الجمهورية الجزائرية الديمقراطية الشعبية
REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
وزارة التعليم العالي والبحث العلمي
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR
ET DE LA RECHERCHE SCIENTIFIQUE

جامعة محمد بوضياف – المسيلة
UNIVERSITE MOHAMED BOUDIAF - M'SILA
كلية التكنولوجيا
FACULTE DE TECHNOLOGIE
قسم الإلكتـرونيك
DEPARTEMENT D'ELECTRONIQUE

MEMOIRE DE MASTER

DOMAINE : SCIENCES ET TECHNOLOGIE
FILIERE : ELECTRONIQUE
OPTION : CONTROLE INDUSTRIEL

THEME

DEVELOPPEMENT ET EVALUATION DES TECHNIQUES CLASSIQUES DE COMPRESSION

Proposé et dirigé par :
Dr. BOUKHENNOUFA N.

Réalisé par :
AHMED Oussama

N° D'ordre: CI07

PROMOTION: JUIN 2016

Dédicace

Je dédie ce modeste travail A :

- ✓ *Mes très chers parents H & S que Dieu les garde et les protège pour leur soutien moral et financier, pour leurs encouragements et les sacrifices qu'ils ont endurés.*
- ✓ *Mes frères*
- ✓ *Mes chers amis,*
- ✓ *Tous les amis d'études surtout ceux de l'électronique promotion 2016*

Remerciements

- ✓ *Je remercie Dieu le tout puissant de m'avoir donné le courage et la patience pour réaliser ce travail malgré toutes les difficultés rencontrées.*
- ✓ *En préambule à ce mémoire, je souhaite adresser ici tous mes remerciements aux personnes qui m'ont apporté leur aide et qui ont ainsi contribué à l'élaboration de ce mémoire.*
- ✓ *Je tiens tout d'abord à remercier vivement mon encadreur Dr. BOUKHENNOUFA N. pour m'avoir dirigé au cours de ce travail, pour sa confiance et son soutien tout au long de ce travail.*
- ✓ *Mes respects et mes remerciements vont aux membres du jury d'examination, d'avoir accepté d'examiner et de juger ce travail.*
- ✓ *Merci à mes amis pour m'avoir soutenu par leur présence dans les bons comme dans les mauvais moments.*
- ✓ *Ma profonde gratitude à tous les Enseignants du Département d'Electronique qui m'ont encouragé à donner le meilleur de moi en m'assurant une formation aussi meilleure que possible.*
- ✓ *Je remercie enfin toute ma famille pour leur présence, leur soutien et leurs conseils.*

Ahmed Oussama

TABLE DES MATIERES

CHAPITRE 2 : Techniques de compression des données

CHAPITRE 3 : Transformée DCT

CHAPITRE 4 : Simulations et résultats

Liste des figures

Liste des Tableaux

Introduction générale

De nos jours, la puissance des processeurs augmente plus vite que les capacités de stockage, et énormément plus vite que la bande passante de réseaux informatiques, qui malgré les nouvelles technologies, a du mal à augmenter car cela demande d'énormes changements dans les infrastructures telles que les installations téléphoniques. Ainsi, on préfère réduire la taille des données en exploitant la puissance des processeurs plutôt que d'augmenter les capacités de stockage et de télécommunications, alors il faut faire une compression pour atteindre un meilleur taux de fidélité d'une capacité de stockage ou de transmission disponible. L'utilisation d'algorithmes de compression de donnée permettent en effet, une réduction importante de la quantité de données aussi que la bande passante.

La compression de données, de façon simplifiée, c'est l'ensemble des méthodes que l'on utilise pour prendre un message long pour en faire un message court, sans perdre d'information importante. Nous trouvons de tels usages dans la vie de tous les jours et pas seulement dans les application technologiques.

Ce travail s'inscrit dans la perspective de l'évaluation de techniques classiques de la compression de données (Huffman, Arithmétique, RLE et LZW) par les critères de performance qui sont:

- ✓ taux de compression,
- ✓ le rapport signal sur bruit en pic (PSNR) pour les images et
- ✓ Taux de distorsion (PRD) pour les signaux.

Ainsi, une comparaison entre ces techniques permet de juger les méthodes les plus performantes ainsi que leurs domaines d'application.

Organisation du mémoire

On a choisi d'articuler notre étude autour de quatre chapitres principaux :

Le premier chapitre est consacré à la présentation générale des signaux et images, il est divisé en deux sections, la première s'intéresse aux généralités sur le signal et prétraitements du signal (notions de base, classifications de signal et le filtrage numérique).

La deuxième section se focalise sur le traitement d'images, types d'images, concepts et définitions de base et formats d'images.

Le deuxième chapitre est plutôt dédié aux notions de base sur la compression des données, des généralité portant sur la compression des données puis on ouvrira un champ de réflexion en indiquant tout d'abord quelles sont les méthodes générales utilisées aujourd'hui pour compresser les données sans perte (codage de Huffman, codage arithmétique, RLE, LZW, …), Ensuite, on présentera deux méthodes très utilisées dans la compression des données avec perte, la quantification scalaire et la quantification vectorielle.

Le troisième chapitre, consacrera à la présentation générale à propos de la transformation en cosinus discrète et ses propriétés.

Le quatrième chapitre concrétisera une étude d'évaluation des performances des trois approches de compression de données. On a commencé par la compression du texte, puis compression du signal unidimensionnel ECG et finalement compression des images. Les résultats obtenus seront présentés dans ce chapitre également.

A la fin, on résumera notre humble travail par une conclusion générale concernant nos résultats empiriques obtenus.

CHAPITRE 1

SIGNAUX ET IMAGES

Introduction

Depuis longtemps, le traitement du signal ou image constitue un moyen essentiel dans le domaine de l'électronique. De ce fait, le traitement est devenu une discipline nécessaire, le fait que le domaine de traitement a vigoureusement évolué et ses techniques sont actuellement utilisées pour résoudre une variété de problèmes pour lesquels on adapte des solutions, selon la nature, les situations et les objectifs à atteindre.

Ce chapitre sera divisé en deux parties. La première s'intéresse aux propriétés et prétraitements de signal (notions de base). La deuxième partie aborde les techniques de traitement d'images.

I.1 Signal

I.1.1 Définitions de base

I.1.1.1 Signal

Un signal est une grandeur qui dépend du temps t. Cette grandeur est souvent physique [1]. La grandeur d'un signal peut être de différents types :

- Information
- Energie
- Matière

I.1.1.2 Bruit

Le bruit est défini comme tout phénomène perturbateur gênant la perception ou l'interprétation d'un signal, par analogie avec les nuisances acoustiques (interférence, bruit de fond, etc.). La différentiation entre le signal et le bruit est artificielle et dépend de l'intérêt de l'utilisateur :les ondes électromagnétiques d'origine galactique sont du bruit pour un ingénieur des télécommunications par satellites et un signal pour les radioastronomes [2].

I.1.1.3 Principales fonctions du traitement du signal

Les fonctions du traitement du signal peuvent se diviser en deux catégories : l'élaboration des signaux (incorporation des informations) et l'interprétation des signaux (extraction des informations). Les principales fonctions intégrées dans ces deux parties sont les suivantes :

– Élaboration des signaux :

- synthèse :création de signaux de forme appropriée en procédant par exemple àune combinaison de signaux élémentaires ;
- modulation, changement de fréquence :moyen permettant d'adapter un signal aux caractéristiques fréquentielles d'une voie de transmission;
- codage :traduction en code binaire (quantification), etc.

– Interprétation des signaux :

- filtrage :élimination de certaines composantes indésirables ;
- détection :extraction du signal d'un bruit de fond (corrélation);
- identification :classement d'un signal dans des catégories préalablement définies ;
- analyse :isolement des composantes essentielles ou utiles d'un signal de forme complexe (transformée de Fourier) ;
- mesure :estimation d'une grandeur caractéristique d'un signal avec un certain degré de confiance (valeur moyenne, etc.).

