République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Ecole Nationale Polytechnique

DEPARTEMENT D'ELECTRONIQUE
OPTION : TRAITEMENT DU SIGNAL ET COMMUNICATIONS

MEMOIRE DE MAGISTER

PRESENTE PAR : **M^elle MEKHALFA FAIZA**

Ingénieur d'état en électronique (ENP)

Thème

Compression d'Images Fixes à base de la Géométrie Fractale

Soutenu devant le jury composé de :

Mr A. BELOUCHRANI	**Maître de Conférence à l'ENP**	**Président**
Mr D. BERKANI	**Professeur à l'ENP**	**Rapporteur**
Mme I.. HAMAMI	**Chargé de cours à l'ENP**	**Examinateur**
Mr B. BOUSEKSOU	**Chargé de cours à l'ENP**	**Examinateur**
Mr H. BOUSBIA-SALAH	**Chargé de cours à l'ENP**	**Examinateur**

E.N.P.10, Avenue Hassen-Badi, EL-HARRACH, ALGER.

Sommaire

Chapitre 3 : *Théorie des IFS et compression d'images par fractales*

Chapitre 4 : *Conception et mise en Œuvre*

Dédicaces

À mon Père et à toute ma famille

Remerciements :

J'exprime ma profonde gratitude à Mr D. Berkani, professeur à l'ENP pour les conseils avisés et l'encouragement constant qu'il n'a cessé de me prodiguer tout au long de cette étude.

Mes remerciements les plus sincères sont adressés à Mr A. Belouchrani, maître de conférence à l'ENP, qui m'a fait l'honneur de présider le jury de mémoire.

Mes vifs remerciements vont aussi à Mme L. Hamami, chargé de cours à l'ENP, je suis très honorée de sa présence dans le jury de mémoire.

J'exprime ma grande gratitude à Mr B. Bouseksou, chargé de cours à l'ENP, pour l'honneur qu'il m'a fait en acceptant de faire partie au jury de mémoire.

Je remercie également Mr H. Bousbia- Salah , chargé de cours à l'ENP, pour l'intérêt qu'il a porté à ce travail en me faisant l'honneur de participer au jury de mémoire.

Je tiens à ce que tous mes collègues et amis trouvent ici mes remerciements les plus sincères.

ملخص

تكتسي طرق تقليص الصور الرقمية أهمية بالغة في التطبيقات الخاصة بتخزين و إرسال الصور.
في هذه الأطروحة،قدمنا خوارزمية لتقليص الصور تعتمد على نظريات الفراكتال و الدوال المتكررة.
نظام التقليص يرتكز على إنشاء دالة خاصة للصورة الأصلية، تدعى برمز الفراكتال.
يتم تكرار هذا الرمز على أي صورة ابتدائية لإعطاء سلسلة من صور، هذه السلسلة تتقارب نحو تمثيل الفراكتال الصورة الأصلية.

كلمات المفاتيح : تقليص الصور، فراكتال، نظام الدوال المتكررة، التقليص بالفراكتال، تقسيم الصورة، تقسيم مربع.

Abstract :

The conception of digital image coding techniques is of great interest in various areas concerned with the storage or transmission of images.
In this work, we have developed algorithm to image coding, based on a fractal theory of iterated transformations.
The coding-decoding system is based on construction, for an original image to encode, of a specific image transformation (code fractal) which when iterated on initial image, produces a sequence of images which converges to a fractal approximation of the original.

Keywords: Image Compression, Fractal, Iterated Functions System IFS, Fractal Compression, Image partition, Square partition.

Résumé:

La conception des techniques de compression d'images digitales, possède une grande importance dans les applications destinées au stockage et transmission d'images.
Dans ce travail, on a développé un algorithme de compression d'images à base de la théorie des fractales et les transformations itérées.
Le système de codage –décodage est basé sur la construction, d'une transformation spécifique de l'image originale, appelé code fractal. Ce code sera itéré sur n'importe quelle image initiale pour générer une séquence d'images reconstruites. La séquence générée converge vers l'approximation fractale de l'originale.

Mots clé : Compression d'images, Fractale, Système de fonctions itérées, Compression fractale, Partition d'image, Partitionnement carré.

Introduction

Introduction

Cette recherche a été effectuée au laboratoire signal et communications de l'Ecole Nationale Polytechnique (ENP). Le but de ce travail est d'introduire une initiation à l'application de la géométrie des fractales dans la compression des images fixes à plusieurs niveaux de gris.

L'image est considérée, de nos jours, comme le support d'information le plus performant. Son utilisation s'intensifie, de jour en jour, en télécommunications, en médecine comme dans d'autres domaines **[1][2]**. Par conséquent, des techniques de traitement, de transmission et de stockage sont nécessaires surtout avec l'apparition des applications multimédias. La grande quantité d'information que représente l'image a fait que plusieurs chercheurs se sont intéressés au domaine de la compression d'images, dans un souci de réduire au maximum le nombre de bits requis pour la représentation d'un point image, en conservant tout de même une qualité acceptable.

Les techniques de compression mises en œuvre jusqu'à présent, ont pour but de conserver dans le signal les informations les plus significatives (non redondantes et perceptibles par le système visuel humain), afin d'obtenir le meilleur rapport qualités/ capacités utilisées.

Différentes techniques de compression ont été étudiées dans la littérature **[3][4]**. Celles-ci peuvent être divisées en deux classes. La première, n'introduit aucune distorsion à l'image après décodage et restitue ainsi l'image originale du codeur. Cette classe ne permet pas d'atteindre des taux de compression élevés. La deuxième, introduit une certaine perte d'information se reflétant par des distorsions plus au moins perceptibles à l'œil. Ces méthodes permettent d'atteindre des taux de compression élevés avec une bonne qualité de l'image.

Afin d'homogénéiser les représentations et d'assurer la compatibilité, le développement de standards de représentation d'informations multimédias devient une nécessité. Les efforts de standardisation qui ont été élaborés par le groupe JPEG (Joint Photographic Experts Group), ont abouti au premier standard de compression d'images numériques pour les images fixes utilisant la transformée en cosinus bidimensionnelle.

La théorie des fractales étudiée par **MANDELBROT [5]** a été depuis, développée dans plusieurs domaines, en particulier dans la génération des images et le codage. Le concept du codage fractal provient de l'idée de **BARNSLEY [6]** pour exploiter les structures auto-similaires dans les images, dans un but de compression. Le procédé de cette méthode est basé sur une association entre certaines structures de l'image présentant des similarités suivant des critères ayant trait aux propriétés des fractales. On recherche des appariements entre des portions de l'image qui correspondent au même motif pour des échelles et des orientations différentes.

Le présent travail fixe comme objectif principal le développement d'un algorithme de compression d'images fixes via la géométrie fractale.

Afin de situer le sujet dans son contexte global, nous avons organisé ce travail en plusieurs chapitres.

On introduit dans le chapitre I, les généralités sur la compression d'images.

Dans le second chapitre, on présente les fondements de la théorie des fractales.

Le Chapitre III est consacré à l'étude théorique des systèmes IFS (Iterated Functions System) et L-IFS (Local Iterated Functions System), ainsi que leur utilisation dans la compression des images fixes.

Le dernier chapitre illustre l'implémentation de l'algorithme étudié. Il sera également consacré aux discussions et interprétations des résultats obtenus.

Enfin, nous terminons par une conclusion.

Plusieurs annexes sont aussi données, leur but est de compléter quelques notions présentes dans les différents chapitres.

Chapitre 1

Introduction à La Compression d'Images

Introduction :

L'image comme support d'information, est de plus en plus utilisée dans plusieurs domaines (les télécommunications, la médecine, la biologie, l'astronomie, la géologie, l'industrie, la météorologie et l'armement). Cependant le très grand nombre de données brutes présent dans une image numérique pose problème au moment de leur traitement, transmission ou stockage. Pour pallier ce problème, des techniques de compression d'images ont été développées, afin de réduire la quantité globale d'information présente dans une image. Ces techniques, tirent profit, de la grande corrélation qui existe entre pixels voisins d'une même image.

I. Notions de base sur le Traitement de l 'image :

I.1. Représentation : [7][8]

Représentation Analogique : Une information est dite analogique, si elle est délivrée par une source continue ou représentée par un signal analogique. Une source est continue (signal analogique) si elle possède un alphabet infini de valeurs.

Représentation numérique : Un signal est dit numérique si son alphabet est fini. On obtient un signal numérique par quantification à partir d'un signal analogique échantillonné ou discret.

I.2. Image numérique :

Le mot image signifie la reproduction exacte ou la reproduction analogique d'une scène réelle. L'image est dite numérique si elle admet une représentation numérique. L'image numérique est considérée comme une juxtaposition de points appelés pixels, ayant chacun comme caractéristique un niveau de gris ou de couleur relevé à l'emplacement correspondant dans l'image réelle [9]. L'image plane la plus générale peut être représentée par une matrice $f(i, j)$ donnant en chaque point (i, j) du plan la valeur de l'intensité lumineuse en ce point.

On appelle image binaire une image pour laquelle les éléments de la matrice $f(i, j)$ ont pour valeur f_1 ou f_2 .

I.3. Caractéristiques d'une image numérique : [2][9][10]

Pixel :

Le mot pixel est une contraction de Picture Element (élément de l'image). Il constitue le plus petit élément de l'image. Un pixel est caractérisé par son niveau de gris. La représentation du pixel dans la mémoire se fait de la manière suivante : dans une image binaire, le pixel est représenté sur un bit, 0 pour le noir et 1 pour le blanc ; pour une image monochrome à plusieurs niveaux de gris, chaque pixel est représenté par un mot de plusieurs bits.
Dans une image couleur, il peut être représenté sur trois octets : un octet pour la couleur rouge (R), un octet pour la couleur verte (V), et un octet pour la couleur bleue (B).

Niveau de Gris :

Le niveau de gris est la valeur de l'intensité lumineuse en un point de l'image. Cette valeur est limitée dans un intervalle ayant pour borne inférieure le noir, et pour borne supérieure le blanc. Le nombre de niveaux de gris dépend du nombre de bits nécessaires pour coder l'intensité de chaque pixel de l'image.

Résolution :

La résolution est le nombre de pixels disponibles en lignes, et en colonnes de l'image. Elle exprime la clarté et la finesse des détails atteinte par un moniteur dans la production des images.

Contours et Textures :

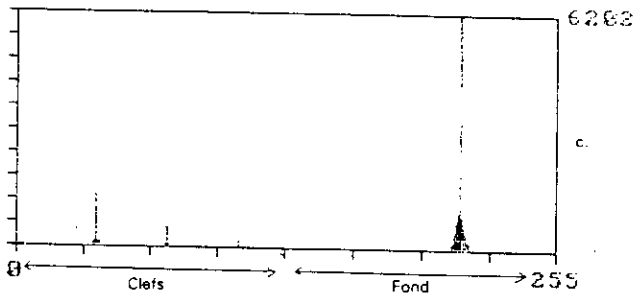
Les contours représentent la frontière entre les objets de l'image ou la limite entre deux pixels dont les niveaux de gris représentent une différence significative. Les textures décrivent la structure de ceux – ci.

Bruit :

Un bruit est défini comme étant des pixels isolés dans des régions homogènes, donc ce bruit affecte de brusques variations de niveaux de gris. Il peut provenir des conditions de l'expérimentation ou des appareils d'acquisition. Pour l'éliminer, on fait appel à des techniques de filtrages.

Histogramme :

L'histogramme des niveaux de gris d'une image est une fonction qui donne la fréquence d'apparition de chaque niveau de gris dans l'image. Les niveaux de gris sont représentés en abscisse. Le nombre de pixels affecté par chaque niveau de gris est affiché en ordonnée.
L'histogramme permet de donner un grand nombre d'informations sur la distribution des niveaux de gris de l'image. Il permet notamment d'analyser entre quelles bornes est répartie la majorité des niveaux de gris dans le cas d'une image trop claire ou trop foncée.



FigureI.1. Histogramme.

I.4. Acquisition du signal image:

Les principaux dispositifs d'acquisition d'images sont les caméras numériques et les scanners. Les scanners sont aujourd'hui bien connus, surtout dans leurs applications médicales **[2][9]**. Dans ces dispositifs, les données doivent subir une conversion analogique / numérique permettant une représentation adaptée dans un traitement par ordinateur.

Le développement technologique a permis l'apparition de nouveaux périphériques d'acquisition appelés cartes d'acquisition, qui reçoivent les données issues d'une source vidéo. Les données reçues vont être numérisées en temps réel pour être restituées sur écran. Il existe une variété de ces cartes qui fait appel à différentes normes de compression.

I.5. Traitement Numérique d'images:

Le traitement d'images est un ensemble de techniques, de transformations et de modifications destinées soit à améliorer la qualité de celles-ci soit à extraire des informations. Les systèmes de traitement d'images sont utilisés essentiellement pour réaliser les objectifs suivants **[1][2][4][9]**:

- L'amélioration de la qualité de l'image, par réduction des distorsions, augmentation du contraste,...
- L'extraction des informations utiles, par la détection des contours, la segmentation, le filtrage des fréquences spatiales et la restauration des images dégradées.
- La reconnaissance des formes, qui fait appel à l'extraction des primitives, c'est à dire d'un ensemble de paramètres caractéristiques tels que le périmètre, la surface,.... puis à l'utilisation de méthodes de classification permettant la reconnaissance d'objets ou de zones de l'image, dont les caractéristiques sont connues
- La synthèse de l'image : qui concerne l'ensemble des techniques graphiques utilisant un calculateur (infographie) permettant, en particulier le tracé de primitives graphiques, le fenêtrage, l'ombrage, la perspective et l'élimination des parties cachées et cela, dans le but d'obtenir un rendu réaliste d'une scène totalement conçue artificiellement.
- La réduction de la quantité d'information à manipuler si on veut transmettre ou stocker des images tout en les dégradant le moins possible, afin de gagner en vitesse de transmission et en capacité de stockage.

I.6. Le format des fichiers d'image : [11]

L'image numérisée, se trouve en mémoire sous forme d'une matrice de pixels. Dans ce format simple l'image est dite en mode **raster**, et le contenu de la mémoire n'est rien d'autre que l'ensemble des valeurs des niveaux de gris acquis par le dispositif de numérisation.

Afin de stocker une telle image dans un fichier, il s'avère nécessaire de précéder les données constituant l'image par une entête qui permettra de la reconstruire par la suite, ainsi plusieurs formats de fichiers images ont été conçus : TIFF, TGA, JMG, JPEG, PCX, GIF, BMP... D'une manière générale, un fichier graphique est constitué de deux parties distinctes à savoir : L'entête, qui contient des informations concernant l'image stockée et la zone de données : qui contient l'image elle-même, c'est à dire l'ensemble des pixels qui la constitués.

II. Notions fondamentales sur la théorie de l'information : [7][12]

II.1. Entropie d'une Source:

Considérons une source discrète X. Chaque échantillon temporel x(i) est indépendant des échantillons x(i-1), x(i-2),....., x(N-1) et ne peut prendre que N-1 valeurs différentes dans l'ensemble $\{a_0, a_1, ...a_{N-1}\}$.
Chaque échantillon est associé à une probabilité $P(x(i) = a_i) = P(a_i)$. On définit ainsi les probabilités : $P(a_0)$, $P(a_1)$, ..., $P(a_{N-1})$.

L'entropie de X est définie comme suit :

$$H(X) = -E[\log_2 P(X)] = -\sum_{i=0}^{N-1} P(a_i) \log_2 P(a_i) \qquad (I.1)$$

H(X) ainsi définie s'exprime en bits. elle mesure l'information moyenne d'une source. Dans notre cas nous pouvons écrire :

$$0 < H(X) < \log_2 N \qquad (I.2)$$

De cette double inégalité on peut souligner les remarques suivantes :

- Lorsque H(X) est nulle, cela se traduit par une seule probabilité non nulle d'un élément de l'alphabet. Dans ce cas la source est totalement prédictible.
- On obtient une probabilité uniforme $P_x = 1/N$ dans le cas où $H(X) = \log_2 N$, la source serait totalement non prédictible.
- L'entropie d'une source discrète est toujours positive.

On définit aussi la redondance de la source R(X) par :

$$R(X) = \log_2 N - H(X) \qquad (I.3)$$

La redondance pouvant être considérée comme la partie inutile d'un signal.

II.2. Codage d'une source :

Le codage d'une source X consiste à associer à chaque symbole x(i) de la source un mot de code c_i de longueur l_i. Les mots de codes c_i apparaissent avec la même probabilité $P(a_i)$ que les symboles x(i) de la source X. La valeur moyenne L_m de longueur des mots de code d'une source X, est égale à la somme des longueurs l_i pondérées par les probabilités de leur apparition $P(a_i)$:

$$L_m = \sum_{i=0}^{N-1} P(a_i).l_i \qquad (I.4)$$

On constate que l'entropie H(X) d'une source X représente la longueur moyenne minimale des mots de code. Le codage d'une source X dont la longueur moyenne L_m est égale à l'entropie H(X) de la source est celui qui comprime le mieux l'information. Ce code est appelé code entropique ou sans distorsion.

II.3. La fonction débit distorsion :

Le codage entropique qui conserve la qualité initiale du signal n'est pas toujours nécessaire. En effet, le système visuel humain accepte une certaine distorsion entre le signal original et le signal décodé.

La théorie du codage nous apprend qu'il est possible de tenir compte de ces distorsions. La courbe débit/ distorsion R(D) donne pour une distorsion donnée, un débit minimal R et réciproquement pour un débit donné la plus faible distorsion possible D. Cette courbe est une limite théorique des performances des systèmes de codage. L'objectif de toute méthode de codage est d'atteindre une performance assez proche de cette limite.

