[25] C. WAGNER , " De l'image vers la compression ", Rapport de Recherche de l'INRIA, n° 2035, ISSN 0249-6399, p. 38, Septembre 1993.

[26] C. E. Shannon, "A mathematical theory of communication", Bell System Technical Journal, Vol. 27, p. 379-423 and 623-656, Oct 1948.

[27] K. L. Chung and S. Y. Tseng, "New progressive image transmission based on quadtree and shading approach with resolution control" , Pattern Recognition Letters, Vol. 22, Issue 14, pp.1545–1555 , Dec 2001.

[28] V. P.Baligar , L.M. Patnaik, and G.R. Nagabhushana, "High compression and low order linear prediction for lossless coding of grayscale images. " Image and Vision Computing., vol. 21, no 6, pp.543–550, 2003.

[29] G. Mercier, C. Roux and G. Martineau, "Technologies du Multimedia", ENST Bretagne, Département ITI, Brest, p. 117, France, 2003.

[30] J.D. GRETILLAT, " modification de l'algorithme de la transformée en ondelettes discrète

pour l'obtention d'une représentation invariante sous rotation ", département de physique, de génie physique et d'optique faculté des sciences et de génie université laval, Québec, juin 2006

[31] S. Abdat ,"Compression d'images fixes par des fractales en utilisant les algorithmes génétiques" , Mémoire de Magister en électronique : Option Télédétection, Faculté de génie Electrique et Informatqiue, Université Mouloud Mammeri de Tizi-Ouzou UMMTO, 2007.

[32] M. Jampour, M. Yaghoobi and M. Ashourzadeh, " Fractal Images Compressing by Estimating the Closest Neighborhood with Using of Schema Theory " , Journal of Computer Science, Vol. 6, no. 5, pp. 591-596, ISSN 1549-3636 , 2010.

[33] S. Douda, A.A.E. Imrani et M. Limouri, "Nouvelle approche d'accélération du codage fractal d'images", Numéro spécial CARI 2008, 98 ARIMA, Vol. 11, pp. 97-114, Maroc, 2008.

[34] Y. Fisher, "Fractal image compression Theory and Application. " Springer-Verlag, London, p. 234, 1995.

[35] "La compression par fractales - Comparatif des méthodes de compression d'images" http://etud.insa-toulouse.fr/~flone_sa/BEmultimedia/index.php?Fractales.