I.1.2 Classification des signaux

On peut classer les signaux selon différentes approches, ce qui induit des recouvrements entre les classes ainsi définies.

a. Représentation temporelle des signaux

On distingue ici les signaux qui prennent des valeurs à chaque instant *t* (Signal continu) et les signaux qui n'ont de valeurs qu'à certains instants t_i (Signal discret).

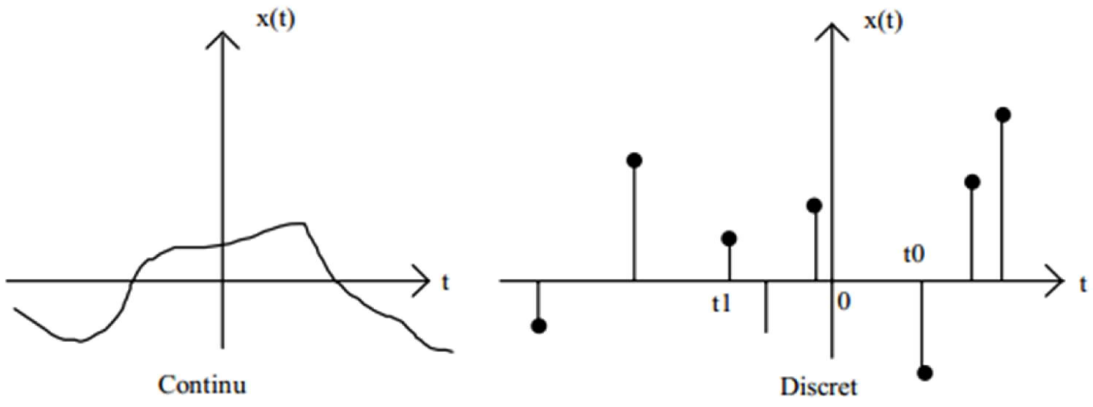


Figure 1.1 : Représentation des signaux continus et discrets

b. Energétique

On définit la puissance instantanée, $p(t)$, dans un dipôle, comme le produit de la tension $v(t)$ le courant $i(t)$ circulant dans le dipôle :

$$p(t) = v(t)i(t). \quad (1.1)$$

L'énergie dissipée dans le dipôle entre deux instants t_1 et t_2 vaut alors :

$$W(t_1, t_2) = \int_{t_1}^{t_2} p(t)dt \quad (1.2)$$

et la puissance moyenne sur l'intervalle est égale à :

$$P(t_1, t_2) = \frac{W(t_1,t_2)}{t_2-t_1} = \frac{1}{t_2-t_1}\int_{t_1}^{t_2} p(t)dt. \quad (1.3)$$

c. Typologique : Signaux certains et aléatoires

- **Signal certain** : Un signal $x(t)$ est certain (ou déterministe) s'il peut être décrit par un modèle mathématique. On distingue deux sous classes :

  Signaux périodiques :

  Un signal $x(t)$ est périodique s'il existe un réel $T > 0$, tel que :

  $$x(t) = x(t + kT), \forall k \in Z \quad (1.4)$$

  Si *T* est le plus petit réel satisfaisant à (1.4), *T* est appelée période du signal.

  - Les signaux non - périodiques.

- **Signal aléatoire** : Un signal $x(t)$ est aléatoire si son évolution est imprévisible et ne peut être décrite que par des grandeurs et méthodes statistiques

  Signal stationnaire : Un signal aléatoire $x(t)$ est stationnaire, si ses caractéristiques statistiques sont invariantes dans le temps.

  Signal stationnaire à l'ordre n : Un signal aléatoire $x(t)$ est stationnaire à l'ordre n, si ses caractéristiques statistiques sont invariantes dans le temps, jusqu'à l'ordre n inclus.

I.1.3 Caractéristiques d'un signal

- **Fréquence :** la fréquence est une grandeur physique en soi positive. Cela est évident si l'on interprète fréquence que le nombre de cycles par unité de temps dans un signal.
- **Distributions** : On appelle distribution d'une fonctionnelle linéaire continue sur l'espace vectoriel des fonctions définies sur n, indéfiniment dérivables et à support borné.

A toute fonction φ appartenant à , la distribution D associe un nombre complexe $D(\varphi)$, qui sera aussi noté par D, φ, avec les propriétés[5] :

- ✓ $D(\varphi_1 + \varphi_2) = D(\varphi_1) + D(\varphi_2)$.
- ✓ $D(\lambda\varphi) = \lambda D(\varphi)$ où λ est un scalaire.
- ✓ Si φ_j converge vers φ quand j tend vers l'infini, la suite $D(\varphi_j)$converge vers $D(\varphi)$.

I.1.4 Notions de traitement du signal analogique

I.1.4.1 Transformée de Fourier

La transformation de Fourier permet de décrire dans l'espace des fréquences un signal dont on connaît l'histoire au cours du temps et réciproquement.

$$F(\omega) = \int f(t)e^{-i\omega t}dt. \quad (1.5)$$

I.1.4.2 Convolution

*** Définition**

Une impulsion brève, injectée à l'entrée d'un système de transmission linéaire, continu et stationnaire, donne en sortie un signal de durée finie. Cette réponse est appelée réponse impulsionnelle (ou percussionnelle) du filtre et notée $h(t)$. Dans le cas général, c'est-à-dire pour signal d'entrée quelconque, nous avons une relation mathématique qui lie le signal d'entrée $e(t)$ et le signal de sortie s(t)pour un système de transmission possédant les trois propriétés vues précédemment ou filtre, noté S.T.-L.C.S., soit [5]:

$$s(t) = \int_{-\infty}^{+\infty} e(\tau) \cdot h(t-\tau) \cdot d\tau = e(t) * h(t). \quad (1.6)$$

I.1.5 De l'analogique au numérique

La conversion d'un signal analogique sous forme numérique implique une double approximation. D'une part, dans l'espace des temps, le signal fonction du temps $s(t)$ est remplacé par ses valeurs $s(nT)$ à des instants multiples entiers d'une durée T, c'est l'opération d'échantillonnage. D'autre part, dans l'espace des amplitudes, chaque valeur $s(nT)$ est approchée par un multiple entier d'une quantité élémentaire q, c'est l'opération de quantification. La valeur approchée ainsi obtenue est ensuite associée à un nombre ; c'est le codage, ce terme étant souvent utilisé pour désigner l'ensemble, c'est-à-dire le passage de la valeur $s(nT)$ au nombre qui la représente[3].

I.1.6 Notions de traitement du signal numérique

I.1.6.1 Echantillonnage

L'opération d'échantillonnage consiste à prélever sur un signal analogique dont l'évolution est continue dans le temps, des échantillons représentant l'amplitude aux instants de prélèvement.

Pour des raisons de simplification, les prélèvements sont réalisés régulièrement avec une périodicité constante Te appelée période d'échantillonnage. L'échantillonnage est qualifié d'idéal dès lors que l'on peut supposer ou approcher une prise instantanée des échantillons. Elle modélisé par le produit entre $X(t)$ et une suite périodique d'impulsions idéales appelée peigne de Dirac [2].

$$X^*(t) = T_e X(t) \sum_{k=-\infty}^{k=+\infty} \delta(t - kT_e). \quad (1.7)$$

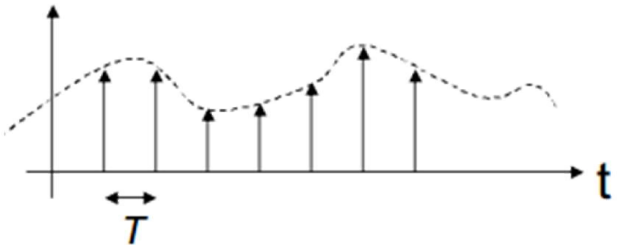


Figure 1.2 : Représentation d'un signal échantillonné

I.1.6.2 Transformée de Fourier Discrète

*** Définition**

Dans le but de calculer la transformée de Fourier d'un signal s(t) à l'aide d'un ordinateur, celui-ci n'ayant qu'un nombre limité de mots de taille finie, on est amené à discrétiser le signal (échantillonnage) et à tronquer temporellement ce signal. On obtient ainsi une suite de N termes représentée par :

$$s_{e,\Pi}(t) = \sum_{k=0}^{N-1} s(kT_e) \cdot \delta(t - kT_e) \quad (1.8)$$

Appelons s_k les valeurs du signal $s_{e,\Pi}(t)$ aux instants kT_e. Le spectre $s_{e,\Pi}(f)$ de ce signal échantillonné limité à N termes $s_{e,\Pi}(t)$ [2].