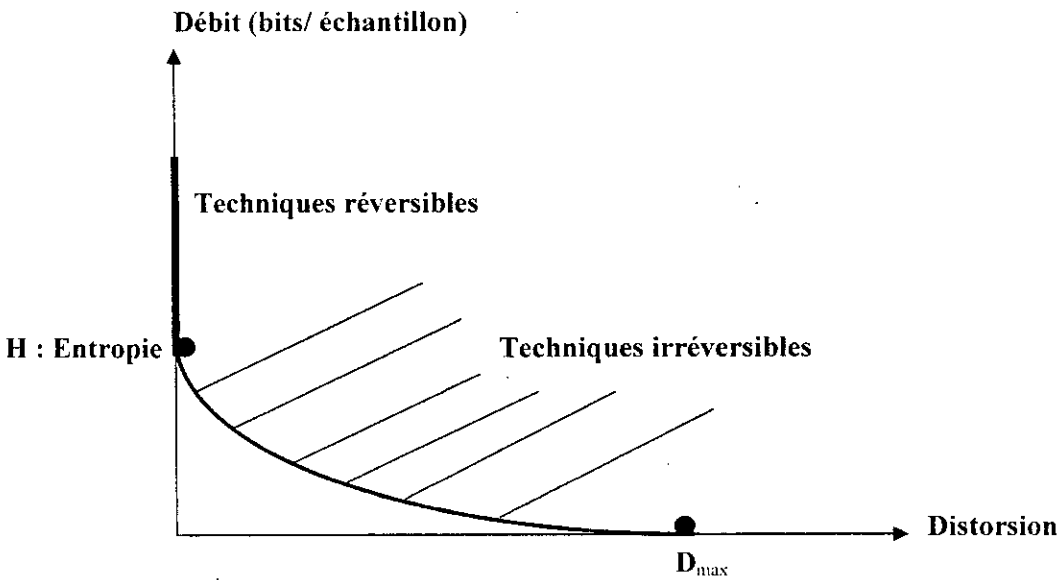


Figure I.2: La courbe R(D) pour une source discrète à alphabet fini.

III. Compression d'images :

La quantité énorme d'éléments binaires qu'il faut pour représenter une image, vient du fait que pour le codage, on ne tient pas compte des particularités de forme et de statistiques du signal image. Pratiquement, on ne peut pas, dans la majorité des cas, tenir compte des particularités de forme de chaque image sans aboutir à une méthode de codage différente. On cherche donc à mettre en œuvre des méthodes de compression valables quel que soit le type d'image.

En général, la compression du signal image consiste à réduire le nombre de bits par pixel en exploitant la redondance informationnelle de l'image.

Lorsque l'on numérise une image sur n bits, chaque pixel ne peut prendre que 2^n valeurs distinctes comprises entre 0 et $2^n - 1$. On peut considérer une image comme étant une source de 2^n valeurs entières comprises entre 0 et $2^n - 1$. On note la source X de 2^n valeurs de niveaux de gris sous la forme : **[13]**

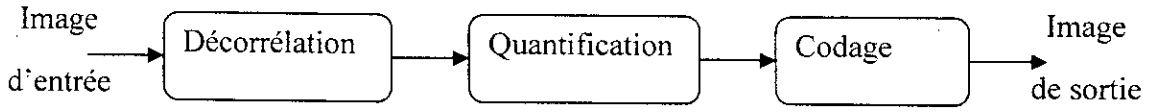
$$X = \{x(0), x(1), \ldots\ldots\ldots\ldots x(N)\}, \text{ avec : } N = 2^n - 1$$

et le champ de probabilités P associé à la source X par l'ensemble :

$$P = \{P_0, P_1, \ldots\ldots\ldots\ldots P_N\}$$

Ces probabilités représentent les fréquences d'apparition de chaque niveau de gris.

La compression se fait en trois grandes étapes : la décorrélation, la quantification et le codage. La décorrélation consiste à extraire et éliminer les données redondantes dans l'image. Le principe de la décorrélation est de transformer les pixels initiaux en un ensemble de coefficients moins corrélés. La quantification des coefficients décorrélés à pour but de réduire le nombre de bits nécessaire pour leur représentation. Les coefficients quantifiés sont ensuite codés. Le codage utilisé est le codage entropique.



FigureI.3 Schéma typique d'un codeur Source **[14]**

III.1. Redondances dans l'image :

Dans un signal image la redondance se présente sous deux formes :

La redondance spatiale :

La redondance spatiale est basée sur la forte corrélation existante entre les pixels d'une même image. Cette corrélation provient du fait que la luminosité varie lentement d'un pixel à un autre.

La redondance temporelle :

Elle est liée aux changements perçus dans les séquences d'images successives. Des méthodes d'estimation et de compensation de mouvements sont généralement utilisées pour extraire cette redondance.

III.2. Techniques de Compression d'Images Fixes :

III.2.1. Techniques sans distorsion :[4][13]

Ces techniques sont représentées dans la courbe R(D) (figure.I.2) par le segment confondu avec l'axe des ordonnées, c'est à dire D = 0 pour les débits supérieurs à H.

Ces méthodes sont appelées aussi les méthodes réversibles, elles permettent de retrouver exactement les échantillons de l'image originale. Dans ces méthodes on associe 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 les plus probables et inversement pour les niveaux possédant une faible probabilité.

Les techniques de codage les plus répandues dans ce cas sont celles de Shannon– Fano et de Huffman.

Codage de Shannon – Fano :

Il est basé sur la simple connaissance de la probabilité d'occurrence de chaque symbole d'une source. Un arbre de Shannon – Fano est construit à l'aide d'un algorithme spécifique conçu pour définir une table de code efficace.

L'algorithme est le suivant :

i)- Classer les symboles (valeurs des niveaux de gris) par ordre de probabilités décroissantes.
ii)- Partager l'ensemble des symboles en deux sous-ensembles de probabilités aussi proches que possible.
iii)- Attribuer à chaque sous-ensemble l'état 0 ou 1.
iv)- Refaire les étapes ii) et iii) pour chaque partie résultante.

Codage de Huffman :

L'algorithme de codage de Huffman s'applique sur les symboles de la source ordonnés par ordre de probabilités décroissantes. Sur cet ensemble on procède à des réductions successives jusqu'à l'obtention d'un groupe de deux symboles.

Une réduction consiste à faire la somme des probabilités des deux derniers symboles de la liste et réordonner les symboles restants. Le codage consiste à attribuer l'état 0 ou 1 à chacun des deux derniers symboles de la dernière liste établie et ainsi de suite en remontant de la dernière réduction à la première.

III.2.2. Techniques avec distorsion :

Ces techniques sont représentées dans la figure I.2 par la surface hachurée où $D \neq 0$. Les signaux sont codés avec un débit inférieur à l'entropie H. Ce sont des méthodes irréversibles qui apportent une distorsion aux images reconstruites. Ces techniques sont les plus implémentées dans les systèmes pratiques car elles permettent une réduction considérable de la quantité d'information.

Ces techniques sont classées en deux sous classes : Les méthodes directes, qui opèrent directement dans le domaine spatial et les méthodes par transformées, qui opèrent sur les coefficients d'une transformation linéaire du signal.

- **Méthodes directes :**

Méthode Prédictive : [3][15]

La méthode prédictive est l'une des plus anciennes méthodes de compression. C'est une méthode décorrélative, qui est optimale lorsque le signal est stationnaire et Gaussien.

Cette technique consiste à exprimer la valeur de chaque échantillon du signal sous forme d'une combinaison linéaire pondérée des échantillons précédents :

$$\hat{x}[n] = \sum_{i=1}^{N} a[i]x[n-i] \qquad (I.5)$$

où :

m est l'ordre de prédiction.

{x[n] ; n= 1,2, ...,N} est la séquence d'entrée.

$\hat{x}[n]$ est l'estimation du symbole x[n].

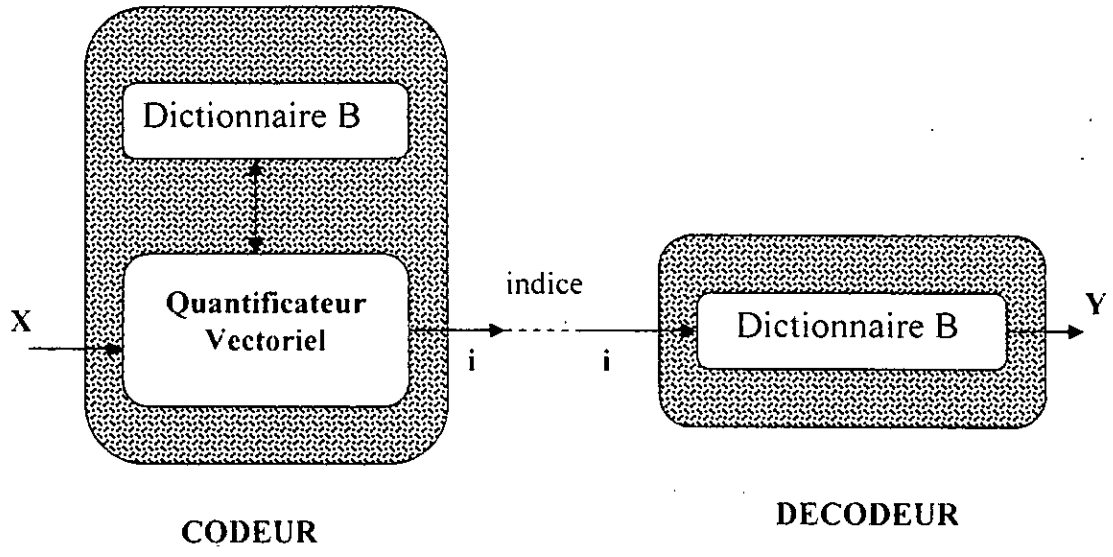
La séquence {e[n] ; n = 1,2,.....N} définie par : $e[n] = x[n] - \hat{x}[n]$ est dite erreur de prédiction. Les coefficients de pondération {a[i] ; i = 1,2,.....N} peuvent être calculés en minimisant l'erreur quadratique moyenne au sens des moindres carrés.

La méthode prédictive appliquée à un signal image permet d'établir une estimation de la valeur de chaque pixel à partir des pixels déjà traités, puis coder l'erreur de prédiction à l'aide d'un codeur entropique. En recevant le mot de code, le décodeur l'ajoute à la valeur calculée par son propre prédicteur et restitue le pixel correspondant.

Méthode de codage par Quantification Vectorielle :

Un quantificateur vectoriel (QV) fait correspondre à tout vecteur x décrit comme suit : x = (x(0), x(1), ..., x(k-1)) un vecteur y_i = (y_i(0), y_i(1), ..., y_i(k-1)) choisi parmi un ensemble fini B de N vecteurs de reproduction. Dans la littérature, on attribue à cet ensemble l'appellation de dictionnaire.

En pratique, dans plusieurs systèmes de communication, seul l'indice i de la séquence choisie y_i est transmis au décodeur qui choisit alors la séquence correspondante à cette valeur de l'indice. Le dictionnaire qui influe considérablement sur les performances techniques d'un QV est caractérisé par sa taille N et sa dimension k, ces paramètres permettent de déterminer le débit binaire $R = \log_2 N/k$ par échantillon **[12][16]**.
La figure I.4 schématise un exemple d'organisation structurelle d'un quantificateur vectoriel souvent utilisé dans les systèmes de communication numérique.



FigureI.4. Schéma d'un quantificateur vectoriel.

L'application directe de la quantification vectorielle dans la compression d'images consiste à diviser en premier lieu l'image en vecteurs de dimensions bien déterminées. Le vecteur erreur résultant, obtenu de la différence entre le vecteur x et sa moyenne sera ensuite quantifiée vectoriellement **[17]**.

- **Méthodes par transformée :**

Ce sont les techniques de compression basées sur la quantification des coefficients résultants d'une transformation linéaire. Le principe du codage par transformée consiste à décomposer chaque bloc d'échantillons du signal à une séquence de coefficients non corrélés **[13][14][18]**.

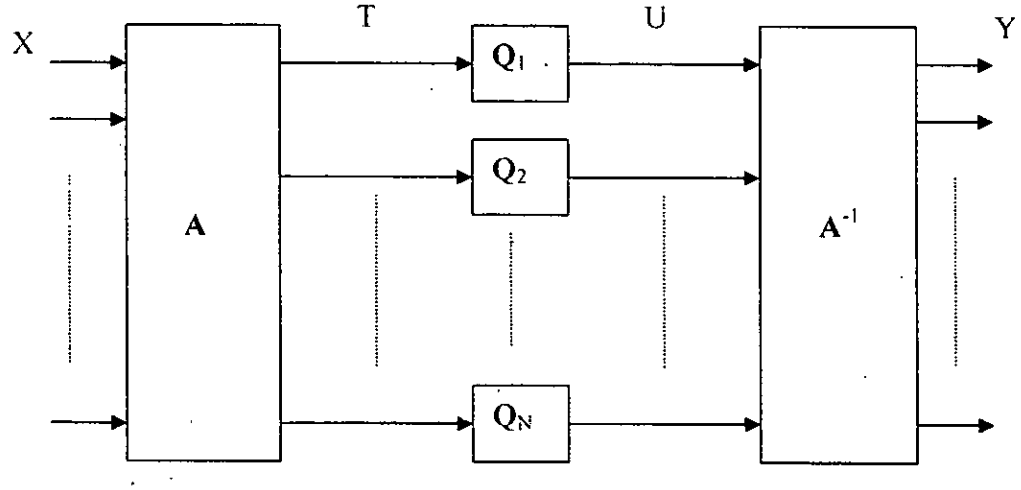


Figure I.5 : Schéma bloc d'un codeur par transformée.

La compression par transformation orthogonale s'adapte bien au partitionnement des images par bloc N×N pixels (pratiquement N = 8 ou 16). Chaque bloc subit une transformation orthogonale à deux dimensions. Le résultat de la transformation d'un bloc représente son spectre à deux dimensions. Si la transformation est optimale, l'énergie est concentrée dans les composantes orthogonales basses fréquences.

Le plan transformé contient peu de valeurs numériques élevées, donc il est plus intéressant de ne transmettre que les quelques valeurs du plan transformé plutôt que toutes les valeurs du bloc original. La technique de la compression d'images est fondée sur ce principe. La compression s'effectue grâce à l'application d'une quantification qui fait apparaître une grande quantité de valeurs nulles qui correspondent à la mise à zéro de toutes les valeurs inférieures en valeur absolue au pas de quantification. Cette quantification entraîne une perte d'information qui se traduit par une dégradation de la qualité de l'image restituée après transformation inverse.

IV. Standardisation : [8][19]

La norme JPEG est conçue par le groupe ISO (International Standards Organisation) et le groupe CEI (Commission Electrothechnique Internationale). Elle est destinée pour la compression des images en couleurs et à niveaux de gris en vue de leur stockage sur les supports numériques et leur transmission à travers un réseau à faible débit.

Les techniques définies par la norme JPEG se divisent en deux classes : les méthodes de compression qui sont basées sur la transformée en cosinus discrète **[20]** suivie d'une quantification et d'un codage entropique. La seconde classe est basée sur le codage DPCM (Differenciel Pulse Coded Modulation) suivie d'un codage entropique.

V. Evaluation de la Compression

La complexité des méthodes de compression d'images varie suivant le domaine d'application. Ces méthodes répondent chacune à des contraintes particulières imposées par l'application étudiée. L'évaluation de la compression peut être donnée par plusieurs paramètres, parmi eux, on cite : **[4][8]**

$$RapportdeCompression = Rapport_c = \frac{Taille\ des\ données\ Avant\ la\ compression}{Taille\ des\ données\ Aprés\ La\ compression}$$

$$Taux\ de\ Compression = \left(1 - \frac{1}{Rapport_c}\right) \times 100\%, exprimé\ en\ pourcentage.$$

VI. Qualité visuelle de l'image : [4]

Certaines techniques de compression d'images apportent des distorsions aux images reconstruites après décodage. Dans ce cas, il est nécessaire de comparer la qualité de l'image codée après construction à celle de l'originale pour juger la méthode. La qualité de l'image peut se mesurer de deux manières :

- Par des méthodes subjectives : Ces méthodes doivent prendre en compte les propriétés imprécises du récepteur visuel humain. Elles consistent à analyser psychovisuellement les images originales puis les images reconstruites.
- Par des méthodes objectives : La distorsion qui a été introduite lors du processus de codage-décodage correspond à l'erreur RMS (Root Mean Square) entre l'image originale et l'image reconstruite, donnée par :

$$RMS = \frac{1}{n}\sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n}\left(I_1(i,j) - I_2(i,j)\right)^2} \qquad (I.6)$$

tel que :

$I_1(i, j)$ et $I_2(i, j)$ sont les valeurs du pixel (i, j) dans les deux images I_1, I_2

n : est la taille des images.

Une autre mesure peut être utilisée pour évaluer la qualité objective de l'image qui est le rapport signal sur bruit (SNR) définit par :

$$SNR = 10\log_{10}\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\left[I_1(i,j)\right]^2}{\sum_{i=1}^{n}\sum_{i=1}^{n}\left[I_1(i,j) - I_2(i,j)\right]^2} \qquad (I.7)$$

Le signal sur bruit de crête noté PSNR (Peak Signal to Noise Ratio), plus souvent utilisé à la compression :

$$PSNR = \frac{(V_{max})^2}{\sum_{i=1}^{n}\sum_{i=1}^{n}\left[I_1(i,j) - I_2(i,j)\right]^2} \qquad (I.8)$$

où V_{max} est l'intensité maximale (255 pour une représentation 8bits/ pixel).

Chapitre 2

Théorie des Fractales

Introduction:

Lorsque l'on observe certains objets de la nature tels que les arbres, les nuages ou les montagnes, on remarque qu'ils sont constitués de parties élémentaires qui se répètent à des différentes échelles. En géométrie classique, ces objets sont modélisés d'une manière approchée grâce à des primitives simples telles que les cylindres, les segments de droites ou des sphères. La géométrie fractale qui est une extension de la géométrie classique, fournit des outils mathématiques permettant la modélisation des objets naturels. Cette géométrie a été relancée d'intérêt suite aux travaux du mathématicien français **MANDELBROT [5]**, depuis la fin des années 70.

Dans ce chapitre nous rappelons les lignes directives de la théorie des fractales. Nous donnons les définitions élémentaires et les notions de base qui caractérisent les objets fractales.

I. Théorie des Fractales :

I.1. Notion de fractale :

Le terme fractal est issu du latin fractus qui signifie fractionné à l'infini. Une définition à la fois précise et générale d'un objet fractal est difficile. On peut définir la fractale comme un ensemble géométrique qui présente des irrégularités à toutes les échelles. Une structure fractale est la même de prés comme de loin **[22]**.

Tout objet fractal possède les caractéristiques suivantes: **[23]**

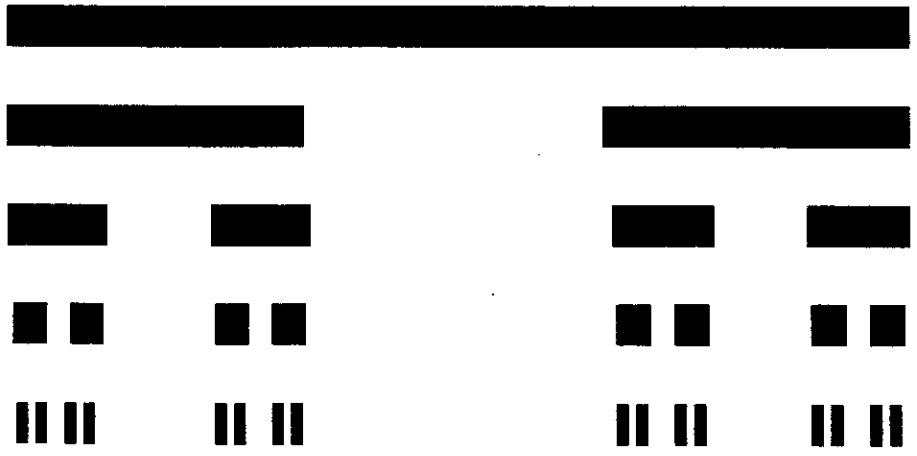
- Ses parties ont la même forme ou la structure que le tout.
- Sa forme est soit extrêmement irrégulière, soit extrêmement interrompue ou fragmentée.
- Il contient des éléments distinctifs dont les échelles sont très variées couvrant une très large gamme.
- Sa dimension fractale est supérieure à sa dimension topologique.
- Il existe une simple description algorithmique pour cet objet.

I.2. Ensembles Irréguliers :

Les ensembles irréguliers fournissent une meilleure représentation d'objets fractals. Ces ensembles complexes avaient déjà été imaginés par les mathématiciens (Cantor, Peano...), il y a plus d'un siècle, avant l'introduction de la notion de fractale. Dans cette section, nous citons les figures plongées dans un espace à deux dimensions **[23][24] :**

L'ensemble de CANTOR :

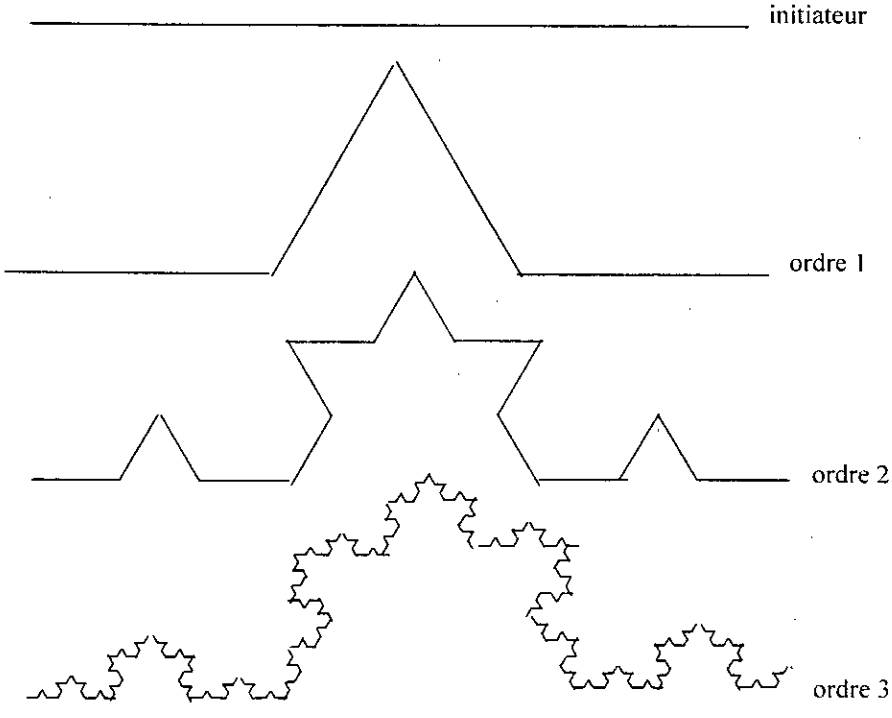
C'est un exemple important en topologie et en analyse. Pour le construire, nous prenons un segment de droite représentant les nombres compris entre zéro et un. Nous lui otons le tiers du milieu. Nous obtenons deux segments dont nous enlevons à nouveau les tiers du milieu (de un neuvième à deux neuvièmes et de sept neuvièmes à huit neuvièmes). Cela nous donne alors quatre segments dont nous enlevons les tiers du milieu, et ainsi de suite jusqu'à l'infini.



FigureII.1. Construction des quatre premières itérations de l'ensemble de Cantor

La courbe de Von KOCH :

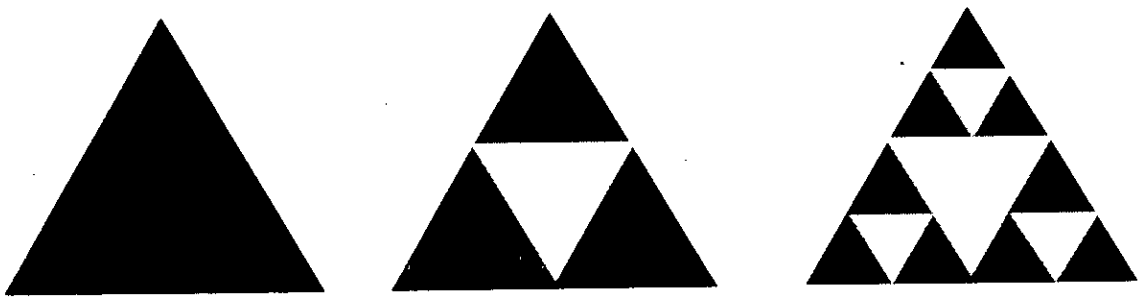
Elle constitue un bon exemple d'objets fractals. Etant donné le segment [0,1], on le remplace par le générateur formé de quatre segments de longueur 1/3 aux points (0, 0), (1/3, 0), (1/2, $(1/2)^{1/2}$), (2/3, 0), (1, 0) (figureII.2). La récurrence consiste à remplacer chacun de ces quatre segments de la même forme mais plus petite dans un rapport d'un tiers. A l'ordre n, la courbe est constituée de 4^n segments de longueur $(1/3)^n$. Continuant ainsi à l'infini, La longueur totale de la courbe tend vers l'infini. On aboutit à une figure célèbre dont la courbe limite ne possède de tangente nul part.



FigureII.2. Courbe de Von Koch.

Le tapis de SIERPINSKY :

Pour le construire, on prend un triangle équilatéral rempli, on enlève ensuite le triangle central dont les sommets sont les milieux des trois cotés du triangle initial. On répète la procédure pour les trois triangles restants et ensuite pour les neufs suivants et ainsi de suite.



FigureII.3. Tapis de **SIERPINSKY** pour les deux premières itérations.

I.3. Classification des Fractales :

Il existe deux grandes classes d'objets fractals : Les fractales géométriques et les fractales stochastiques ou naturelles.

I.3.1. Les fractales géométriques :

Ces fractales sont construites d'une façon déterministe sans faire intervenir le hasard. Partant d'une figure de base, qui est l'ordre zéro de la fractale (Initiateur) et d'une certaine figure que nous appellerons la graine de la fractale (Générateur). On ajoute ou l'on remplace certaines parties de l'initiateur par une image semblable au générateur. On obtient alors l'ordre un de la fractale. Il suffit de recommencer avec l'ordre un pour obtenir l'ordre deux et ainsi de suite. La figure obtenue à l'ordre infini, si elle existe, est une fractale.
Selon la manière dont les fractales géométriques sont construites, nous distinguons trois types : **[21][22][23]**

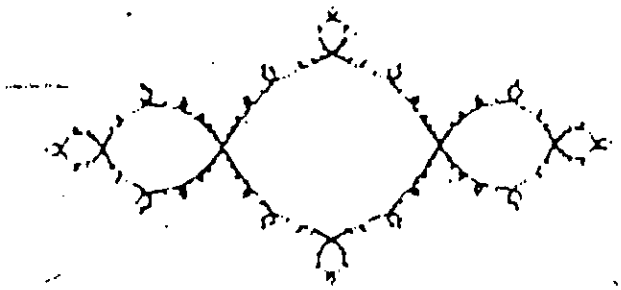
Les fractales Particulaires :

Les fractales particulaires sont constituées entièrement de points, elles sont obtenues à partir de suites récurrentes de forme : $x_{n+1} = F(x_n, y_n)$ et $y_{n+1} = G(x_n, y_n)$.
si par exemple, on pose :

$$\begin{cases} (x_0, y_0) = (0,0) \\ x_{n+1} = \sigma\left(r_n + (x_n - \alpha)/2\right)^{1/2} \\ y_{n+1} = \pm\sigma\left(r_n + (x_n - \alpha)/2\right)^{1/2} \end{cases} \qquad (II.1)$$

avec : $r_n = \left((x_n - \alpha)^2 + (y_n - \beta)^2\right)^{1/2}$; α et β sont des constantes fixes.
$\pm$ désigne le signe de $(y_n - \beta)$
σ est aléatoirement égale à $+1$ ou -1.

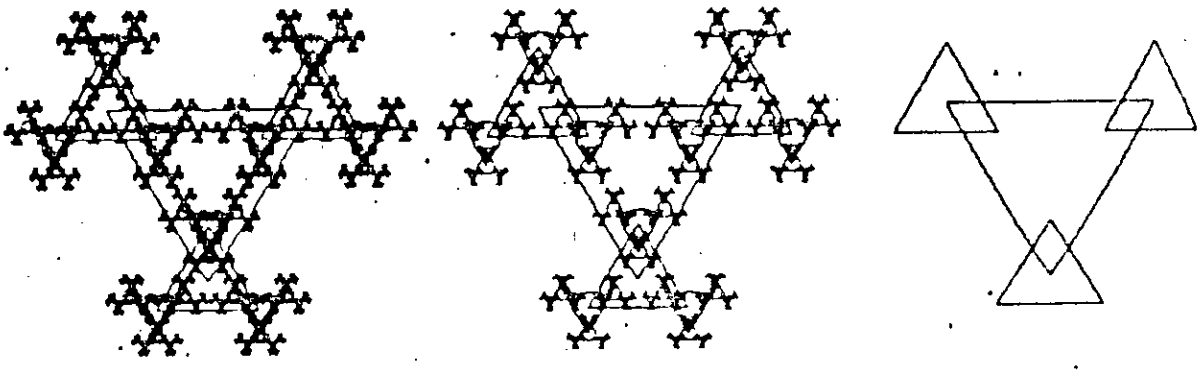
Au bout de quelques itérations, la suite ainsi obtenue donne des points situés à la frontière d'un certain ensemble appelé ensemble de JULIA. Cette frontière est appelée la courbe de JULIA (figureII.4).



FigureII.4. Courbe de JULIA avec $\alpha = -1$ et $\beta = 0$.

Les Fractales en Cascades :

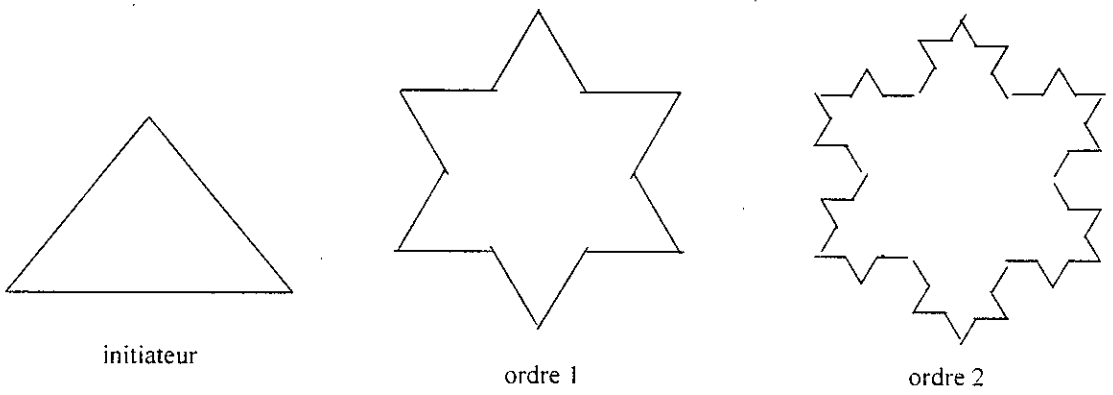
Ces fractales sont déduites d'une forme géométrique simple, dessinée un nombre de plus en plus important de fois entre deux ordres successifs, mais de plus en plus petite. Donnons un exemple basé sur des triangles équilatéraux. Partant d'un tel triangle, on dessine centré sur chacune de ses pointes, un triangle semblable tourné vers le centre du triangle initial et deux fois plus petit. Puis on recommence pour chacun des trois triangles ainsi obtenus. A l'ordre n, le nombre de triangles tracés au total est : $T_n = 1+3+3^2+.....3^n$.



FigureII.5. Cascade des triangles.

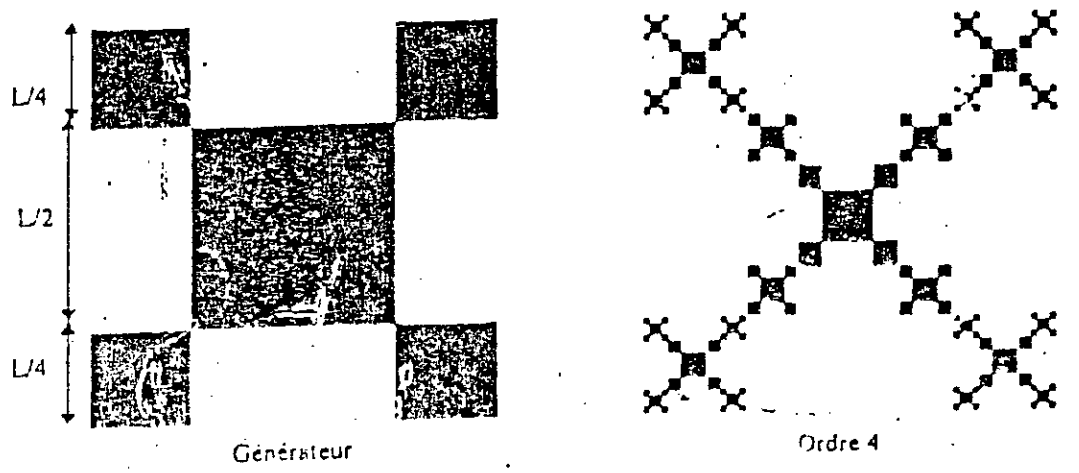
Les Fractales Linéaires :

Elles sont composées à tout ordre fini de segments de droite. Chaque ordre se déduisant du précédent en remplaçant tout ou partie de ces segments de droite, par une ligne polygonale semblable à une ligne fixée au départ. La courbe de Von KOCH constitue un bon exemple de ces fractales. Une structure dérivée est obtenue en choisissant un triangle équilatéral comme initiateur, on génère ainsi la très connue île de Von KOCH ou courbe flocon de neige.



FigureII.6. Flocon de neige.

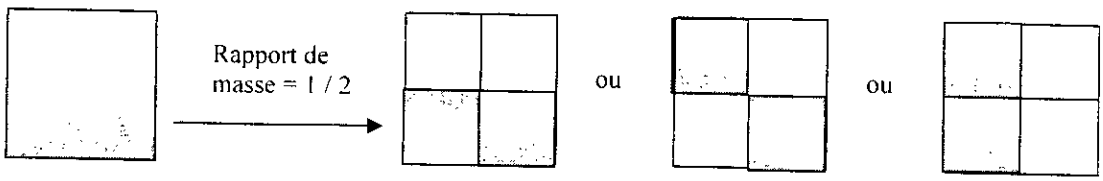
Une autre variation possible dans les structures fractales géométrique réside dans la présence simultanée de plusieurs échelles de dilatation. La figure II.7 est un exemple d'une telle structure obtenue par itération déterministe contenant les facteurs 1/ 4 et 1/ 2. Sa géométrie est évidemment fractale et sa structure est en réalité plus complexe dans la distribution de son support que les fractales décrites plus haut : elle est en fait multifractale.



FigureII.7. Fractale non Uniforme.

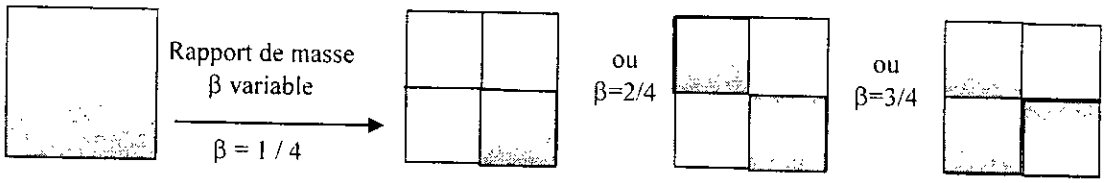
I.3.2. Les Fractales Stochastiques:

Ces fractales se distinguent des fractales géométriques par leur forme qui rappelle certains éléments de la nature. Dans les structures stochastiques la récurrence, définissant la hiérarchie, est régie par une ou plusieurs lois de probabilités précisant le choix de l'application de tel ou tel générateur à chaque itération. Ces fractales stochastiques peuvent être soit homogènes, soit hétérogènes. Les fractales stochastiques homogènes se caractérisent par la masse de la structure qui est répartie de façon uniforme à chaque niveau de hiérarchie, c'est à dire que les divers générateurs, servants à construire la fractale conservent le rapport de masse d'un niveau au suivant.



FigureII.8 Générateur d'une fractale stochastique homogène.

Les fractales stochastiques hétérogènes quant à elles se caractérisent par un rapport de masse qui fluctue au sein même d'une hiérarchie. Les fractales stochastiques sont à quelques exceptions remarquables prés, les seules rencontrées dans la nature. La génération d'une montagne est un exemple typique et particulièrement simple de fractales naturelles



FigureII.9. Générateur d'une fractale stochastique hétérogène.