I.1.6.3 Transformée en Z

On appelle transformée en z de $(X_n)_{n\in N}$ la fonction, notée $Z[X_n]$, de la variable complexe z définie, lorsqu'il y a convergence, par :

$$Z[X_n](z) = \sum_{n=0}^{+\infty} X_n \, Z^{-n}. \tag{1.9}$$

L'origine de cette transformée en z s'explique en considérant le calcul de la transformée de Laplace d'un signal échantillonné causal. Soit le signal causal $s(t)$ échantillonné à la fréquence F_e [3].

I.1.7 Filtrage numérique

I.1.7.1 Définition

On appelle« filtre numérique »un système utilisé pour modifier la distribution fréquentielle d'un signal numérique selon des spécifications données [2]. Elle généralement utilisée pour 2 but : la séparation des signaux qui ont été combinés, et la restauration des signaux qui ont été déformées d'une manière quelconque [4]. Cela impose que la fonction générale précédente donnant les échantillons de sortie y_k, soit une combinaison linéaire des éléments x_i et y_i:

$$y_k = \sum_{i=0}^{N} a_i \cdot x_{k-i} - \sum_{j=0}^{N} b_j \cdot y_{k-j} \, . \tag{1.10}$$

Cette équation générale des filtres numériques est appelée équation aux différences.

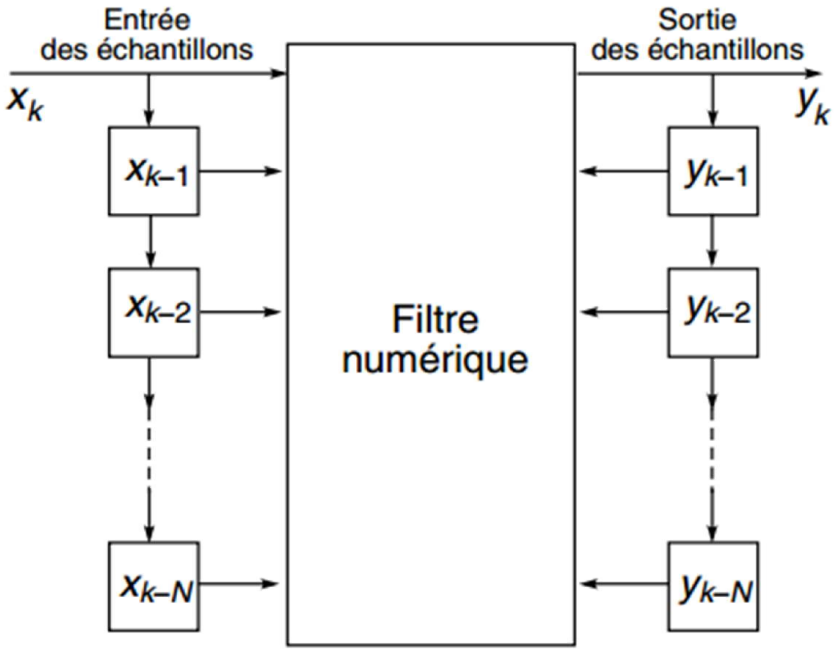


Figure 1.3: Représentation schématique d'un filtre numérique.

I.1.7.2 Classification des filtres numériques

Un filtre numérique peut être classé selon :

- la durée de sa réponse impulsionnelle

  finie : les filtres RIF ont leur réponse impulsionnelle à support fini i.e. $h(n) = 0$ pour $n < 0$ et $n > N$.

  infinie : les filtres RII ont leur réponse impulsionnelle à support infini i.e. $h(n) = 0$ $n = 0$

- le type de représentation temporelles

  récursifs : la sortie $y(n)$ dépend de l 'entrée courante, des entrées précédentes et des sorties précédentes.

  non récursifs : la sortie $y(n)$ ne dépend que de l 'entrée courante et des entrées précédentes.

I.1.7.3 Filtres à réponse impulsionnelle infinie (RII)

Le filtre RII général est un système qui, à la suite de données $x(n)$ fait correspondre la suite $y(n)$ telle que [5]:

$$y(n) = \sum_{l=0}^{L} a_l x(n-l) - \sum_{k=1}^{K} b_k y(n-k). \qquad (1.11)$$

I.1.7.4 Filtres à réponse impulsionnelle finie (RIF)

Les filtres numériques à réponse impulsionnelle finie (RIF) sont des systèmes linéaires discrets invariants dans le temps définis par une équation selon laquelle un nombre de sortie, représentant un échantillon du signal filtré, est obtenu par sommation pondérée d'un ensemble fini de nombres d'entrée, représentant les échantillons du signal à filtrer. Les coefficients de la sommation pondérée constituent la réponse impulsionnelle du filtre et un ensemble fini d'entre eux seulement prennent des valeurs non nulles. Ce filtre est du type « à mémoire finie», c'est-à dire qu'il détermine sa sortie en fonction d'informations d'entrée d'ancienneté limitée. Il est fréquemment désigné par filtre non récursif, en raison de sa structure, car il ne nécessite pas de boucle de réaction dans sa réalisation, comme c'est le cas pour une autre catégorie de filtres, celle des filtres à réponse impulsionnelle infinie. Les propriétés des filtres RIF vont être mises en évidence sur deux exemples simples [3].

I.2 Image

I.2.1. Introduction

Aujourd'hui, l'image constitue l'un des moyens les plus importants qu'utilise l'homme pour communiquer avec autrui. C'est un moyen de communication universel dont la richesse du contenu permet aux êtres humains de tout âge et de toute culture de se comprendre. C'est aussi le moyen le plus efficace pour communiquer, chacun peut analyser l'image à sa manière, pour en dégager une impression et d'en extraire des informations précises.

Le traitement d'images est l'ensemble des méthodes et techniques opérant sur cellesci, dans le but d'améliorer l'aspect visuel de l'image et d'en extraire des informations jugées pertinentes qu'on va utiliser dans différentes applications par exemple la reconnaissance, la classification, …etc.

Dans ce chapitre, on présente quelques principaux concepts de traitements d'images qui sont en relation avec notre sujet d'étude.

I.2.2 Historique

Le traitement d'image commence à être étudié vers les années 1920, pour la transmission d'images par le câble sous-marin allant de New York à Londres. En effet, la première numérisation d'image avec compression de données pour envoyer des fax de Londres à New York1. Les années 1950, les premiers travaux sur l'analyse d'images dans les chambres à bulles. Les années 1960, ont vu le véritable départ du traitement d'images quand les ordinateurs commencent à être suffisamment puissants pour travailler sur des images. Les premiers essais sur l'extraction automatique d'information ont commencé avec le début de la décennie 70. Apres les années 80, il y a eu un passage de l'image 2D aux modèles 3D.

I.2.3 Définition de l'image

Une image est une forme discrète d'un phénomène continu obtenue après discrétisation. Le plus souvent, cette forme est bidimensionnelle. L'information dont elle est le support est caractéristique de l'intensité lumineuse (couleur ou niveaux de gris).