I.4. Notion d'Auto- similarité : [22]

La théorie des fractales modélise la nature de certaines courbes, et plus généralement de certains objets en y reconnaissant entre autre, une auto- similarité. Un objet est auto-similaire lorsqu'il est constitué d'objets identiques à lui-même après d'éventuelles transformations simples et cela quelle que soit l'échelle d'observation.

De nombreux exemples peuvent être donnés, l'arbre en est un. En effet, à mesure que l'on s'en approche, on peut distinguer des branches maîtresses qui globalement ressemblent à l'arbre lui-même, mais plus petites et disposées dans une direction principale qui n'est plus nécessairement verticale. Si l'on s'approche davantage, on constate qu'une branche maîtresse est elle-même constituée de branches qui ressemblent à nouveau à l'arbre mais en dimensions réduites, et selon des directions variées et ainsi de suite.

I.5. Notion de la Dimension Fractale :

La notion de dimension fractale était connue dans le début du siècle (Hausdorff-Beescovitch). Des figures géométriques avaient bien vu le jour. La dimension fractale est un concept métrique qui mesure le degré d'irrégularité ou de brisure de l'objet fractal. Cette dimension peut ne pas être entière. La dimension fractale d'un objet traduit la manière dont il occupe l'espace. Une simple courbe Euclidienne à une dimension n'occupe aucun espace, mais la courbe de Von KOCH, avec sa longueur infinie contenue à l'intérieur d'une surface finie, remplit l'espace, sa dimension est supérieure à un mais inférieure à deux **[24]**. Plusieurs définitions de dimensions fractales ont été proposées depuis le début du siècle, certaines ont un intérêt purement théorique, d'autres sont plus faciles à calculer numériquement **[Annexe A]**.

II. Géométrie des Fractales :

II.1. Fondements mathématiques :

Les outils mathématiques sur les quels a été fondé la géométrie des fractales sont : la théorie de l'espace, les notions de topologie et la théorie d'itération.

Quelques définitions dans l'espace : [22]

Définition d'un espace Métrique :

Un espace métrique est un ensemble X sur lequel une fonction de distance à valeurs réelles $d : X \times X \longrightarrow R$ est définis, avec les propriétés suivantes :

- $d(a, b) \geq 0, \forall a, b \in X.$
- $d(a, b) = 0$, si et seulement si $a = b, \forall a, b \in X.$
- $d(a, b) = d(b, a), \forall a, b \in X.$
- $d(a, c) \leq d(a, b) + d(b, c), \forall a, b, c \in X.$

Définition d'une Suite de Cauchy :

Une suit $\{x_n\}$ d'éléments d'un espace métrique X est dite suite de Cauchy, si à tout nombre réel $\varepsilon > 0$, on peut associer un entier naturel n, tel que pour tout couple d'entiers p et q supérieur à n, on a :

$$d(x_p, x_q) < \varepsilon$$

Définition d'un espace métrique complet :

On appelle espace métrique complet, un espace métrique dans lequel toute suite de Cauchy est convergente.

II.2. Transformations affines :[21]

Transformations affines Dans R:

Les transformations affines dans R sont des transformations $w: R \rightarrow R$ de la forme :

$$w(x) = a.x + b \quad , \forall a, b \in X$$

où a et b sont des constantes réelles.

Transformation affine Dans le Plan Euclidien R^2 :

Une Transformation $w : R^2 \rightarrow R^2$ de la forme : $w(x, y) = (a.x + c.y + e, b.x + d.y + f)$, où a, b, c, d, e, et f sont des nombres réels, est appelée transformation affine bi- dimensionnelle. On peut utiliser la notation matricielle :

$$w\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a & c \\ b & d \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix} = Mat.\begin{bmatrix} x \\ y \end{bmatrix} + Vect$$

avec : $Mat = \begin{bmatrix} a & c \\ b & d \end{bmatrix}$ et $Vect = \begin{bmatrix} e \\ f \end{bmatrix}$

Mat, peut toujours s'écrire sous la forme :

$$Mat = \begin{bmatrix} r_1 \cos\theta_1 & -r_2 \sin\theta_2 \\ r_1 \sin\theta_1 & r_2 \cos\theta_2 \end{bmatrix}$$

où (r_1, θ_1) sont les coordonnées polaires du point (a, c) et $(r_2, \theta_2 + \frac{\pi}{2})$ les coordonnées polaires du point (b, d), avec :

$r_1 = \sqrt{a^2 + b^2}$, $\theta_1 = \text{arctg}(c/a)$ si $a \neq 0$, $\theta_1 = \frac{\pi}{2}$ si $a = 0$ et $c \geq 0$, $\theta_1 = \frac{3\pi}{2}$ si : $a = 0$ et $c < 0$.

$r_2 = \sqrt{b^2 + d^2}$, $\theta_2 = \text{arctg}(b/d)$ si $d \neq 0$, $\theta_2 = \frac{\pi}{2}$ si $d = 0$ et $b \geq 0$, $\theta_2 = \frac{3\pi}{2}$ si : $d = 0$ et $b < 0$.

Les paramètres a, b, c, d produisent une rotation et un changement d'échelle, et les paramètres e et f définissent les facteurs de translation du point sur lequel on opère.

Une transformation affine $w : R^2 \rightarrow R^2$ est appelée similitude si elle possède l'une des formes spéciales suivantes :

$$w\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} r\cos\theta & -r\sin\theta \\ r\sin\theta & r\cos\theta \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix}; \qquad w\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} r\cos\theta & r\sin\theta \\ r\sin\theta & -r\cos\theta \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix}$$

θ est appelée l'angle de rotation et r est appelé facteur d'échelle ou rapport de similitude.

Transformations affines dans R^3 :

La transformation affine $w : R^3 \rightarrow R^3$ peut s'exprimer en notation matricielle :

$$w\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} a & b & t \\ c & d & u \\ r & s & p \end{bmatrix}\begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} e \\ f \\ q \end{bmatrix} = Mat.\begin{bmatrix} x \\ y \\ z \end{bmatrix} + Vect$$

où a, b, c, d, e, f, p, q, r, s, t, u sont des constantes réelles et $(x, y, z) \in R^3$, donc :

$$w(x, y, z) = (a.x + b.y + t.z + e,\ c.x + d.y + u.z + f,\ r.x + s.y + p.z + q).$$

II.3 Transformations Contractantes :[21][23]

Défintion1 :

Une Transformation f : X → X dans un espace métrique (X, d) est appelée contractante, s'il existe une constante 0 < s <1, tel que : $d(f(x),f(y)) \leq s.d(x,y)$; $\forall (x,y) \in X^2$.
s : est appelé facteur de contraction de f.

Définition2 :

Soit f : X → X une transformation dans un espace métrique. Les itérations successives de f sont les transformations : $f^{\circ n} : X \to X$ définies par :

$f^{\circ 0}(x) = x; f^{\circ 1}(x) = f(x); \ldots\ldots\ldots\ldots f^{\circ(n+1)}(x) = f \circ f^{n} = f(f^{\circ n}(x)).$ $n = 0,1,2,\ldots\ldots\infty.$

Théorème du point fixe :

Ce théorème formalise une constatation intuitive. Si une fonction est contractante, alors lorsqu'on l'applique de façon récurrente à partir d'un point initial on converge vers un unique point fixe.
Autrement- dit: si f : X → X une fonction contractante dans un espace (X, d), alors f possède un point fixe $x_f \in X$ tel que $f(x_f) = x_f$, et de plus pour chaque point $x \in X$, la séquence : $\{f^{\circ n}(x) ; n = 1, 2, \ldots\ldots\infty\}$ converge vers x_f, c'est à dire :

$$\lim_{n\to\infty} f^{\circ n}(x) = f(x_f) = x_f, \quad \forall x \in X.$$

Chapitre 3

Théorie des IFS et Compression d'images par fractales

Introduction:

La compression d'images par fractale est l'une des applications les plus récentes de la géométrie fractale. La modélisation mathématique de cette technique, sous l'appellation d'IFS (Iterated Functions System), a été proposée par le mathématicien **HUTCHINSON [25]** au début des années 80. Les IFS regroupent toute une classe de transformations dont les itérées successives ont la propriété de converger vers un élément unique, appelé attracteur de l'IFS.

I. Théorie des IFS (Iterated Functions System) : [22]

Définition 1 : IFS

Un système de fonctions itérées (IFS), est constitué d'un espace métrique complet (X, d) avec un ensemble fini de transformations contractantes, $w_i : X \rightarrow X$ dotées respectivement d'un facteur de contraction s_i, i = 1, 2, ...N. La notation pour cet IFS est :

$\{X, w_i ; i = 1, 2, ...N\}$. Son facteur de contraction est donné par :

$$s = \max\{s_i ; i = 1, 2, ...N\}$$

Théorème 1: Théorème de l'IFS

Soit $\{X, w_i ; i = 1, 2, ...N\}$ un système de fonctions itérées avec un facteur de contraction s et soit $E \subset X$ un sous-ensemble de X, alors la fonction W définie par : $W(E) = \bigcup_{i=1}^{N} w_i(E)$ pour tout E, est une fonction contractante. Elle possède un point fixe unique, $A \subset X$ qui obéit à :

$$A = W(A) = \bigcup_{i=1}^{N} w_i(A) = \lim_{n \to \infty} W^{\circ n}(E) \text{ pour tout } E \subset X. \qquad (III.1)$$

Le point fixe $A \subset X$ est appelé **l'attracteur** de l'IFS.

Exemples des IFS :

Exemple 1: Tapis de SIERPINSKI

Le Tapis de SIERPINSKI est généré par les trois transformations suivantes :

$$w_1\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix}, \quad w_2\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0.5 \\ 0 \end{bmatrix}, \quad w_3\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 0.5 & 0 \\ 0 & .0.5 \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}$$

Pour chaque transformation w_i, on affecte une probabilité P_i, avec une importance relative aux autres transformations.

si: $$w_i\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a_i & c_i \\ b_i & d_i \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix}$$

La probabilité P_i est donnée par la relation suivante :

$$P_i = \frac{|a_i d_i - b_i c_i|}{\sum_{i=1}^{N} |a_i d_i - b_i c_i|} \quad \text{pour } i = 1, 2, \ldots\ldots\ldots N \qquad \text{(III.2)}$$

avec, $\sum_{i=1}^{N} P_i = 1$

La notation pour un IFS, en utilisant des matrices est inadaptée.

C'est ainsi l'IFS $\{R^2, w_1, w_2, w_3\}$ peut s'exprimer par la table suivante :

Table III.1 : Le code IFS du tapis de SIERPINSKI :

W	A	B	C	D	e	f	P
1	0.5	0	0	0.5	0	0	0.33
2	0.5	0	0	0.5	0.5	0	0.33
3	0.5	0	0	0.5	0.5	0.5	0.34

En itérant le code IFS à partir d'une image quelconque, on obtient l'attracteur de ce code qui est le tapis de SIERPINSKY (figure II.3).

Exemple2: Fougère de BARNSLEY

C'est un exemple classique d'IFS. En effet cette fougère peut être obtenue en itérant les transformations suivantes sur une image de départ quelconque.

$$w_1\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0.16 \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$w_2\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 0.85 & 0.04 \\ -0.04 & 0.85 \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0 \\ 1.6 \end{bmatrix}$$

$$w_3\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 0.2 & -0.26 \\ 0.23 & 0.23 \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0 \\ 1.6 \end{bmatrix}$$

$$w_4\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} -0.15 & 0.28 \\ 0.26 & 0.4 \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

FigureIII.1: Fougère de BARNSLEY

Théorème 2 : Théorème de collage

Soient : (X, d) un espace métrique complet, $E \subset X$, $\{X, w_1, w_2, \ldots\ldots\ldots w_N\}$ un IFS de facteur de contraction $0 < s < 1$, A l'attracteur de l'IFS et $\varepsilon \geq 0$.

si: $d(E, \bigcup_{i=1}^{N} w_i(E)) \leq \varepsilon$.

alors :

$$d(E,A) \leq \frac{\varepsilon}{1-s} \qquad \text{(III.3)}$$

avec, $d(E,A) = \max\{\min\{d(x,y) : y \in A\} x \in E\}$

Le théorème de collage nous donne une borne supérieure de l'erreur commise en approximant E par l'attracteur A de W. On peut ainsi s'attendre à une bonne approximation de E avec ε petit. Le facteur de contraction s contrôle la vitesse de convergence, les petites valeurs de s permettent une convergence rapide.

II. Construction des fractales IFS :

Les IFS permettent la génération d'images totalement auto- similaires, en utilisant un nombre restreint de transformations. Une fractale est construite à partir du collage de copies transformées d'elle-même et ainsi naturellement auto- similaire et infiniment échelonnée. Des méthodes simples pour générer des fractales définies par des codes de système de fonctions itérées (IFS) ont été développées par différents chercheurs. Nous allons présenter deux approches algorithmiques de calcul de fractales **[6] [21]**.

II.1. Algorithme à Itération aléatoire :

Cet algorithme offre une méthode simple pour construire une approximation grossière de l'attracteur. Soit : $\{X, w_k ; k = 1, 2, ...N\}$ un IFS, où une probabilité $P_k > 0$ est affectée à w_k, pour k = 1, 2, ...N. On choisit arbitrairement un point initial $x_0 \in X$, ensuite on calcule les points x_i successifs, en appliquant les fonctions choisies aléatoirement à partir de l'IFS. On peut montrer que les points approchent progressivement de l'attracteur indépendamment du point de départ. Donc il est pratiquement commun d'ignorer les premières itérations et supposer que les points suivants sont dans ou très proches de l'attracteur.

L'algorithme peut être décrit par le pseudo- code :

Choisir un point initial x_0
Pour i = 1 à max. itérations faire
Début
Choisir k aléatoirement parmi {1,2,.....N}
Calculer $x_i = w_k(x_{i-1})$
Si i > min – itérations alors tracer x_i
Fin

Le choix de k est fait suivant la probabilité que l'on assigne à chaque transformation w_k.

II.2. Algorithme à Itération déterministe :

Cet algorithme applique les itérations sur des ensembles de points plutôt que sur des points singuliers. Il commence avec un ensemble de points quelconque et applique chaque fonction de l'IFS $\{X, w_k ; k = 1, 2, ...N\}$ à cet ensemble. Tous les nouveaux points générés deviennent les points de départ pour la prochaine itération.

Le pseudo- code de l'algorithme est donné par :

Choisir un ensemble de points initial X_0
Pour i = 1 à max- itérations faire
Pour k = 1 à N faire
Pour chaque point $x \in X_{i-1}$ *faire*
$X_{i+1} = X_i \cup w_k(x_i)$
Fin de x
Fin de k
Fin de i

Après un certain nombre d'itérations, les ensembles X_i deviennent arbitrairement très proche de l'attracteur, indépendamment de l'ensemble initial.

Notons que les fractales exactes contiennent une infinité de points et en principe, chacun des algorithmes proposés peut les calculer en un temps infini. En pratique, on se contente seulement d'un ensemble représentatif de points.

III. Compression d'images fixes par IFS:

Le Principe de la méthode de compression d'images fixes par IFS est l'exploitation des auto- similarités dans les images. En effet au lieu de rechercher la corrélation entre pixels adjacents, on s'intéresse à des corrélations entre parties plus au moins espacées dans l'image.

Le système de codage/ décodage est basé sur la construction d'un ensemble de transformations contractantes définissant un système de fonctions itérées (code IFS). Le code IFS trouvé pour une image donnée est utilisé dans la construction de l'attracteur (la fractale), qui devra ressembler à l'image initiale. Les images codées, comme tous les objets fractales possèdent la propriété d'auto- similarité **[6]**.

La compression d'images par IFS est performante du point de vue taux de compression, car les transformations nécessitent un taux d'information assez faible comparé à celui des images originales. C'est une méthode de compression avec distorsion, du fait que l'attracteur ne constitue qu'une approximation de l'image originale. Toutefois, cette méthode est applicable seulement pour des images purement fractales, c'est à dire celles qui présentent des auto- similarités globales.

Pour les images naturelles, il s'agit de trouver les similarités permettant de les représenter sous forme de transformations contractantes. Les premières recherches tentaient de trouver des transformations globales dans le sens qu'elles agissent sur toute l'image, mais elles n'ont pas abouti vu que les images naturelles ne possèdent pas forcément la propriété d'auto- similarité globale. La solution proposée par **JACQUIN [26]** consistant à rechercher les auto- similarités locales ou partielles a conduit au premier algorithme de compression par IFS et à la notion des L-IFS (Local Iterated Functions System).

FigureIII.2: Similarités locales dans l'image de Lena.

IV. Les L-IFS (Local Iterated Functions System):

IV.1. Définitions:

Définition 3:

Une transformation w_i, définie dans un espace métrique complet X est dite localement contractante si : $\exists E \subset X$ telle que $w_i : E \rightarrow X$ est contractante.

Définition 4 : L-IFS

Un L-IFS W est un ensemble fini de transformations localement contractantes dans un espace métrique complet (X, d) : $W = \{w_i : E_i \to X, E_i \subseteq X, i = 1,\ldots N\}$

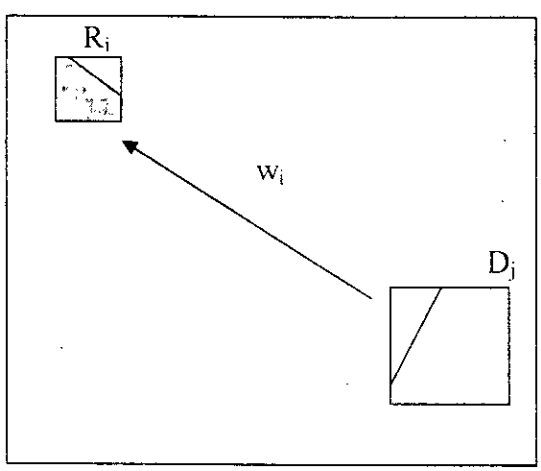
avec : $W\left(\bigcup_{i=1}^{N} E_i\right) = \bigcup_{i=1}^{N} w_i(E_i)$

IV.2. Compression d'images fixes par Les L-IFS (Fractales):

IV.2.1. Principe de la compression par L-IFS: [26][27][28][29]

L'idée de base de la compression d'images fixes à base de la géométrie fractale est inspirée du théorème de collage. Elle consiste à trouver des transformations localement contractantes w_i et leurs domaines de départ qui sont des parties de l'image, de façon à ce que l'image de ces domaines par les transformations soit assez proche de l'image de départ.