I : [0, L-1] x [0, C-1] 7−→ [0, M]p définit une image de L lignes et C colonnes dont l'information portée est définit dans un espace à p dimensions. [9].

Si I est une image binaire, alors (p, M) = (1,1).

Si I est une image en niveaux de gris, alors p = 1 et le plus souvent M = 255.

Si I est une image couleur, alors p = 3 et le plus souvent M = 255.

I.2.4 Définition de l'image numérique

Le terme d'image numérique désigne, dans son sens le plus général, toute image qui a été acquise, traitée et sauvegardée sous une forme codée représentable par des nombres (valeurs numériques).La numérisation est le processus qui permet de passer de l'état d'image physique (image optique par exemple) qui est caractérisée par l'aspect continu du signal qu'elle représente (une infinité de valeur de l'intensité lumineuse par exemple), à l'état d'image numérique qui est caractérisée par l'aspect discret (l'intensité lumineuse ne peut prendre que des valeurs quantifiées en un nombre fini de points distincts). C'est cette forme numérique qui permet une exploitation ultérieure par des outils logiciels sur ordinateur [7].

Pour une mesure de simplicité, on utilise le terme image au lieu d'image numérique dans la suite du mémoire.

I.2.5 Types d'images

I.2.5.1 Image matricielle (bitmap)

Une image matricielle (voir Figure I.4) est une image en mode point. Le système de codage le plus universel consiste en effet à décomposer la représentation graphique, l'image, en un certain nombre de points élémentaires caractérisés par leurs coordonnées spatiales et leur couleur.

Figure 1.4 : Image matricielles.

I.2.5.2 Image vectorielle

Dans une image vectorielle les données sont représentées par des formes géométriques simples qui sont décrites d'un point de vue mathématique. Il s'agit de représenter les données de l'image par des formules géométriques qui vont pouvoir être décrites d'une façon mathématique.

Autrement dit, on stocke la succession d'opérations conduisant au tracé dans le cas d'une image vectorielle, alors qu'on mémorise une mosaïque de points élémentaires dans le cas d'image matricielle. Ces images présentent deux avantages : elles occupent peu de place en mémoire et peuvent être redimensionnées sans perte d'information.

I.2.6 Images bitmap

L'image est un ensemble structuré d'information caractérisé par les paramètres suivants :

I.2.6.1 Pixel

Contraction de l'expression anglaise « Picture éléments» : élément d'image, le pixel est le plus petit point de l'image, c'est une entité calculable qui peut recevoir une structure et une quantification. Si le bit est la plus petite unité d'information que peut traiter un ordinateur, le pixel est le plus petit élément que peuvent manipuler les matériels et logiciels d'affichage ou d'impression [9]. La lettre A, par exemple, peut être affichée comme un groupe de pixels dans la figure ci-dessous (Figure I.5) :

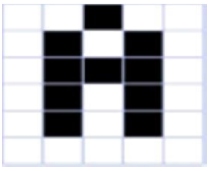


Figure 1.5 : Représentation de la lettre A sous forme d'un groupe de pixels

La quantité d'information que véhicule chaque pixel donne des muances entre image monochromes et image couleurs. Dans le cas d'une image monochrome, chaque pixel est codé sur un octet, et la taille mémoire nécessaire pour afficher une telle image est directement liée à la taille de l'image.

Dans une image couleur (R . V. B),un pixel peut être représenté sur trois octets : un octet pour chacune des couleurs : rouge (R), vert (V) et bleu (B) [9].

I.2.6.2 Codage de pixel

Pratiquement toujours, la valeur d'un pixel est un mot binaire de longueur Kbits, par conséquent un pixel peut prendre l'une des valeurs de l'intervalle [0…2K-1]. La valeur K est appelée profondeur de l'image.

L'ensemble de ces pixels est contenu dans un tableau à deux dimensions (une matrice) constituant l'image finalement obtenue.

I.2.6.3 Taille d'une image

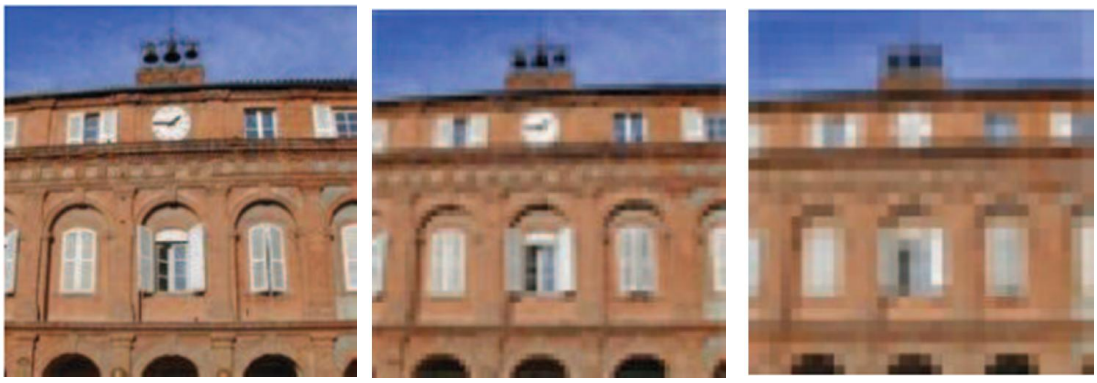
La taille d'une image est le nombre de pixels de cette image, la taille d'une image qui est représenté par (328×456) dont 328 est le nombre de lignes, et 456 est le nombre de colonnes, est égale à : 328 × 456 = 149568 pixels.

I.2.6.4 Résolution d'une image

Dans le domaine de l'imagerie numérique, la résolution est une mesure de la finesse de l'affichage ou de la capture d'une image, exprimée en nombre de pixels par unité de surface, c'est-à-dire la « densité » en pixels.

La résolution d'une image numérique s'exprime en PPI (Pixels Per Inch) ou PPP (Pixels Par Pouce).

Plus la résolution d'une image est grande plus sa qualité est meilleure (voir Figure I.6) [8].



(a) (b) (c)

Figure 1.6 : Résolution d'une image (a) image acquise à 256 dpi, (b) image acquise à 64 dpi, (c) image acquise à 32 dpi.

I.2.6.5 Luminance

C'est une forme d'énergie issue de deux composantes :

- une onde électromagnétique ondulatoire
- un aspect corpusculaire (les photons)

La lumière à une vitesse de déplacement d'environ 300000 km/s, et une fréquence d'environ 600000 GHz [9].

I.2.6.6 Contraste

C'est l'opposition marquée entre deux régions d'une image, plus précisément entre les régions sombres et les régions claires de cette image. Le contraste est défini en fonction des luminances de deux régions d'une image.

Si L1et L2 sont les degrés de luminosité respectivement de deux régions voisines A1 et A2 d'une image, le contraste C est défini par le rapport :

$$C=L1-L2/L1+L2. \quad (1.12)$$

I.2.6.7 Bruit

Un bruit (parasite) dans une image est considéré comme un phénomène de brusque variation de l'intensité d'un pixel par rapport à ses voisins, il provient de l'éclairage des dispositifs optiques et électroniques du capteur [10]. C'est un parasite qui représente certains défauts (poussière, petits nuages, baisse momentanée de l'intensité électrique sur les capteurs, … etc.). Il se traduit par des taches de faible dimension et dont la distribution sur l'image est aléatoire.

I.2.6.8 Histogramme

L'histogramme des niveaux de gris ou des couleurs d'une image est une fonction qui associe à chaque valeur d'intensité le nombre de pixels de l'image ayant cette valeur.