Ainsi l'algorithme de codage consiste à partitionner l'image à coder en des blocs R_i, usuellement des carrés de taille B, qui constituent les domaines d'arrivée des transformations et qui seront appelés "Range". On définit en suite un dictionnaire de domaines de départ de blocs D_j de l'image souvent des carrés de taille D = 2B. On les appellera blocs "Domain". Puis on définit un dictionnaire des transformations w_i. Pour chaque bloc "Range" R_i, il faut donc trouver une transformation w_i et un bloc D_j qui minimise la distorsion $d(R_i, w_i(D_j))$. Le code de l'image compressée est alors composé de l'ensemble de N transformations affines contractantes w_i, N étant déterminé par le type de partitionnement.
L'étape de décodage consiste en l'itération de l'ensemble de transformations à partir d'une image quelconque.



FigureIII.3: Collage d'un bloc Domain sur un bloc range.

IV.2.2. Structure et construction du code fractal :

Terminologie :

Ce vocabulaire spécifique à la compression par fractales est utilisé dans la plupart des articles traitant ce sujet. Malheureusement certains termes peuvent difficilement être traduit. C'est pourquoi tout au long des chapitres qui suivent, ils seront conservés avec la formulation anglaise.

Cellule : On appelle cellule un cadre carré de r × r pixels, repéré par les coordonnées de ses sommets dans l'image.

Bloc : On appelle bloc d'une image, la restriction de cette image à une cellule donnée.

Bloc- Domain : Un Bloc Domain est un bloc d'une image appartenant à un ensemble de blocs particulier.

Bloc- Range : Un Bloc Range est un bloc d'une image qui fait partie d'une partition de cette image.

Soit (X, d) un espace métrique d'images numériques, où d est une métrique donnée (mesure de distorsion). On souhaite coder une image numérique I_{orig} de l'espace (X, d) correspondant. Pour cela, on va rechercher à construire une transformation contractante τ de X vers lui-même, pour laquelle I_{orig} est une approximation d'un point invariant, qui est l'attracteur de cette transformation.
Notons F l'ensemble de ces transformations. Soit τ un élément de F et I_0 une image quelconque de X. Dans ce cas pour tout entier naturel n, on a **[26]**:

$$d(I_{orig}, \tau^n(I_0)) \leq (1/(1-s)) \bullet d(I_{orig}, \tau(I_{orig})) + s^n \bullet d(I_{orig}, I_0) \qquad \text{(III.4)}$$

Posons $I_n = \tau^n(I_0)$. Donc $\{I_n = \tau^n(I_0) \text{ tel que } n \geq 0\}$ reste au voisinage de I_{orig}, dans (X, d). Ce voisinage sera d'autant plus petit que n est grand (car s^n tend vers 0 quand n tend vers l'infini).

Notons que Le rapprochement de $\tau^n(I_0)$ à I_{orig} est conditionné par la distance $d(I_{orig}, \tau(I_{orig}))$. Plus précisément, si $d(I_{orig}, \tau(I_{orig})) = \varepsilon$, alors en passant à la limite dans (III.4), on aura :

$$d(I_{orig}, I_\infty) \leq \varepsilon/(1-s) \qquad \text{(III.5)}$$

où :

$I_\infty = \lim_{n\to\infty}\{I_n\}$

ε erreur du codage.

On construit τ tel que son attracteur représente l'image I_{orig}, Théoriquement, il faudrait que : $I_{orig} = \tau(I_{orig})$, mais en pratique, ceci s'avère rarement possible. Il faut donc lors du codage de l'image s'attacher à construire τ tel que ε soit minimal, et lors du décodage prendre n suffisamment grand pour approcher à l'attracteur au mieux. Dans ces conditions, pour n correctement choisi et pour n'importe quelle image initiale I_0, on a : $I_n \cong I_{orig}$.

La transformation τ est appelée **CODE FRACTAL** pour I_{orig}. Le code fractal est construit localement sur l'image à coder (le principe des L-IFS).

En résumé, l'étape du codage est basée sur la recherche d'une transformation τ définie par bloc dans l'espace des images. La transformation τ s'écrit sous la forme suivante :

$$\tau(I_{orig}) = \bigcup_{i=1}^{N} (\tau I_{orig})\big|_{R_i} = \bigcup_{i=1}^{N} \tau_i (I_{orig}\big|_{D_i})$$

où : $\{R_i,\ i = 1,\ldots,\ N\}$ décrit un partitionnement (sans chevauchement) du support de l'image en N régions carrées appelées cellules "Range" et qui peuvent être de différentes tailles.

$V|_{cel}$: représente le bloc d'une image V définie par la restriction de cette image à la cellule cel.

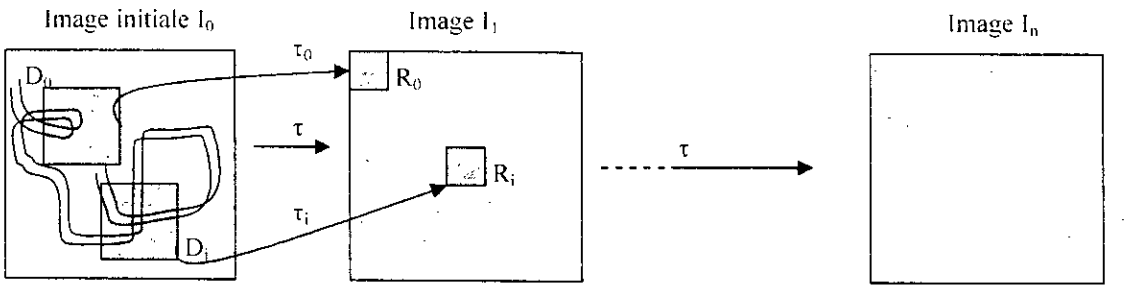
τ_i : est une transformation élémentaire contractante

La construction de la transformation τ, qui constitue le code fractal est effectué en définissant chacune des transformations élémentaires τ_i séparément est indépendamment des autres transformations. Le codage de chaque cellule "Range" revient à trouver une transformation τ_i est une cellule "Domain" D_j, à laquelle la transformation est appliquée, tel que le bloc "Domain" transformé $\tau_i(I_{orig}|_{Dj})$ sera une bonne approximation du bloc "Range" $I_{orig}|_{Ri}$. C'est à dire la distorsion $d(R_i, \tau_i(I_{Dj}))$ est minimale. L' $i^{ème}$ étape du codage de I_{orig} correspond, donc au codage de l' $i^{ème}$ bloc R_i dans l'image I_{orig}.

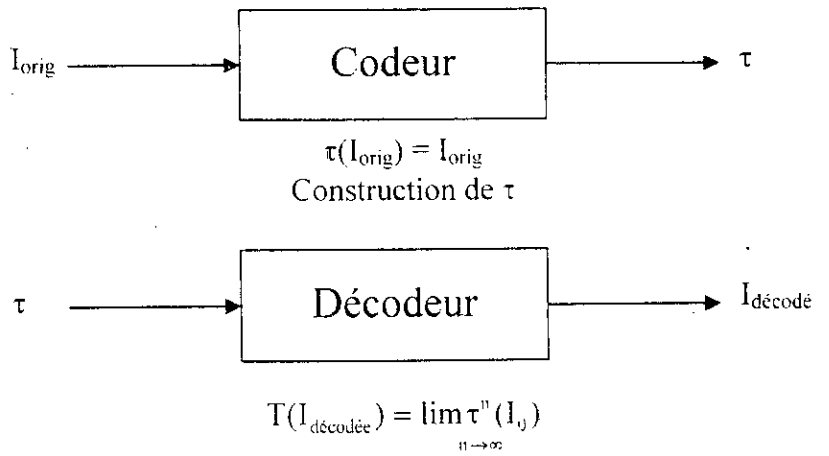
IV.2.3. Reconstruction Itérative de l'image:

La procédure de décodage consiste simplement à itérer la transformation τ sur n'importe quelle image initiale I_0, jusqu'à ce que la convergence vers une image finale stable soit observée.

On suppose donné la transformation τ décrite dans les paragraphes précédents. Donc si elle est correctement construite, $I_n = \tau^n(I_0)$ est de plus en plus proche de I_{orig} quand n augmente. On appelle $\{I_n / n \geq 0\}$ **la séquence de reconstruction** de l'image de I_{orig} pour le code fractal τ **[26][28]**. A chaque étape de reconstruction on applique τ_i, pour tout indexage i de cellule, au bloc "Domain" $I_{orig}|_{Dj}$ ce qui permet de connaître le bloc relatif à la cellule R_i.



FigureIII.4: Séquence de reconstruction.



FigureIII.5: Schéma général du système de codage fractal **[26]**.

IV.2.4. Conception d'un système de codage par fractales:

Les conditions de base pour la conception d'un système de codage d'images basé sur les fractales sont **[26]**:

- Le choix du partitionnement de l'image.
- Le choix de la mesure de distorsion
- Le choix des transformations fractales.

a) Le partitionnement de l'image :

Le support de l'image peut être partitionné de plusieurs façons et les partitions peuvent avoir plusieurs formes, l'essentiel c'est que l'ensemble des partitions couvrent toute l'image **[28]**.

Un Partitionnement carré:

C'est un partitionnement simple et facile à réaliser. Ce type de partitionnement est le plus utilisé par la majorité des schémas de codage par fractales.
Si la taille des blocs est choisie≤4*4, alors ces blocs sont :

- Faciles à analyser et a les classer.
- Faciles à coder et avec précision.
- Permettent une évaluation rapide pour le calcul de la distorsion inter- blocs.
- Rends le système plus efficace.

Si la taille des blocs est choisie≥ 5*5, alors :

- Ils permettent une grande exploitation de la redondance.
- Ils guident théoriquement vers un taux de compression élevé.

Un Partitionnement Quadtree :

L'inconvénient du partitionnement carré est l'utilisation des blocs de taille fixe. On trouve d'une part des régions de l'image difficiles à couvrir avec ce partitionnement. D'autre part, il y a des régions qui doivent être couvertes avec des ranges plus larges. Une généralisation du partitionnement à taille fixe est l'utilisation d'un partitionnement quadtree.

Un quadtree est une représentation de l'image sous forme d'un arbre quaternaire, où chaque nœud possède quatre fils. Ce partitionnement est rigide car il est réalisé par un découpage récursif en blocs carré. Il n'est pas adapté aux formes des objets de l'image même s'il s adapte bien au contenu de celle- ci.

Un Partitionnement triangulaire adaptatif :

Une autre manière de partitionner l'image est basée sur des triangles. Dans le schéma de partitionnement triangulaire, l'image est découpée récursivement on deux triangles, chacun à son tour est subdivisé en quatre triangles. Ce partitionnement est souple, car il est calculé sur un ensemble de points pouvant être positionnés à peu prés n'importe où, sur le support de l'image. Un autre avantage de ce partitionnement est qu'il exploite mieux les auto-similarités.

Partitionnement H-V (Horizontal- Vertical) :

Dans la partition H-V, l'image est découpée récursivement en deux rectangles soit horizontalement soit verticalement, et qui ne sont pas nécessairement de même taille. Ce partitionnement s'adapte au contenu de l'image et fait apparaître les auto- similarités entre régions. Il est flexible car la position de la partition est variable.

Partitionnement à base de Polygones :

L'image est partitionnée adaptativement en polygones. Cette technique permet une meilleure représentativité des objets formant l'image. Elle est flexible et permet de représenter les régions avec beaucoup de détails.

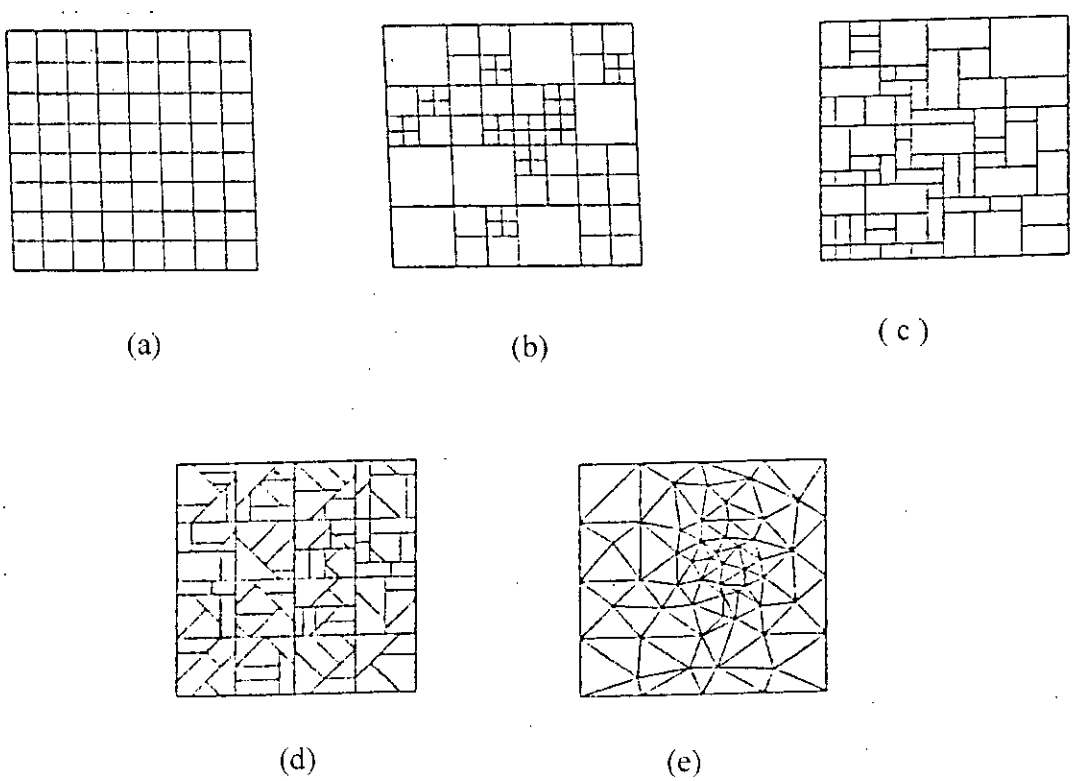


Figure III.6 : Différents types de partitionnement. (a) Carré. (b) Quadtree. (c) Horizontal- vertical. (d) Delaunay triangularisation. (e) Polygonale.

Le choix du partitionnement :

Bien évidement le type de partitionnement joue un rôle important dans la compression, et le taux change avec le changement du type et cela en minimisant chaque fois le nombre de transformations locales. En tenant compte de la variété et la complexité des images, il est aisé de voir que le partitionnement de l'image peut être d'une grande importance. C'est pour cela qu'on doit choisir de soit la partitionner selon sa complexité ou tout simplement partitionner toutes les images de la même manière, c'est à dire un partitionnement standard. Le choix de la forme carré est un choix naturel, car le partitionnement carré est facile à implémenter, alors que les autres sont dynamiques, c'est à dire qu'au fur et à mesure qu'on code, on divise l'image et cela prend beaucoup de temps. Le plus important est ce qui a guidé notre choix, c'est que le partitionnement utilise des petits blocs et donc peut couvrir toute l'image avec ses plus fins détails.

b) La mesure de distance (distorsion) :

La construction d'une fonction de distance ou une mesure de distorsion entre les images numériques, revient à celui d'une fonction de distance entre les blocs des images.
Soit $S(i_0, j_0, B)$, une cellule carrée de taille B×B, repérée par le sommet (i_0, j_0) qui représente l'intersection de la ligne i_0 et la colonne j_0.
Soit I une image de taille n×n, et $\tilde{I}$ une approximation de l'image I. Soit $I|_C, \tilde{I}|_c$ leurs restrictions aux cellules $S(i_0, j_0, B)$.
La L_2 distance (Mean Squared) distorsion entre les blocs est donnée par **[26]**:

$$d_{L_2}(I|_C, \tilde{I}|_c) = \sum_{i=1}^{n}\sum_{j=1}^{n}\left(I_{i0+i,j0+j} - \tilde{I}_{i0+i,j0+j}\right)^2 \qquad \text{(III.6)}$$

c) Les Transformations fractales :

Les transformations fractales sont dites itératives et se décomposent en deux parties, une partie géométrique et une partie massique. La forme générale d'une transformation fractale est :

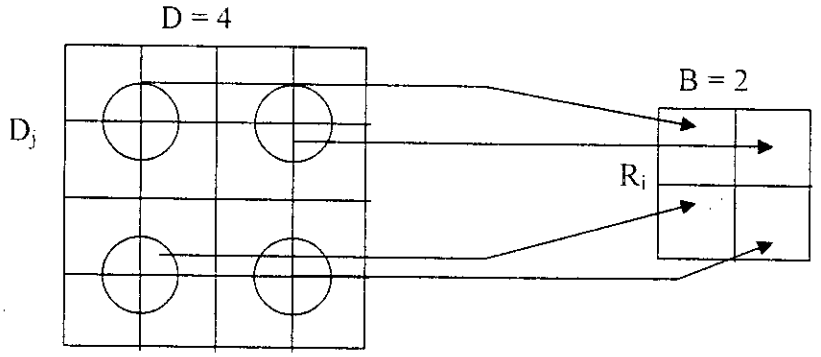
$$\tau = \sum_{i\in C}\tau_i = \sum_{i\in C} M_i \circ G_i \qquad \text{(III.7)}$$

où :
C ={1, ..., N} est l'ensemble des indices des blocs "Range", tel que N représente le nombre de blocs "Range" dans l'image originale.
G_i : est la transformation géométrique.
M_i : est la transformation massique.

c1) La partie géométrique G_i de la transformation: [26]

On considère des blocs "Range" de taille B×B, et des blocs "Domain" de taille D×D tel que D = 2B.
On décrit la forme discrète de la contraction spatiale G_i D : B, qui transforme un bloc "Domain" relatif à la cellule $S(i_d, j_d, D)$, en un bloc "Range" relatif à la cellule $S(i_b, j_b, B)$ afin de pouvoir les comparer avec la mesure de distorsion d_{L2}. Puisque D = 2B, cette transformation est une réduction de moitié du bloc. L'application G_i définie par

"moyennage", permet d'obtenir la valeur de chaque pixel de $G_i(S(i_d,j_d,D)$ en effectuant la moyenne des quatre pixels correspondants à la cellule $S(i_d,j_d,D)$.