On appelle histogramme de l'image I, la fonction H définie sur l'ensemble des entiers naturels par : H(x) = CardP : I(P) = x c'est à dire que H(x) traduit le nombre d'apparitions du niveau de gris x dans l'image I. Cette définition se généralise aux images multi-bandes, l'histogramme est alors une fonction de p variables où p désigne le nombre de canaux.
L'histogramme est un outil privilégié en analyse d'images car il représente un résumé simple, mais souvent suffisant du contenu de l'image. Pour diminuer l'erreur de quantification, pour comparer deux images obtenues sous des éclairages différents, ou encore pour mesurer certaines propriétés sur une image, on modifie souvent l'histogramme correspondant [10].

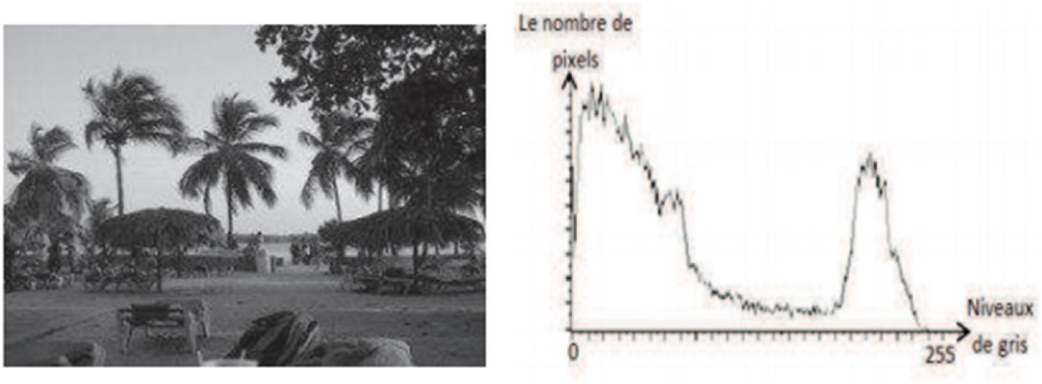


Figure 1.7: Exemple d'histogramme d'une image.

I.2.7 Codage des couleurs

I.2.7.1 Images binaires (noir ou blanc)

Les images binaires sont des images de profondeur K=1 bit, donc un pixel peut prendre l'une des valeurs : noir ou blanc (0 ou 1).

C'est typiquement le type d'images que l'on utilise pour scanner du texte quand celui-ci est composé d'une seule couleur.

I.2.7.2 Images en niveaux de gris

En général, les images en niveaux de gris sont des images de profondeur k=8 bits, donc chaque pixel peut prendre l'une des valeurs de l'intervalle [0…255], où la valeur 0 représente la brillance minimale(le noir) et 255 la brillance maximale (le blanc). Ce type d'image est fréquemment utilisé pour reproduire des photos en noir et blanc ou du texte.

Dans plusieurs applications professionnelles de photographie et d'impression ainsi qu'en médecine et astronomie, 8 bits par pixel n'est pas suffisant, pour cela il existe d'autre types d'images en niveaux de gris de profondeur K=12, K=14 ou K=16 bits.

I.2.7.3 Images couleurs

L'espace couleur est basé sur la synthèse additive des couleurs, c'est à dire que le mélange entre différentes couleurs(trois, quatre…) donne une couleur.

La plupart des images couleurs sont basées sur trois couleurs primaires : Rouge, Vert et Bleu (RVB) (RGB en anglais), et utilisent typiquement 8 bits pour chaque composante de couleur, donc chaque pixel nécessite 3×8=24 bits pour coder les trois composantes, et chaque composante de couleur peut prendre l'une des valeurs de l'intervalle [0…255].

I.2.8 Formats de fichiers d'images

Un format d'image est une représentation informatique de l'image, incluant des informations sur la façon dont l'image est codée et fournissant éventuellement des indications sur la manière de la décoder et de la manipuler.

La plupart des formats sont composés d'un en-tête contenant des attributs (dimension de l'image, type de codage, LUT, etc.), suivi des données (l'image proprement dite). La structuration des attributs et des données diffère d'un format à un autre. Il existe plusieurs formats d'images, nous allons mentionner quelques-uns :

I.2.8.1 BitMaP (BMP)

Le format BMP est l'un des formats les plus simples. Il a été développé conjointement par Microsoft et IBM. Cette technologie a pour principal avantage la qualité des images fournies pas de compression (pas de perte de qualité). Cela fait de lui un format d'image très lourd, pas ou peu utilisé sur Internet.

I.2.8.2 Tagged Image File Format (TIFF)

Ce format est orienté vers les professionnels (imprimeurs, publicitaires...) car il a l'avantage d'être reconnu sur tous types de système d'exploitation : Windows, Mac, Linux, Unix, …etc.

Il permet d'obtenir une image de très bonne qualité, mais sa taille reste volumineuse, même si elle est inférieure à celle du fichier BMP.

I.2.8.3 Joint Photographic Expert Group (JPEG)

C'est le format le plus courant, on le rencontre sur Internet. Il occupe peu d'espace disque. C'est le format développé par les photographes pour transmettre des images de qualité photographique professionnelle. Il gère des millions de couleurs mais il ne possède pas de palette de couleurs associée et donc les couleurs peuvent être différentes sur des machines et des systèmes différents.

I.2.8.4 Graphics Interchange Format (GIF)

Les fichiers au format GIF sont fortement compressés tout en gardant une qualité très correcte. Ils possèdent une palette de couleurs associée (limité à 256 couleurs) et occupent peu d'espace disque.

I.2.8.5 Portable Network Graphic (PNG)

Le format PNG reprend le principe de codage du format GIF mais n'est pas limité à 256 couleurs, et offre une compression généralement plus efficace. Il permet donc contrairement à GIF d'enregistrer des photographies sans perte de qualité, mais avec un gain d'espace de stockage moindre comparativement au format JPEG.

Nous avons utilisé dans notre système des images de format BMP, Pour cela, nous allons décrire la structure d'un fichier BMP.

Conclusion

Les techniques de traitement (signal ou image) sont des techniques très diverses et le choix de l'une parmi elles, est un choix qui dépend essentiellement de la nature de l'application et des résultats qui peuvent être obtenus par l'application de l'une ou de l'autre. Cependant, chaque ensemble de ces techniques est destiné à une application spécifique. Certains ensembles ont des applications communes, mais les résultats obtenus seront différents du point de vue avantages et inconvénients.

CHAPITRE 2

TECHNIQUES DE COMPRESSION DES DONNEES

Introduction

La compression de données, de façon simplifiée, c'est l'ensemble des méthodes que l'on utilise pour prendre un message long pour en faire un message court, sans perdre d'information importante. Nous trouvons de tels usages dans la vie de tous les jours et pas seulement dans les application technologiques. Nous utilisons constamment des abréviations pour prendre des notes manuscrites plus rapidement, nous remplaçons des noms d'items par leur numéros d'inventaire, etc.

Dans ce chapitre, on le scindera en deux phase ; on en citera comme suit : la première phase consacrera pour quelques notions de base sur la compression des données, puis on créera un atmosphère de réflexion en indiquant tout d'abord quelles sont les méthodes générales utilisées aujourd'hui pour compresser les données sans perte (codage de Huffman, codage arithmétique et l'algorithme RLE, LZW, …), En ce qui concerne la deuxième phase qui s'articulera autour la présentation de deux méthodes très utilisées dans la compression des données avec perte, « la quantification scalaire et la quantification vectorielle ».

I. Définition et objectif de la compression

I.1 Définition

La compression de données est un processus de réduction du nombre de bits utilisés pour stocker ou transmettre informations et diminuer l'espace parce que la taille de données est réduite, temps de transmission, et le coût. Données compression est couramment utilisées dans les systèmes modernes de base de données [17]. Plusieurs mesures sont couramment utilisées pour exprimer l'exécution d'un procédé de compression, elles basées sur :

Les différents algorithmes de compression trois critères :

- Le taux de compression : c'est le rapport de la taille du fichier compressé sur la taille du fichier initial.
- La qualité de compression : sans ou avec pertes.
- La vitesse de compression et de décompression.

I.2 Objectif de compression

La compression consiste à réduire la taille physique de blocs d'information. Un compresseur utilise un algorithme qui sert à optimiser les données en utilisant des

considérations propres au type de données à compresser. Un décompresseur est donc nécessaire pour reconstruire les données originelles grâce à l'algorithme inverse de celui utilisé pour la compression.

La méthode de compression déprend intrinsèquement du type de données à compresser, on ne compressera pas de la même façon une image qu'un fichier audio[18].

II. Types de Compression

On considère généralement la compression comme un algorithme capable de comprimer énormément de données dans un minimum de place (compression physique), mais on peut également adopter une autre approche et considérer qu'en premier lieu un algorithme de compression a pour but de recoder les données dans une représentation différente plus compacte contenant la même information (compression logique).

La distinction entre compression physique et logique est faite sur la base de comment les données sont compressées ou plus précisément comment est-ce que les données sont réarrangées dans une forme plus compacte. Afin de gagner de la place, on est amené à comprimer les fichiers, mais le résultat n'est pas toujours idéal. Il faut donc faire un choix en connaissance de cause, il y a deux types de compression :

- La compression avec pertes
- La compression sans pertes

II.1 Compression sans perte

La compression sans perte (lossless) est principalement applicable aux données qui demandent une restitution exacte. Dans le contexte de la compression sans perte, la méthode prend en entrée une série de bits X qu'elle transforme en une nouvelle série de bits Y plus courte que X. La série de bits Y est transmise ou stockée pour usage ultérieur. Lorsque l'on veut récupérer les données, on prend Y et on applique la méthode de compression inverse (la méthode de décompression) pour récupérer X intact. La figure (2.1) illustre schématiquement le processus [20].

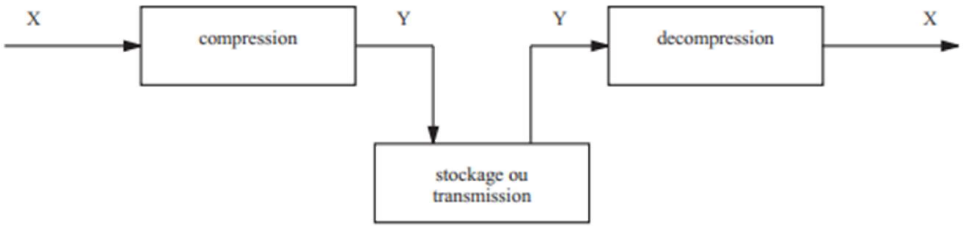


Figure 2.1 : Une vision simplifiée de la compression sans perte [10].

II.2 Compression avec perte

La compression avec perte (on dira lossy, en anglais) joue un rôle tout à fait différent. La compression avec perte permet de choisir, judicieusement on l'espère, quelle information conserver et reconstruire et quelle information simplement détruire. Alors qu'avec la compression sans perte, les données restituées étaient identiques aux données avant la compression, avec la compression avec perte, on acceptera que les données restituées soient une approximation satisfaisante des données originales. Ce que l'on jugera alors comme une approximation satisfaisante dépend du champ d'application considère. La figure 2.1 illustre le processus général. Ici, on compresse une série de bits X pour obtenir la série Y , qui est transmise ou stockée. Lorsqu'on décompresse Y on retourve Z, qui est potentiellement différent de X. Si Z diffère de X, il faudra alors que $Z \approx X$, selon une mesure appropriée[20].

II.3 Compression symétrique et asymétrique

Les algorithmes de compression peuvent être divisés en deux catégories distinctes symétrique et asymétrique. Une méthode de compression symétrique utilise le même algorithme, et demande la même capacité de calcul, aussi bien pour la compression que pour la décompression. Les méthodes de compression asymétriques demandent plus de travail dans une direction que dans l'autre. Normalement, l'étape de compression demande beaucoup plus de temps et de ressources systèmes que l'étape de décompression.

Dans la pratique, cela prend tout son sens : par exemple, imaginons que nous ayons une base de données où les données seront compressées une seule fois mais décompressées un grand nombre de fois pour la consultation, alors on pourra certainement tolérer un temps beaucoup plus grand pour la compression, dans le but de trouver le taux de compression le plus élevé, que pour la décompression, où la vitesse est prédominante. Un algorithme asymétrique qui utilise beaucoup plus de temps CPU pour la compression mais qui est beaucoup plus rapide à la décompression serait un bon choix dans ce cas là.

III. Concepts généraux

III.1 Entropie

En informatique, l'entropie désigne la quantité d'informations que peut représenter ou contenir une source d'information. Ce concept a été introduit en 1948 par Claude Shannon, une source d'information est généralement un fichier informatique quelconque.

Plus une source d'information est redondante, moins elle contient d'information au sens de Shannon. Une source d'information dont tous les symboles sont équiprobables a une entropie maximale.

L'entropie est aussi la plus petite quantité moyenne de bits nécessaire pour communiquer la vraie valeur d'une variable aléatoire. En d'autres mots, l'entropie est la plus petite quantité moyenne de bits capable de représenter l'information tirée de la variable aléatoire. Il s'agit d'une limite mathématique fondamentale pour la compression de données sans perte [15].

L'entropie d'une variable aléatoire discrète X, qui peut prendre les valeurs $\{x_1, x_2,\ldots,x_i\}$, est :

$$H(X) = \sum_{i=1}^{N} p(x_i) \log_2 \left(\frac{1}{p(x_i)}\right) = -\sum_{i=1}^{N} p(x_i) \log_2 p(x_i) \qquad (2.1)$$

Où $p(x_i)$ est la probabilité de x_i.

III.2 Redondance

La redondance est le rapport entre le nombre de bits utilisé pour stocker des données et le nombre de bits minimal nécessaire pour représenter l'information que ces données renferment. La compression de données sans perte a pour objectif de réduire le plus possible la redondance [15].

III.3 Codage

Le codage consiste à donner un mot de code à chaque symbole d'un alphabet donné. Une fonction de codage $C : \sum \rightarrow \{0, 1\}$ Permet de traduire un symbole de l'alphabet $\sum$ en une séquence de bits, ou mot de code. De plus, grâce au théorème de Shannon sur le codage source, nous savons que la longueur optimale d'un mot de code m est log(p), p étant la fréquence d'apparition du symbole représenté par m [15].

IV. Codages sans pertes

IV.1 Codage de Shannon-Fano

C . Shannon du laboratoire Bells et R .M . Fano du MIT ont développé à peu près en même temps une méthode de codage basée sur la simple connaissance de la probabilité d'occurrence de chaque symbole dans un message [14].

Algorithme :

1. Trier les symboles en ordre décroissant de fréquence d'apparition (du plus fréquent au moins fréquent).

2. Diviser Σ en deux sous-ensembles $\sum_1$ et $\sum_2$ en respectant les deux contraintes suivantes:

a) $\sum_1$ doit contenir les n plus fréquents symboles de Σ et $\sum_2$ doit contenir les $|\sum| - n$ moins fréquents symboles de Σ.

b) La différence entre la somme des fréquences d'apparition des symboles de l'ensemble $\sum_1$ et la somme des fréquences d'apparition des symboles de l'ensemble $\sum_2$ doit être la plus petite possible. Formellement, nous cherchons les ensembles $\sum_1 = \{s_1, s_2, \dots, s_{j-1}\}$ et $\sum_2 = \{s_j, s_{j+1}, \dots, s_N\}$ tels que :

$$\arg\_\min_{i \le j \le N} | \sum_{i=1}^{j-1} p(s_i) - \sum_{i=j}^{N} p(s_i). \qquad (2.2)$$

où $N = |\sum|$ et $p(s_i)$ est la fréquence d'apparition de s_i .

3. Attribuer aux symboles dans $\sum_1$ le bit 0 comme premier bit de leurs mots de code et attribuer aux symboles dans $\sum_2$ le bit 1.