FigureIII.7 : Transformation géométrique.

1	2	5	6
3	2	7	2
9	10	13	11
11	6	10	2

2	5
9	9

FigureIII.8 : Exemple de la transformation géométrique.

c2) La partie massique M_i de la transformation:

La partie massique est composée de deux transformations prises parmi deux classes de transformations particulières **[26]**.

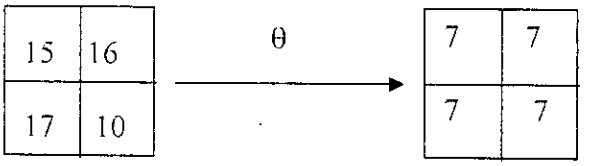
- **Première classe :** Opérations sur les valeurs des pixels

Les transformations de cette classe agissent seulement sur les valeurs des pixels. On trouve dans cette classe la transformation affine composée d'une translation et d'un échelonnage, aussi bien que les transformations d'absorption.

1. **Une absorption par g_0 :** La valeur du pixel (i, j) est affectée dans l'image I par la valeur $g_0 \in \{V_i\}_{0\le i<NGM}$. où NGM est le nombre des niveaux de gris pris en considération.

$$\forall (i,j) \in C, \quad (\theta I)(i,j) = g_0 \qquad \text{(III.8)}$$

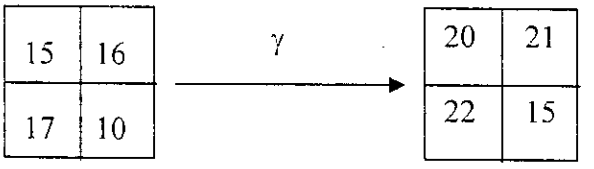
par exemple si $g_0 = 7$, on a les résultats suivants :



2. **Translation par Δg :** On ajoute à la valeur du pixel (i, j) une constante $\Delta g \in \{V_i\}_{0 \le i < NGM}$.

$$\forall i, j \in C, \quad (\gamma I)(i, j) = I(i, j) + \Delta g \tag{III.9}$$

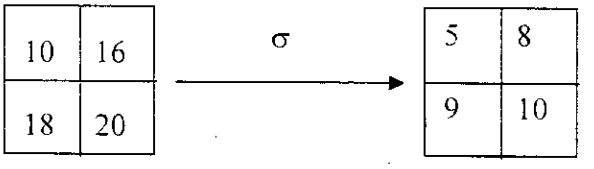
si $\Delta g = 5$, on a pour l'exemple suivant ce qui suit :



3. **Un Echelonnage (Scaling) par α :** La valeur du pixel (i, j) est multipliée par un scalaire $\alpha \in [0, 1]$.

$$\forall i, j \in C, \quad (\sigma I)(i, j) = \alpha . I(i, j) \tag{III.10}$$

Par exemple pour $\alpha = 0.5$, on obtient les résultats suivants :



- **Deuxième classe :** Opérations sur les positions des pixels (Les isométries):

Les transformations suivantes, ne modifient pas les valeurs des pixels. Simplement, elles les déplacent dans le bloc. Ces transformations sont appelées isométries. Il existe huit isométries.

Identité: Les positions des pixels du bloc ne changent pas.

$$\forall i, j \in C \qquad (i_0 I)(i, j) = I(i, j) \tag{III.11}$$

Réflexion par rapport à l'axe horizontal: (j = (B-1)/ 2)

$$\forall i, j \in C \qquad (i_1 I)(i, j) = I(i, B-1-j) \tag{III.12}$$

Réflexion par rapport à l'axe vertical: (i = (B-1)/ 2)

$$\forall i, j \in C \qquad (i_2 I)(i, j) = I(B-1-i, j) \tag{III.13}$$

Réflexion par rapport à la première diagonale: (i = j)

$$\forall i, j \in C \qquad (i_3 I)(i, j) = I(j, i) \tag{III.14}$$

Réflexion par rapport à la deuxième diagonale: (i + j = B-1)

$$\forall i, j \in C \qquad (i_4 I)(i, j) = I(B-1-j, B-1-i) \tag{III.15}$$

Rotation de 90° par rapport au centre de bloc :

$$\forall i, j \in C \qquad (i_5 I)(i, j) = I(j, B-1-i) \tag{III.16}$$

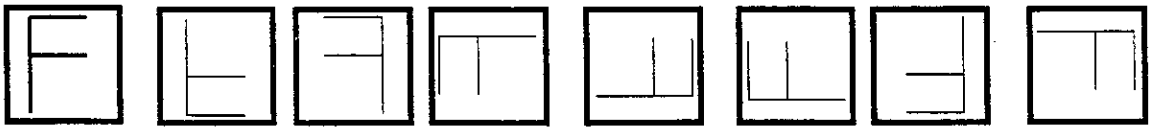
Rotation de180° par rapport au centre de bloc:

$$\forall i, j \in C \qquad (i_6 I)(i, j) = I(B-1-i, B-1-j) \tag{III.17}$$

Rotation de -90° par rapport au centre de bloc :

$$\forall i, j \in C \qquad (i_7 I)(i, j) = I(B-1-j, i) \tag{III.18}$$

La figure III.9 montre l'effet de l'application de chacune des isométries à un bloc carré représentant un "F".



FigureIII.9. Les Huit Isométries.

V. Optimisations : [28]

Dés 1989, de nombreuses recherches ont débuté, visant toutes à accélérer la phase de calcul des transformations locales, et / ou à répondre au compromis taux de compression-distorsion. Suite au travail de JACQUIN, L. Lundheim a très vite proposé une formulation algébrique visant à faciliter la compréhension des différents problèmes théoriques et pratiques soulevés par l'extension de la théorie des IFS au codage des images naturelles. Sa formulation appliquée au signal mono dimensionnel a ensuite été étendue au cas des signaux bidimensionnels. La notion d'optimisations peut être comprise de différentes manières :

- Accélération de la phase de codage : limitation du nombre de comparaison inter-blocs par classification ou quantification des blocs "Domain".
- Suppression des recherches de similarités inter-blocs.
- Accélération de la phase de décodage : orthogonalisation des vecteurs (blocs) "Domain" par rapport au vecteur constant unitaire.
- Augmentation du taux de compression en fusionnant les blocs similaires au sein d'une même partition "Range".
- Utilisation des bases orthogonales lors du calcul des transformations locales.

En parallèle avec les travaux visant à définir un opérateur optimal agissant dans l'espace des niveaux de gris, des chercheurs utilisent actuellement la théorie des IFS dans des Schémas hybrides fondés sur la transformée en ondelettes ou en cosinus discrète. La méthode proposée par Barthel et Al améliore sensiblement la norme JPEG à taux de compression élevé. Elle combine la transformation en cosinus discrète (codage de la redondance intra-blocs) et la transformation fractale (codage inter- blocs). La méthode consiste pour un bloc "Range" donné, a approximer la majeure partie de son spectre par une transformation fractale opérant sur les coefficients fréquentiels d'un autre bloc de l'image, appelé bloc "Domain".

Les études liées à la compression de séquences vidéo par IFS sont relativement récentes (1991) et se situent le plus souvent dans le prolongement de travaux antérieurs réalisés dans le cadre de compression d'images fixes. La plupart des publications datent de 1994-1995.

Actuellement deux approches ont été proposées dans la littérature (plus les versions hybrides de ces deux approches) sur l'extension possible de l'algorithme de Jacquin pour la compression de séquences vidéo :

Approche volumique :

La séquence à coder est subdivisée en plusieurs GOP (Group of Pictures/ groupes d'images). Au lieu de segmenter une image en un ensemble de blocs carrés, chaque GOP est découpé en un ensemble de cubes. la troisième dimension correspond à la dimension temporelle. Plusieurs algorithmes ont été proposés dont :

Algorithme de Lazar et Bruton :

La séquence vidéo est divisée en des suites d'images de deux sortes, les R- frames et les D-frames, on définira respectivement les cubes "Range" et "Domain", ces cubes sont considérés comme des suite ordonnées de blocs "Range" et "Domain" bidimensionnels. Le sous échantillonnage est tridimensionnel et effectué dans les trois dimensions.

Algorithme de Naemura et Harashima :

Les auteurs exposent un modèle de codage fractal tridimensionnel pour compresser et interpoler des prises de vue réalisées à l'aide de plusieurs caméras (l'idée de base est d'exploiter le fait qu'un code fractal puisse être décodé à toute échelle).
Le modèle de codage fractal utilise des cubes "Range" tridimensionnels avec chevauchement (l'aspect tridimensionnel est obtenu en plaçant les différentes vues les unes après les autres formant ainsi un volume, la troisième dimension sera donc celle des vues), une recherche en spirale des cubes "Domain" et comme isométrie applicable à des cubes.

Approche mixte :

Cette classe d'algorithmes effectue un codage intra par IFS combiné à un codage inter par block matching.

Algorithme de Hurtgen et Buttgen :

Ce codage vidéo est une combinaison de prédiction par DPCM (Différentiel Pulse Coded Modulation) et de codage IFS. Le but recherché est le codage à très bas débit : 8 à 64 Kb/s. Chaque image de la séquence est prédite par une boucle temporelle. Les blocs prédits sont codés par IFS. L'algorithme de référence est augmenté d'une segmentation par quadtree. Les blocs sont recherchés dans toute l'image et la recherche est accélérée par un classement hiérarchique des blocs "Domain".

Algorithme de Fisher, Ragovin et Shen :

Chaque image de la séquence est codée par le schéma de codage classique de A. JACQUIN, en utilisant l'image précédente pour établir le dictionnaire de blocs "Domain", avec en plus une partition en quadtree et une recherche accélérée par classification. On pense atteindre 20 images par seconde au décodage. La recherche des blocs dans l'image précédente est plus ou moins une forme simplifiée de compensation de mouvement. Cela a pour conséquence de rendre les itérations inutiles au décodage.
Le problème du codage de l'image initiale a été posé pour cet algorithme : elle peut être soit codée différemment (i.e. codage IFS classique) soit codée à partir d'une image fictive en acceptant que les premières images décodées soient erronées.

Conclusion :

La procédure de codage fractal est basée sur une recherche de similarités au sein de l'image originale via un ensemble de transformations contractantes. Pour chaque bloc "Range", les similarités sont sélectionnées par la minimisation de la mesure de distorsion entre le bloc "Range" et le bloc "Domain" transformé. D'après la formule (III.5) la distance $d(I_{orig}, I_{décodé})$ vérifie l'inégalité suivante :

$$d(I_{orig}, I_{décodé}) \leq \varepsilon/(1-s).$$

Pour que le codage soit vraiment performant, il faudrait chercher simultanément à minimiser l'erreur du codage ε et trouver une contractivité s la plus proche possible de 0. En effet si s est proche de 1, le majorant devient très grand et lorsque s est supérieur à 1, la transformation n'est plus contractante. Malheureusement, il se trouve qu'aucun critère précis ne permet de réaliser cette deuxième condition lors du codage.

Chapitre 4

Conception et mise en œuvre

Introduction:

Notre algorithme a été développé sous l'environnement Windows dans le cadre de compresser et décompresser des images naturelles fixes en 256 niveaux de gris de taille 256 × 256 (cette taille est la plus répandue pour les tests mondiaux)

Le langage de programmation utilisé pour implémenter l'algorithme est le **Borland C++BUILDER 5 [30][31][32][33]**.

Le sujet de base de l'algorithme est la compression des images par la méthode fractale en utilisant un partitionnement carré selon l'approche de **JACQUIN [26]**.

A la compression, les fichiers d'images d'entrée sont reconnus par l'extension (.BMP **[Annexe B]**) utilisé par Windows. Les fichiers de sortie ou les codes de compression ont l'extension (.FRC).
A la décompression, les codes ayant l'extension (.FRC) servent à reconstruire les images décompressées sous forme (.BMP)

Figure IV.1 : Processus de compression.

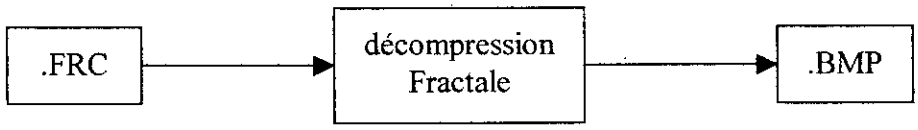


Figure IV.2 : Processus de décompression.

I. Approche adoptée:

La méthode de codage d'images à base de fractales utilisée se décompose en trois parties essentielles **[26]**:

Partie1: La construction de l'ensemble des "Domain" (dictionnaire des blocs "Domain") avec une classification.
Partie2: La construction de L'ensemble des transformations.
Partie3: Le codage de l'ensemble global (Code- book ou Code fractal).

Considérons une image I de taille n×n (n = 256), quantifiée à G niveaux de gris (G = 256). Cette image est partitionnée en blocs "Range" R_i de taille B×B (non chevauchés). Pour chaque bloc "Range" R_i, on doit trouver un bloc "Domain" D_j de taille D×D (D = 2B), et une transformation τ_i tel que l'erreur entre R_i et $\tau_i(D_j)$ est minimale. Nous rappelons que la forme des transformations fractales par bloc est:

$$\tau = \sum_{i=1}^{N} \tau_i = \sum_{i=1}^{N} M_i \circ G_i \qquad (IV.1)$$

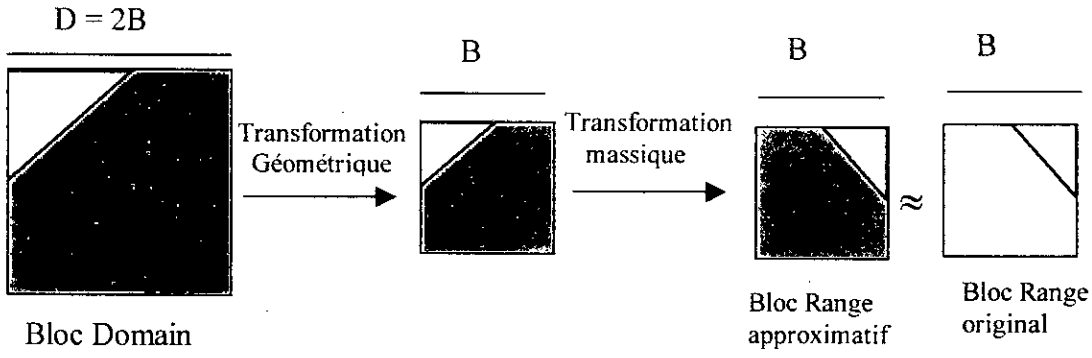
tel que :

τ est la transformation fractale de l'image.
τ_i est la transformation fractale du bloc.
G_i est la transformation géométrique.
M_i est la transformation massique

La construction de la transformation τ_i qui représente le code fractal, peut être effectuée en deux étapes différentes, qui correspondent aux transformations G_i, et M_i.

La première étape de la contraction spatiale G_i, est équivalente à la sélection d'un bloc "Domain" D_j de l'image de taille D×D, qui sera transformé en bloc R_i de taille B×B. Le bloc résultat est dénoté par $G_i(D_j)$.

La deuxième étape de construction de τ_i, consiste à sélectionner un traitement propre pour le bloc $G_i(D_j)$, en lui trouvant une transformation massique de bloc M_i, qui minimise la distorsion entre $M_i \circ G_i(D_j)$ et R_i.



FigureIV.3: Effet des deux transformations G_i et M_i sur un bloc 'Domain'.

Donc, on définira formellement un ensemble de blocs "Domain" D_{init}, qui contiendra tous les blocs d'image possibles et qui peuvent être extraites de l'image originale. Cet ensemble peut être vu comme un ensemble d'arguments de la partie géométrique i.e la contraction spatiale. On définira encore un ensemble de transformations massiques M. Le codage d'un bloc "Range" consiste à trouver la meilleure paire $(D_j, M_i) \in D_{init} \times M$, qui satisfait :

$d(R_i, M_i \circ G_i(D_j))$ est la valeur minimale.

On appellera le bloc de produit $D_{init} \times M$ le **Code-Book**.

I.1. Construction de l'ensemble de blocs "Domain":

L'ensemble initial de blocs "Domain" D_{init}, peut être obtenu en glissant une fenêtre de taille D×D (D = 2B) à travers l'image originale. La fenêtre est localisée la première fois au point (0,0) de l'image, puis elle est déplacée d'un pas p_x de pixels horizontalement et d'un pas p_y pixels verticalement. Les p_x, p_y sont appelés les pas horizontaux et verticaux, ils sont typiquement choisis égaux à B ou B/ 2. L'ensemble maximal des blocs "Domain" est constitué de tous les blocs de taille D×D situés à l'intérieur de l'image.

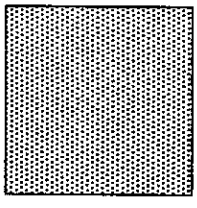
Dans le but d'optimiser la recherche pendant le codage, les blocs "Domain" dans l'ensemble D_{init} sont classifiés, selon leurs propriétés relatives aux contours.

On définit trois types de blocs et on crée trois classes D_s, D_e et D_m pour des blocs 'Shade', "Edge' et 'Midrange' respectivement **[26]**.

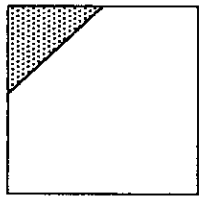
- Un bloc 'Shade' (nuance) est un bloc presque uniforme, il présente des variations négligeables de luminance.
- Un bloc 'Edge' (contour) présente une brusque variation d'intensité de pixels le long de la limite d'un objet.
- Un bloc 'Midrange' se caractérise par des gradients moyens, il présente les textures d'objets.

On a donc : $D_{init} = D_s \cup D_e \cup D_m$.