Implémentation :

X	*P(X)*	Etape					Code
		1	2	3	4	5	
E	0.4					1	1
A	0.3				0	1	10
D	0.15			0	1	1	110
B	0.1		0	1	1	1	1110
F	0.03	0	1	1	1	1	11110
C	0.02	1	1	1	1	1	11111

Tableau 2.1 : Exemple de Codage de Shannon-Fano

IV.2 Codage de Huffman

Le codage de Huffman est basé sur la fréquence de occurrence d'un élément de données (pixel dans les images). Le principe est d'utiliser un nombre inférieur de bits pour coder les données qui se produit plus fréquemment. Les codes sont stockés dans un dictionnaire de code qui peut être construit pour chaque image ou un ensemble d'images. Dans tous les cas, le dictionnaire de code ainsi que des données codées doivent transmettre pour permettre le décodage [1]. Le principe est le suivant :

1. Répartir les fréquences f_i des lettres.

2. Classer les symboles dans l'ordre décroissant des fréquences d'occurrence. Le résultat de l'algorithme ne change donc pas si l'on remplace les fréquences f_i par les probabilités

$$P_i = f_i / \sum f_i \,. \tag{2.3}$$

3. Regrouper par séquences les paires de symboles de plus faible probabilité, en les reclassant si nécessaire. Plus précisément : calculer $s = f(i_n) + f(i_{n-1})$, la somme des deux plus faibles fréquences.

4. Choisir le plus petit indice k tel que s soit supérieur ou égal à $f(i_k)$, remplacer k par $k + 1$.

5. Recomposer la table des fréquences en plaçant à la k^{me} position la valeur s et en décalant les autres d'une position vers le bas. Puis décrémenter n d'une unité, poursuivre jusqu'à ce que la table des fréquences ne comporte plus que deux éléments.

6. Coder avec retour arrière depuis le dernier groupe, en ajoutant un 0 ou un 1 pour différencier les symboles préalablement regroupés [2].

Implémentation :

Imaginons que nous ayons compté la fréquence des lettres dans un texte :

M	A	C	E	Espace	H	O	N	T	R
3	2	2	2	2	1	1	1	1	1

On peut construire l'arbre binaire de la figure 2.1.

En Huffman

M	A	C	E	Espace	H	O	N	T	R
00	001	011	010	110	0111	1111	0101	01101	11101

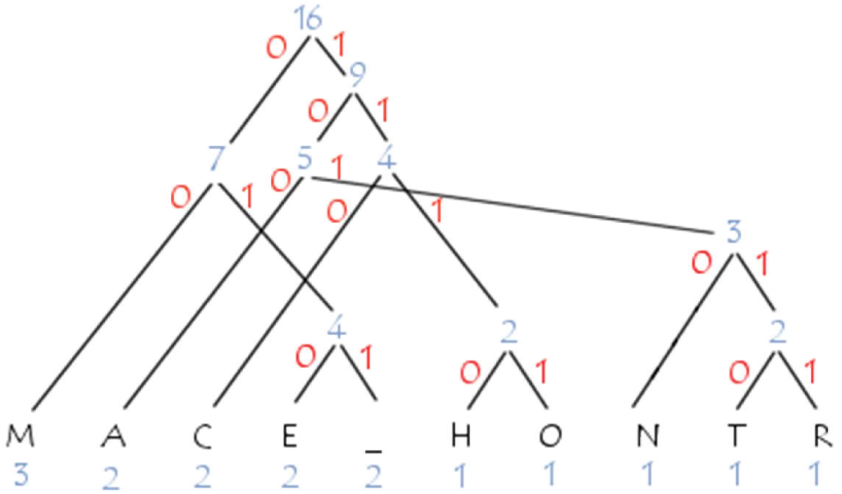


Figure 2.2 : Arbre de Huffman.

IV.3 Codage arithmétique

Le codage arithmétique se base sur l'idée que la somme des fréquences d'apparition de tous les caractères doit être égale à 1. En conséquence chaque fréquence d'apparition doit être contenue dans un certain sous-intervalle de l'intervalle [0,1]. Il faut donc seulement indiquer le sous-intervalle où est localisée la fréquence d'apparition de chaque caractère pour coder la séquence de caractères. Le nombre de bits de la représentation de chaque intervalle doit être suffisamment grand pour identifier sans confusion l'intervalle correspondant [13].

Algorithme du codage arithmétique :

1. Calculer la probabilité associée à chaque symbole dans la chaîne à coder.
2. Associer à chaque symbole un sous intervalle proportionnel à sa probabilité, dans l'intervalle [0,1] (l'ordre de rangement des intervalles sera mémorisé car il est nécessaire au décodeur).
3. Initialiser la limite inférieure de l'intervalle de travail à la valeur 0 et la limite supérieure à la valeur 1.
4. Tant qu'il reste un symbole dans la chaîne à coder :

- largeur = limite supérieure - limite inférieure,
- limite inférieure = limite inférieure + largeur×(limite basse du sous intervalle du symbole),
- limite supérieure = limite inférieure + largeur×(limite haute du sous intervalle du symbole),

5. La limite inférieure code la chaîne de manière unique [14].

Implémentation :

Soit, par exemple, à coder la séquence X="BILL GATES". En appliquant l'algorithme du codage arithmétique sur la séquence X, on obtiendra l'évolution suivante:

Symbole	Bas	Haute
B	0.2	0.3
I	0.25	0.26
L	0.256	0.258
L	0.2572	0.2576
Espace	0.25720	0.25724
G	0.257216	0.257220
A	0.2572164	0.2572168
T	0.25721676	0.2572168
E	0.257216772	0.257216776
S	0.2572167752	0.2572167756

Tableau 2.2 : Exemple de codage arithmétique

Donc 0.2572167752 est la représentation arithmétique du message "BILL GATES".

IV.4 LZW

La variante la plus populaire du LZ78 est l'algorithme LZW (Lempel, Ziv, Welch) créé en 1984. Le principe est de réduire l'envoi d'informations au décodeur. Au lieu du couple (i,c),seul l'index (i) sera transmis.

Compression LZW remplace des chaînes de caractères avec codes simples. Il ne fait aucune analyse de l'arrivée texte. Au lieu de cela, il ajoute juste chaque nouvelle chaîne de caractères, il voit à une table de chaînes. La compression se produit lorsqu'un seul code est sorti au lieu d'une chaîne de caractères.

Le code que les sorties de l'algorithme LZW peuvent être de tout longueur arbitraire, mais il doit avoir plus de bits à ce qu'un un seul caractère. Les 256 premiers codes (en utilisant huit caractères binaires) sont par défaut attribué à la norme jeu de caractères. Les codes restants sont attribués à des chaînes que l'algorithme passe. Le programme

d'échantillonnage fonctionne comme montré avec 12 codes binaires. Cela signifie que les codes se réfèrent à (0-255) octets individuels, tandis que les codes (256-4095) réfèrent à substrings.

Compression LZW qui fonctionne le mieux pour les fichiers contenant beaucoup de données répétitives. Ce qui est souvent le cas avec du texte et images monochromes. Les fichiers sont compressés, mais qui ne contient aucune information répétitive du tout peut même se développer plus gros [16].

Algorithme :

- On initialise la table (le dictionnaire) en y plaçant les codes des caractères ASCII étendu,
- On regarde le caractère à transmettre :
  s'il existe déjà dans la table, on regarde le caractère suivant,
  si le groupe des deux existe également, on regarde le suivant, etc.
- Lorsqu'un nouveau groupe est découvert, on le définit en l'insérant dans le dictionnaire. Dans un premier temps, on transmet les codes des morceaux qui le composent. La prochaine fois qu'on le rencontrera, on transmettra son code propre. Les mots ajoutés au dictionnaire seront déterminés par l'intermédiaire d'une "fenêtre" évoluant au fil de l'analyse du texte à compresser. Ce concept est explicité dans l'exemple ci-dessous.
- On procède de la sorte jusqu'à la fin de la transmission.
- Lorsque le dictionnaire est plein, soit on procède à son extension, soit on se borne à utiliser les codes déjà existants.