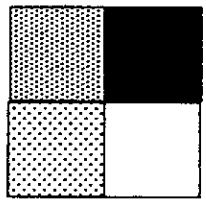
Durant l'étape de codage seul les blocs "Domain" de même type que le bloc "Range" sont analysés.



Bloc 'Shade'



Bloc 'Edge'



Bloc 'Midrange'

L'Algorithme de Classification:

L'algorithme de classification est implémenté en deux étapes: une étape de détection de limites, et une étape de décision **[34]**. L'algorithme donne des résultats significatifs seulement pour les petits blocs typiquement 4*4 ou 5*5 comme taille.

- **Première étape:** Détection de limites:

Soit X(i, j) l'élément de la position (i, j) du bloc de taille p×p. On définit deux seuils T_s et T_e qui ont des valeurs entre 0 et 1.

Six compteurs S_h, S_v, H_p, H_n, V_p et V_n sont mis à zéro. Les compteurs sont utilisés pour décider si le bloc X est un bloc 'Shade', Edge' ou 'Midrange'.

Deux tables appelées les tables des gradients sont construites. La table horizontale, dénotée par G_h, de taille p×(p-1) pixels, contient des informations sur les limites verticales. La table verticale dénotée par G_v, de taille (p-1)×p pixels contient des informations sur les gradients horizontaux.

On définit les gradients d_h et d_v:

$$d_h = \frac{2X(i,j) - X(i,j+1)}{X(i,j) + X(i,j+1)}, \qquad d_h = \frac{2X(i,j) - X(i+1,j)}{X(i,j) + X(i+1,j)}$$

qui sont les gradients dans la direction horizontale et verticale normalisés par la moyenne des intensités des pixels.

Les entrées dans les tables G_h et G_v sont données par :

$$G_h(i,j = \begin{cases} 1 & \text{si } d_h > T_e \\ -1 & \text{si } d_h < -T_e \\ 0 & \text{ailleurs} \end{cases} \qquad \begin{matrix} i = 1,2,\ldots,p \\ j = 1,2,\ldots,(p-1) \end{matrix}$$

$$G_v(i,j = \begin{cases} 1 & \text{si } d_v > T_e \\ -1 & \text{si } d_v < -T_e \\ 0 & \text{ailleurs} \end{cases} \qquad \begin{matrix} i = 1,2,\ldots,(p-1) \\ j = 1,2,\ldots,p \end{matrix}$$

Les compteurs S_h et S_v sont incrémentés comme suit:

$$S_h \leftarrow S_h + 1 \qquad \text{si } |d_h| > T_s \qquad i = 1,2,\ldots,p \text{ et } j = 1,2,\ldots,(p-1)$$

$$S_v \leftarrow S_v + 1 \qquad \text{si } |d_v| > T_s \qquad i = 1,2,\ldots,(p-1) \text{ et } j = 1,2,\ldots,p$$

H_p est la somme des +1 dans G_h et H_n est la somme des –1 dans G_h, d'une façon similaire V_p est la somme des +1 dans G_v et V_n est la somme des –1 dans G_v.

Les seuils T_s et T_e sont en fonction de l'intensité moyenne dénotée par d_{av} des deux pixels considérés:

$$T_e = \begin{cases} \dfrac{8.0}{d_{av}} & \text{si } d_{av} < 30.0 \\ 0.2 & \text{sinon} \end{cases}$$

$$T_s = \begin{cases} 0.1 & \text{si } d_{av} < 30.0 \quad \text{ou} \quad d_{av} > 255.0 \\ 0.025 & \text{sinon} \end{cases}$$

L'intensité est supposée dans l'intervalle [0, 255]. Pour éviter les fausses détections et détecter, donc convenablement la plupart des 'Edge', la valeur de T_e peut être considérée égale à 0.2 sauf quand d_{av} est très petite car dans ce cas, la valeur de T_e est choisie dépendante de d_{av}. Les seuils donnés précédemment sont choisis expérimentalement. Pour T_s, la valeur 0.025 a été choisie **[34]**.

- **Deuxième étape:** Décision

L'algorithme de décision travaille avec les tables et les compteurs qui ont été calculés. On définit deux seuils supplémentaires, J_s et J_e, tels que J_s = p-1 et J_e = p/2.

Algorithme:

Début
Si(S_h<J_s et S_v<J_s)
Alors "bloc Shade"
Sinon
Si(V_p≥J_e ou V_n≥J_e ou H_p ≥J_e ou H_n≥J_e)
Alors "bloc Edge"
Sinon "bloc Midrange"
Fsi
Fsi
Fin

I.2. Construction de l'ensemble des transformations:

On propose la procédure de construction de l'ensemble des transformations dans ce qui suit, chaque bloc "Range" R_i de l'image est codé selon sa nature de la manière suivante **[26]**:

Cas N°1: R_i est un bloc 'Shade':

Aucune recherche à faire, On approxime le bloc R_i par un bloc uniforme avec un niveau de gris égal à la moyenne des niveaux de gris de R_i, qui est dénoté par $\overline{R}_i$, la transformation M_i, qui génère ce type de bloc est l'absorption par g_i:

$$g_i = \overline{R}_i . \qquad \text{(IV.2)}$$

Cas N°2: R_i est un bloc 'Midrange' ou 'Edge':

Chaque bloc "Domain" D_j de l'ensemble D_{init} de même type que le bloc R_i est analysé. L'analyse de la paire (R_i, D_j) détermine la sélection de la transformation M_i utilisée pour coder R_i.

Cas N°2.1: R_i est un bloc 'Midrange':

Pour les blocs 'Midrange', on restreint notre attention aux transformations massiques qui sont composées d'une modification du contraste et d'un décalage de la luminance de la forme:

$$M_i(G_i(D_j)) = \alpha_i \times (G_i(D_j)) + \Delta g_i \qquad \text{(IV.3)}$$

tel que:

M_i est la transformation massique.

G_i est la transformation géométrique.

α_i est le facteur du scalaire appartenant à l'ensemble $\{0.7, \ldots, 1.0\}$.

Δg_i est le décalage de la luminance calculé comme suit:

$$\Delta g_i = \text{moyenne}(R_i) - \alpha_i \times \text{moyenne}(D_j) \qquad \text{(IV.4)}$$

Cas N°2.2: R_i est un bloc 'Edge':

L'analyse de la paire (R_i, D_j) consiste en une segmentation des blocs R_i et $G_i(D_j)$. On suppose que les blocs de l'image peuvent être segmentés en deux régions uniformes: une qui est foncée et l'autre qui est claire, séparées par une région de transition. Les blocs segmentés sont dénotés par R_i^{seg} et $G_i(D_j)^{seg}$ respectivement. On calcule par la suite la dynamique des blocs segmentés comme une différence des moyennes des niveaux de gris entre les régions foncée et claire. La transformation massique utilisée, dans ce cas la, est la composition d'une modification du contraste, d'un décalage de luminance et d'une isométrie de la forme:

$$M_i(G_i(D_j)) = i_n(\alpha_i \times (G_i(D_j)) + \Delta g_i) \qquad \text{(IV.5)}$$

tel que:

i_n est une isométrie.

M_i est la transformation massique.

G_i est la transformation géométrique.

α_i est le scalaire du contraste.

Δg_i est le décalage de la luminance.

Le coefficient α_i est calculé de la manière suivante:

$$\alpha_i = \min\left\{\frac{dr(R_i^{seg})}{dr(G_i(D_j)^{seg})}, \alpha_{max}\right\} \qquad \text{(IV.6)}$$

tels que :

α_{max} est la valeur maximale du coefficient du contraste, quantifié par la valeur proche dans l'ensemble {0.5, 0.6, 0.7, 0.8, 0.9, 1.0}.

dr est la gamme dynamique d'un bloc: dr = moyenne (région claire)- moyenne (région foncée).

Δg_i est calculé de la manière suivante :

$$\Delta g_i = \text{moy (région dominante } (R_i^{seg}) - \alpha_i \times \text{moy(région dominante } (G_i(D_j)^{seg})) \qquad \text{(IV.7)}$$

Une des huit isométries $\{i_n\}_{0 \le n \le 7}$ qui minimise la distorsion $d(R_i, M_i \circ G_i(D_j))$ est sélectionnée.

I.3. Codage des paramètres:

L'opération du codage se fait comme suit:

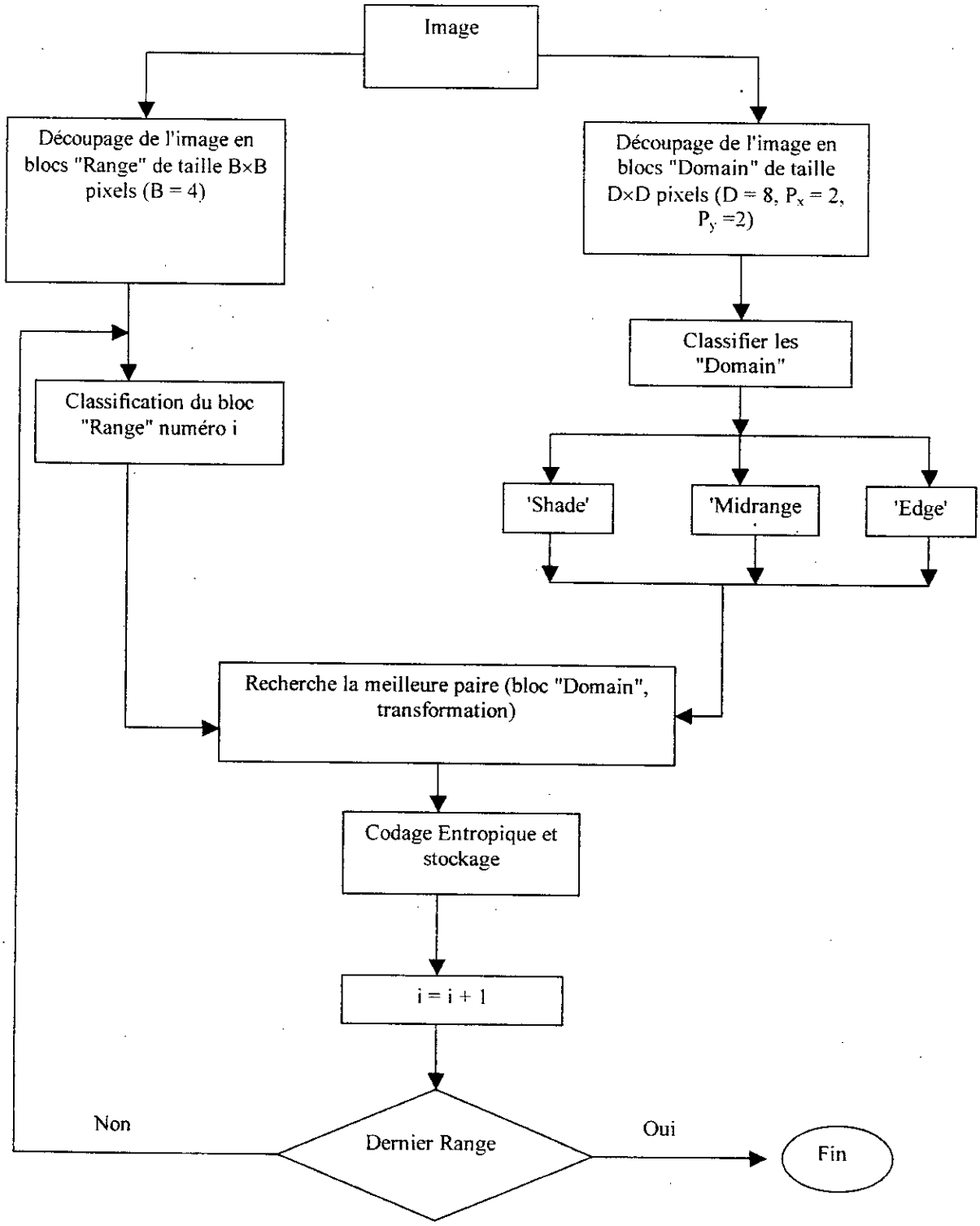
- Cas d'un bloc 'Shade': Pour les blocs "Range Shade", on calcule la moyenne et on stocke les paramètres suivants: l'identificateur de la classe et la moyenne du bloc.
- Cas d'un bloc 'Midrange': La transformation massique utilisée est formée d'une modification du contraste et d'un décalage de luminance, donc les paramètres à stocker sont: l'identificateur de la classe, la position du bloc "Domain", le scalaire du contraste et le décalage de la luminance.
- Cas d'un bloc 'Edge': Pour ces blocs, la transformation fractale à stocker est constituée d'une transformation géométrique et d'une transformation massique traitant l'intensité des pixels (le scalaire du contraste et le décalage de la luminance) et agissant sur la position des pixels (isométrie), donc les paramètres à stocker sont: l'identificateur de la classe, la position du bloc "Domain", le scalaire du contraste, le décalage de luminance et le numéro d'une des huit isométries possibles.

La table IV.1 représente les paramètres stockés pour chaque type de bloc et le nombre de bits requis pour le codage des différents coefficients:

Tableau IV.1 Le Code fractal

Type du bloc	Paramètres	Nombre de bits	Total
'Shade'	Classe du bloc Moyenne	2 8	I_s = 10bits
'Midrange'	Classe du bloc Cordonnées du "Domain" α_i Δg_i	2 7+7 2 9	I_m = 27 bits
'Edge'	Classe du bloc Cordonnées du "Domain" α_i Δg_i isométrie	2 7+7 3 9 3	I_e = 31 bits

II. Organigramme de la méthode:



II.1. Pseudo-algorithme de codage:

Début
*/*Création de l'ensemble des "Domain" */*
Lancer la création et la classification de l'ensemble des "Domain"
*/*Codage d'un bloc "Range" par fractal*/*
Pour chaque bloc R_i de l'image originale
faire
{
Si le bloc R_i est un bloc 'Shade'
Alors lancer le traitement 'Shade'
Fsi
Si le bloc R_i est un 'Midrange'
Alors lancer le traitement 'Midrange'
Fsi
Si le bloc R_i est un bloc 'Edge'
Alors lancer le traitement 'Edge'
Fsi
*/*Codification du code- book*/*
Lancer le codage entropique du bloc
Sauvegarder le code fractal final dans un fichier binaire.
}
Fin

Nous présentons maintenant les trois pseudo-algorithmes concernant les traitements cités dans l'algorithme de base présenté au–dessus:

Pseudo-algorithme de traitement 'Shade'

Début
/ le bloc R_i est un bloc 'Shade'*/*
sauvegarder (la moyenne)
Fin

Pseudo-algorithme de traitement 'Midrange'

Début
/ le bloc R_i est un bloc 'Midrange' */*
pour chaque bloc "Domain" D_j – 'Midrange'
faire
{
Appliquer la transformation géométrique sur le bloc "Domain" D_j
Pour Chaque $\alpha_i \in \{0.7, 0.8, 0.9, 1.0\}$
Faire
{
g_i = moyenne(R_i)- $\alpha_i \times$ moyenne(D_j);
Estimer α_i et g_i tel que l'erreur RMS entre R_i et $\alpha_i \times G_i(D_j)+g_i$ est minimale
}
}
sauvegarder (α_i, g_i, D_j) dans le code book
Fin

Pseudo-algorithme de traitement 'Edge':

Début
/ le bloc R_i est un bloc 'Edge' */*
pour chaque bloc "Domain" D_j 'Edge' dans l'ensemble des "Domain"
faire
{
Appliquer la transformation géométrique sur le bloc "Domain" D_j
Pour Chaque valeur α_i trouvée
Faire
{
g_i = moyenne(R_i)- $\alpha_i \times$ moyenne(D_j);
Estimer α_i et g_i tel que l'erreur RMS entre R_i et $\alpha_i \times G_i(D_j)+g_i$ est minimale
Pour chaque isométrie i_n
{
appliquer i_n au bloc résultat ($\alpha_i \times G_i(D_j)+g_i$)
sélectionner l'isométrie i_n / La RMS entre R_i et $i_n(\alpha_i \times G_i(D_j)+g_i)$ est minimale.
}
}
}
sauvegarder (i, α_i, g_i, D_j) dans le code-book
Fin

II.2. Pseudo-algorithme de décodage:

Début
Prendre une image initiale I_0 quelconque.
Itérer
Pour chaque bloc "Range" R_i de I_0
{
Lire le code-book (i, α_i, g_i, D_j)
Appliquer la transformation géométrique sur le bloc D_j.
Si $\alpha_i = 0$
Alors
Appliquer l'absorption sur le bloc contracté
Faire une translation du bloc résultant
Fsi
Si $i = 0$
Alors appliquer la transformation affine sur le bloc contracté
Faire une translation du bloc R_i transformé
Fsi;
Si ($i \neq 0$ et $\alpha_i \neq 0$)
Alors
Appliquer la transformation affine sur le bloc contracté;
Appliquer l'isométrie n° i sur le bloc transformé
Faire une translation du bloc R_i résultant;
Fsi;
}
Prendre l'image résultante comme étant image – initiale;
Fin de Itérer
FIN

III. Résultats Expérimentaux:

On a testé la performance de l'algorithme sur une des images de test standards "Lena" au format BMP. Les tests sont basés sur l'évaluation des trois critères essentiels de toute méthode de compression d'images :

- Le taux de compression.
- La qualité de l'image reconstruite.
- Le temps de compression.

III.1. Description des paramètres :

Les caractéristiques de base de l'image digitale donnée au codeur, ainsi que les spécifications du codage sont présentées dans le tableau suivant :

Tableau IV.2.

Paramètres	Valeurs
Image de test	
Nom	**Lena**
Résolution	**256 × 256**
Les niveaux de gris	**256 (g_{min} =0, g_{max} = 255)**
Spécifications du codage	
Partition	
Type de partition	**Carré**
Taille des blocs "Range"	**4 × 4**
Nombre de blocs "Range"	**4096**
Ensemble des blocs "Domain"	
Taille des blocs "Domain"	**8 × 8**
Les déplacements	**P_x = 2, P_y = 2**
Nombre de blocs "Domain"	**16384**
Classification	
Nombre de Blocs 'Shade'	**171**
Nombre de Blocs 'Midrange'	**2263**
Nombre de Blocs 'Edge'	**1662**

III.2. Reconstruction de l'image :

Le décodage consiste à itérer le code fractal sur n'importe quelle image initiale, jusqu'à la convergence vers une image stable.

Le tableau suivant montre l'erreur entre l'image originale (**Lena**) et les images obtenues après 14 itérations. L'image initiale à laquelle on applique le code fractal est une image uniformément grise.

Tableau IV.3

Itération n°	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**
Erreur RMS	60.51	35.02	15.93	9.93	8.02	7.73	7.57	7.49	7.46	7.44	7.43	7.42	7.42	7.42

Pour l'image de (**Lena**), le processus de décodage s'arrête lorsque la distorsion entre deux images successives devient inférieure à un seuil fixé (0.01). Comme le montre le tableau ci-dessus, l'erreur se stabilise à partir de la douzième itération, le nombre d'itérations est donc fixé à 12

Image Originale

Image reconstruite

III.3. Présentation des résultats :

Les principaux résultats obtenus sont les suivants :

1. Le taux de compression évalué par **85.94 %.** Ce taux est comparable avec les taux déjà réalisés.
2. Une qualité acceptable comparée à celle rencontrée dans les différents articles, elle est évaluée par une distorsion RMS égale à **7.42** et un PSNR égal à **30.7** dB.
3. L'image est compressée en 58 minutes et 37 s. (sur un PIII)

III.4. Analyse des résultats :

D'après les résultats obtenus, nous sommes arrivés aux remarques suivantes :

- Premièrement, on notera que la reconstruction de l'image est d'une totalité bonne est fidèle à l'image originale. Les textures de l'image sont bien préservées en général, malgré que quelques-unes unes trop fines (le ruban autour de chapeau de Lena), même chose pour les formes et les contours. Une petite déformation est visible pour certains blocs de l'image, cette dernière est due à la forme des blocs "Domain" et "Range" utilisée.

- On observe par expérience, et dans la majorité des cas, que la reconstruction est visuellement proche de l'image originale. Ceci peut être expliqué par le fait que les transformations itératives ont une contractivité efficace.

- On soulignera l'importance de la classification des blocs et l'appariement de ces derniers, car il est montré que la majorité d'effets visibles au niveau de l'image décodée sont dues à une mauvaise classification de blocs.

- On constate que les blocs d'une taille trop grande, rendront la classification et l'appariement des blocs un peu difficile en pratique.

- Le nombre de transformations ou de comparaisons exigé est égal au produit :

(nombre de blocs "Domain")×(nombre de blocs "Range")×(nombre de transformations affines).

Ce qui explique le calcul intensif dans la phase du codage et par conséquent le temps consommé par son algorithme, c'est un inconvénient de cette méthode.

- Contrairement à la compression, la décompression fractale prend moins de temps.

- Enfin, l'image traitée contient des copies de parties. Ces parties n'étant pas sous certaines transformations affines, exactement des copies identiques d'elles-mêmes, il faut donc s'accorder à une certaine erreur dans notre représentation d'une image comme un ensemble de transformations. Ce qui implique que les images que nous encodons sont des plus ou moins bonnes approximations des originaux, donc l'établissement d'un bon code fractal revient à la bonne exploitation des similarités au sein des images.

III.5. Etude Comparative :

Pour une comparaison de plus des résultats obtenus, notre algorithme a été testé avec les résultats trouvés avec la méthode JPEG.

Tableau IV.4. Résultats obtenus avec la méthode fractale

Nom Image	Temps de compression	Taux de Compression (%)	PSNR (dB)
Lena	**58 min 37 s.**	**85.94**	**30.7**

Tableau IV.5. Résultats obtenus avec la méthode JPEG [8]

Nom Image	Temps de compression	Taux de Compression	PSNR (dB)
Lena	**1 s**	**79.69**	**39.67**

Conclusion :

Les résultats obtenus par la transformation fractale dans la compression des images fixes, sont prometteurs au niveau du taux de compression. La qualité de l'image reconstruite est presque égale. L'inconvénient majeur de cette méthode réside dans la lenteur de l'algorithme, cela est dû au nombre important de transformations et de comparaisons exigé pour le codage de chaque bloc "Range".

Conclusion

Conclusion

L'objectif principal de ce travail à été réalisé, du fait qu'on est arriver à compresser et décompresser des images naturelles fixes, en se basant sur les concepts de la géométrie fractale. Nous espérons que notre travail, soit une contribution enrichissante dans le domaine de la compression, et pourrait servir de support pour les recherches futures au laboratoire signal et communications.

La géométrie des fractales est utilisée pour analyser et modéliser des objets et structures complexes, elle permet des applications dans divers domaines. La compression fractale des images représente une nouvelle approche. La méthode implémentée est basée sur l'utilisation de transformations contractantes itératives.

Au début de notre travail nous avons présenté quelques méthodes de compression d'images, et nous avons introduit des notions sur la théorie des fractales. Par la suit, on a donné les fondements mathématiques de la géométrie fractale et des systèmes IFS (Iterated Functions System) qui s'applique sur des images fractales. Pour les images réelles, le concept des IFS a été généralisé au IFS local (L-IFS), qui mène à choisir un certain partitionnement de l'image. Différents types de partitionnements ont été introduit. Dans cette étude, nous avons utilisé le partitionnement carré.

Le codage fractal nécessite, pour chaque bloc de l'image, des comparaisons de manière systématique et séquentielle, afin de trouver le bloc plus similaire pour une distorsion donnée. Le temps de codage obtenu est donc prohibitif. Pour réduire le temps de cette recherche et accélérer la phase du codage, nous avons choisi la classification. Cette technique est basée sur le calcul du gradient.

Le décodage qui consiste simplement à iterer le code fractal sur n'importe quelle image initiale, prend moins du temps.

Pour un partitionnement carré de l'image, le choix de la taille des blocs se fait conformément à nos besoins et suivant la qualité de l'image reconstruite que nous voulons obtenir. Dans le cas où les détails de l'image ne sont pas importants, on peut privilégier le taux de compression et négliger un peu la qualité de l'image décodée ou reconstruite, en choisissant une taille large des blocs "Range". D'autre part, si les détails de l'image sont importants, il faut donc privilégier la qualité de l'image décodée et négliger le taux de compression, en utilisant des petites tailles.

De point de vue pédagogique, cette étude a été très bénéfique, du moment qu'elle nous a permis d'acquérir des connaissances intéressantes sur le traitement d'images en général et la compression en particulier et de comprendre les bases théoriques et pratiques de la compression par fractales. Ainsi, elle nous a permis d'implémenter un logiciel de compression d'images fixes à base de la géométrie fractale que nous pouvons juger qu'il est efficace d'après les tests et les résultats obtenus.

La performance de la méthode fractale dépend de plusieurs paramètres : le partitionnement de l'image, la méthode de recherche des blocs "Domain", la classification des blocs de l'image... Ceux ci pourraient optimiser la méthode et rivaliser, à terme, la norme de compression JPEG.

Bibliographie

[1] A. Marion, "Introduction aux techniques de traitement d'images," Eyrolles, 1987.

[2] M. Kunt, "Traitement numérique des images," Presses polytechniques et Université Romande, 1993.

[3] A.N. Netravali and al., "Picture Coding : A review," Proceeding of the IEEE, vol. 68, pp 366-407, March 1980.

[4] M. Nelson, "La compression de donnés," Dunod, 1993.

[5] B. Mandelbrot, "Fractals form, chance, and dimension," W.H. Freeman and Company , 1977.

[6] M.F. Barnsley and A.D. Sloan, "A Better way to compress images," Byte Magazine, pp 215-223, Janvier 1988.

[7] R.M. Gray, "Source coding theory," Kluwer, Academic Press, 1990.

[8] B. Brahim, "Etude et implémentation d'algorithmes de compression d'images fixes," Thèse de Magister, Institut d'électronique, USTHB, 1998.

[9] R.C. Gonzalez and P. Wintz, "Digital Image Processing," Addison Wesley, 1987.

[10] J.J. Toumazet, "Traitement de l'image sur micro- ordinateur," Sybex, 1987.

[11] C. Shouchard, " Les formats de fichiers image PC," PC Expert, pp 241-244, Juillet 1992.

[12] D. Berkani, " Spiral Quantizer Design," Thèse de doctorat, ENP, 1991.

[13] J.P. Dubus, " Compression d'images statiques et dynamiques," techniques de l'ingénieur E3070, 1996.

[14] N. Gamaz, " Etude et développement d'algorithmes de compression d'images fixes par TCD- 2D." Thèse de Magister, CDTA, 1992.

[15] N.S. Jayant, "Digital coding of speech waveforms: PCM, DPCM and DM quantizers." Speech," Proceeding of IEEE, vol. 25, pp 611-634, May 1974.

[16] J.P. Adoul, "La quantification vectorielle des signaux : approches algébriques," Annales des télécommunications, vol. 41, 1986.

[17] M. Goldberg and A. al, "Image compression using adaptive vector quantization," IEEE Trans. on Communications, vol. COM-34, pp 180-187, February 1986.

[18] W.A. Pearlman, R.M. Gray, " Source coding of the discrete Fourier Transform," IEEE Trans. On Information theory, vol. IT-24, pp 683-692, November 1978.

[19] G. Wallace, "The JPEG still picture compression standard," Gregory K. Wallace, Communication of the ACM, vol. 34, pp 31-44, April 1991.

[20] N. Ahmed and al., "Discrete cosine transform," IEEE Trans. on Computer, pp 90-93, January 1974.

[21] L. Mehiddine, "Contribution à l'utilisation des fractales en stéréovision," Thèse de Magister, USTHB, 1998.

[22] K. Falconer, "The geometry of Fractal sets," Cambridge University Press, 1985.

[23] N. Lassouaoui, "Segmentation des images biomédicales par des approches fractales," ENP, 2000.

[24] G. Gherbit, " Fractals : Non integral dimensions and applications," Masson, 1987.

[25] J. Hutchinson, "Fractals and self similarity," Indiana University Journal on mathematics, 1988.

[26] A. Jacquin, "Image coding based on a fractal theory of iterated contractive image transformation," IEEE Trans. on Image Processing, Vol. 1, pp 18-30, January 1992.

[27] H. Hartenstein and M. Ruhl, "Region-Based fractal image compression," IEEE Trans. on Image processing, vol. 9, pp 1171-1183, July 2000.

[28] B. Wohlberg and G. de Jager, "A review of the fractal image coding literature," IEEE Trans. on Image processing, vol. 8, December 1999.

[29] H.T. Chang and C.J. Kuo, "Iteration-free fractal image coding based on efficient domain pool design," IEEE Trans. on Image processing, vol. 9, pp 329-338, March 2000.

[30] C. Delannoy, "Apprendre à programmer en Turbo C," Eyrolles, 1994.

[31] C. Delannoy, "Exercices en langage C," Eyrolles, 1994.

[32] T. Swan," Tom Swan's mastring Borlan C++5," Sams Publishing, 1996.

[33] G. Leblanc, "Borland C++ Builder," Eyrolles,1998.

[34] B. Ramamurthi and A. Gersho, "Classified vector Quantization of images," IEEE Trans. on communications, vol. COM-34, pp 1105-1115, November 1986.

Annexes

Annexe A : Techniques de Calcul de la Dimension Fractale :

Plusieurs définitions de dimensions fractales ont été proposées depuis le début du siècle, certaines ont un intérêt purement théorique, d'autres sont plus faciles à calculer numériquement **[39]**.

1. Dimension de Hausdorff- Beescovitch :

Cette notion a été introduite par Hausdorff en 1919. C'est la dimension la plus connue et la plus utilisée théoriquement, par contre son calcul numérique est délicat.
Soit E un ensemble de R^n et soi ε un nombre réel positif. Un ε- recouvrement de E, est une famille de boules U_i tel que le diamètre de U_i est inférieur ou égal à ε et $E \subset \bigcup_i U_i$.

Pour α positif, on définit la mesure de Hausdorff de E par :

$$m^{\alpha}(E) = \lim_{\varepsilon \to 0} \left\{ \sum (\text{diam} V_i)^{\alpha} : E \subset \cup V_i, \text{diam} V_i \leq \varepsilon \right\}$$

L'inf est pris pour tous les recouvrements de E.

On définit aussi la dimension de Hausdorff ou (Hausdorff- Beescovitch) par :

$$D_H = \inf\{\alpha, H(\alpha, E) = 0\} = \sup\{\alpha, H(\alpha, E) = \infty\}$$

La dimension de Hausdorff est la valeur de α pour laquelle la mesure fait un saut de zéro à l'infini.

2. La dimension d'homothétie :

Les calculs relatifs à la dimension de Hausdorff sont délicats à exécuter. Dans une telle situation, on tourne vers d'autres moyens pour définir la dimension. La définition de la dimension d'homothétie découle de la définition d'auto-similarité. Etant donné un ensemble auto- similaire, composé de N copies distinctes de lui-même. Chaque copie est mise à l'échelle avec un rapport d'homothétie r pour toutes les coordonnées. La dimension fractale ou d'homothétie de cet ensemble est alors donnée par la relation :

$$Nr^D = 1 \text{ ou } D = \frac{\log N}{\log(1/r)}$$

Les surfaces fractales naturelles, en général, ne possèdent pas cette auto- similarité déterministe. Cependant, elle montre une auto- similarité statistique. Ceci étant, elles sont composées de N sous-ensembles distincts misent à l'échelle avec un rapport r à partir de

l'ensemble original. La dimension fractale pour ses surfaces est aussi donnée par l'équation (B.3).

Par exemple, dans le cas de la courbe de Von Koch : N = 4, r = 1/3, D = log4/ log3 = 1.2618. A l'ordre n de la construction, le tapis de Sierpinski est constitué de 3^n triangles équilatéraux avec des côtés de longueur $(1/2)^n$. Par conséquent, D = log3/ log2 = 1.585. Pour le cas de la fractale non uniforme de la figureII.7, contenant les facteurs 1/ 4 et 1/ 2 et. Sa dimension se détermine sur une seule itération comme suit : $4(1/4)^D + (1/2)^D = 1$, soit :

$$D = (\log(1+\sqrt{7})/\log 2) - 1$$

3. Dimension de Bouligand- Minkowski :

Elle fut originalement définit par Kolmogorov dans le cas des dimensions entières. Elle permet de caractériser l'aspect auto- similaire d'un ensemble. Nous citons deux méthodes permettant de la calculer :

La saucisse de Minkowski :

Soit E un ensemble plongé dans un espace euclidien de dimension d. Soit E(ε) l'ensemble des points de R^d distants de moins de ε de E. E(ε) définit une saucisse de Minkowski. On calcule :

$$D_M = \lim_{\varepsilon \to 0}\left(d - \frac{\log \mathrm{Vol}_d(E(\varepsilon))}{\log \varepsilon} \right)$$

où Vol_d représente le volume en dimension d (longueur, surface ou volume). Si la limite existe D_M est par définition la dimension de Bouligand- Minkowski.

La méthode des boîtes :

Soit N(ε) le nombre de pavés de côté ε recouvrant E :

$$D_B = \lim_{\varepsilon \to 0} \frac{\log N(\varepsilon)}{-\log \varepsilon}$$

En pratique, on obtient cette dimension comme la pente de droite de régression (moindres carrés) de l'ensemble des points du plan de coordonnées ($\log 1/\varepsilon$, $\log N(\varepsilon)$).

Ensemble, Nous pouvons dire que chacune de ces dimensions, révèlent quelques aspects différents de la structure d'un élément donné, et que les fractales appartiennent à une théorie complète basée sur la dimension fractale.

Annexe B: Le format BMP

1. Définition :

Un bitmap est une représentation point par point d'une image. Le fichier bitmap doit avoir l'extension BMP et être conforme au format « bitmap WINDOWS ».

2. La structure des fichiers BMP de Windows :

Un fichier BMP est composé de:

2.1. BITMAPFILEHEADER :

Cette partie correspond à l'entête du fichier bitmap. Elle contient les informations générales sur ce fichier :

- Un identificateur du fichier qui contient les lettres "BM" dans les deux premiers octets.
- La taille du fichier sur quatre octets.
- Une suite de quatre octets réservés à zéro.
- Un déplacement à partir du début de fichier.

La taille totale de la structure BITMAPFILEHEADER est de 14 octets.

2.2. BITMAPINFO :

Elle-même est composée de :

BITMAPINFOHEADER :

On trouve dans cette partie toutes les informations concernant les caractéristiques de l'image :

- La taille occupée par la structure BITMAPINFOHEADER.
- La largeur de l'image (en nombre de pixels).
- La hauteur de l'image (en nombre de pixels).
- Un nombre de surface de l'image doit valoir 1.
- Un nombre de bits nécessaires pour coder une couleur (1, 4, 8, 16 ou 24).
- Type de compression de données : BI_RGB, BI_RLE4, BI_RLE8. La majorité des bitmaps ont le format BI_RGB, ce qui signifie qu'ils ne sont pas compressés.
- Résolution horizontale (en nombre de pixels).
- Résolution verticale (en nombre de pixels).
- Nombre de couleurs dans la table des couleurs. Ce champ contient zéro, pour informer que toutes les couleurs sont utilisées.
- Nombre de couleurs considérées comme importantes. Il est égal à zéro si toutes les couleurs sont importantes.

2.3. RGBQUAD :

Cette structure représente la palette des couleurs utilisée pour l'affichage. Une couleur est une combinaison de trois couleurs de base : Bleu (BLUE), le vert (GREEN) et le rouge (RED). Chaque couleur de base à une intensité donnée pour approcher la couleur souhaitée. La structure RGBQUUAD, offre la possibilité de savoir ces intensités. Le nombre d'octets réservé à chaque couleur est :

- 1 octet pour le bleu.
- 1 octet pour le vert.
- 1 octet pour le rouge.
- 1 octet n'est pas utilisé et doit valoir 0.

Les différentes intensités sont représentées par des nombres entre 0 et 255.

3. Les données BMP :

Chaque pixel est représenté par 1, 4, 8, 24 bits. L'origine est définie comme étant le coin inférieur de l'image, et les pixels sont analysés ligne par ligne. Tous les bits représentant les pixels d'une même ligne sont cancatenés entre eux. Cependant, la représentation d'une ligne complète devra, suivant les cas, être complétée avec des bits zéro afin d'occuper un nombre entier de 32 bits.