* Comment se construit le dictionnaire ?

Le principe est d'utiliser une fenêtre grandissant jusqu'à l'obtention d'un mot inexistant dans le dictionnaire. Auquel cas le mot en question est ajouté, le code du préfixe connu envoyé, et la fenêtre ramenée au dernier caractère analysé.

Pour plus de clarté, soit l'exemple suivant, sachant que les 256 premières entrées du dictionnaire sont initialisées avec les 256 valeurs du code ASCII étendu :

– Imaginons le message "ma maison" a compressé.

1. Le premier caractère a analysé est "m". Il est dans le dictionnaire.

2. Le caractère suivant est concaténé avec "m" et forme la chaine "ma". Elle n'est pas dans le dictionnaire : il faut donc l'ajouter. On lui affecte donc l'entrée 256 dans le dictionnaire (on commence à 0). On envoie la valeur correspondante à "m"(109).
3. Un nouveau mot venant d'être ajouté, on reprend alors au dernier caractère pris en compte (ici "a"). Le caractère suivant est concaténé et forme la chaine "a ". Elle n'est pas dans le dictionnaire : il faut donc l'ajouter. On lui affecte donc l'entrée 257 dans le dictionnaire. On envoie la valeur correspondante à "a"(97).
4. Un nouveau mot venant d'être ajouté, on reprend alors au dernier caractère pris en compte (ici " "). Le caractère suivant est concaténé et forme la chaine " m". Elle n'est pas dans le dictionnaire : il faut donc l'ajouter. On lui affecte donc l'entrée 258 dans le dictionnaire. On envoie la valeur correspondante à " "(32).
5. Un nouveau mot venant d'être ajouté, on reprend alors au dernier caractère pris en compte (ici "m"). Le caractère suivant est concaténé et forme la chaine "ma". Cette chaine est déjà dans le dictionnaire. On concatène le caractère suivant et cela forme la chaine "mai". Elle n'est pas dans le dictionnaire : il faut donc l'ajouter. On lui affecte donc l'entrée 259 dans le dictionnaire. On envoie donc la valeur correspondante à "ma"(256).
6. Un nouveau mot venant d'être ajouté, on reprend alors au dernier caractère pris en compte (ici "i"). Le caractère suivant est concaténé et forme la chaine "is". Elle n'est pas dans le dictionnaire : il faut donc l'ajouter. On lui affecte donc l'entrée 260 dans le dictionnaire. On envoie la valeur correspondante à "i"(105).
7. Un nouveau mot venant d'être ajouté, on reprend alors au dernier caractère pris en compte (ici "s"). Le caractère suivant est concaténé et forme la chaine "so". Elle n'est pas dans le dictionnaire : il faut donc l'ajouter. On lui affecte donc l'entrée 261 dans le dictionnaire. On envoie la valeur correspondante à "s"(115).
8. Un nouveau mot venant d'être ajouté, on reprend alors au dernier caractère pris en compte (ici "o"). Le caractère suivant est concaténé et forme la chaine "on". Elle n'est pas dans le dictionnaire : il faut donc l'ajouter. On lui affecte donc l'entrée 262 dans le dictionnaire. On envoie la valeur correspondante à "o"(111).
9. Un nouveau mot venant d'être ajouté, on reprend alors au dernier caractère pris en compte (ici "n"). Le caractère suivant est concaténé et forme la chaine "n(eof)". Elle n'est pas dans le dictionnaire : il faut donc l'ajouter. Etant donné le caractère spécifique de fin de fichier, il n'est pas ajouté au dictionnaire. On envoie la valeur correspondante à "n"(110).

– Au final, le flux compressé sera (109)(97)(32)(256)(105)(115)(111)(110).
– Lors de la décompression, le décodage se fera de manière inverse.

IV.5 Méthodes de plages (Compression Run Length Encoding)

Le RLE (Run-length encoding, codage par plages) est probablement l'algorithme de compression de données le plus simple à ce jour. Il ne fait que remplacer plusieurs apparitions d'un même symbole par un exemplaire du symbole et le nombre de fois qu'il apparait consécutivement. Par exemple, si nous avons une série de A dans un document : AAAAA, RLE remplace la sérié de A par un message du type A5. Evidemment, il faut un moyen pour différencier le message A5 de la séquence A5 dans les données originales. Une séquence d'échappement, c'est à dire un nouveau symbole que l'on introduit dans notre alphabet, peut être utilisée à cette fin. Il suffit alors de transmettre la séquence d'échappement avant de transmettre le nombre de fois qu'un symbole apparait, le cas échéant [15].

Implémentation :

Texte à compresser : Gooooooooooooogle
Texte compressé : G@15ogle
La taille du texte initial est de 20 caractères tandis que celle du fichier compressé est de 9 caractères.

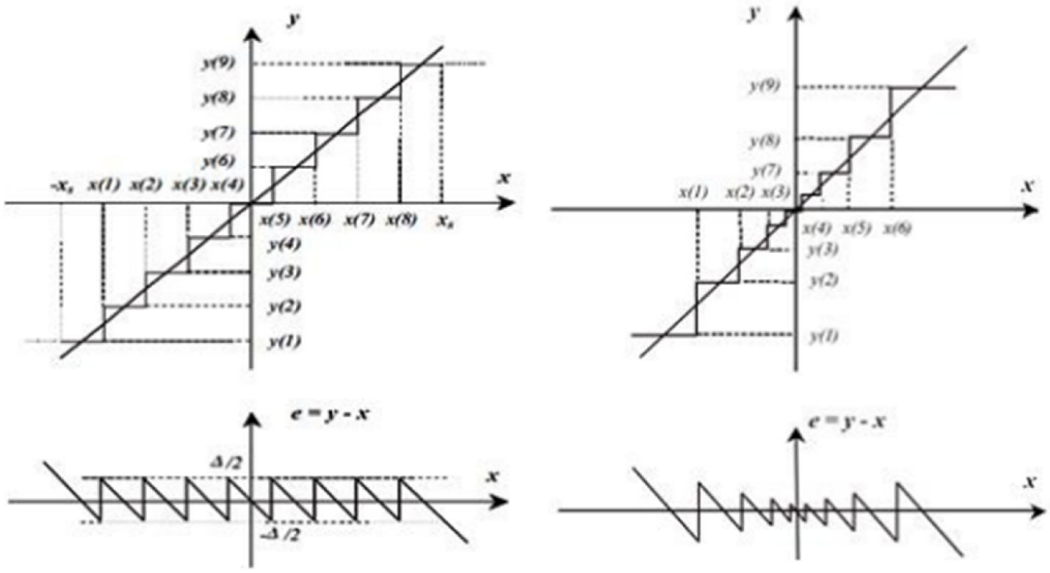
V. Techniques de compression avec perte

Afin d'augmenter les performance de compression des signaux numériques, une solution consiste à imposer une réduction sur le nombre de symboles de l'alphabet en entrée du processus de codage entropique. Cette réduction peut s'effectuer par quantification scalaire : tout symbole de l'alphabet initial (ensemble des valeurs numériques possibles pour la représentation dans le domaine image ou transformé, ou encore pour les résidus de prédiction) est associé à un nouveau symbole dans un alphabet plus petit. Il est également possible d'utiliser une quantification vectorielle : une séquence de symboles en entrée du système est associée à l'index de la séquence la plus similaire dans un dictionnaire de séquences représentatives.

Dans ces deux situations, une opération surjective est effectuée et effectuée et las symboles ou séquences originaux ne pourront pas être récupérés. Il faut alors chercher à optimiser les pertes d'informations[19].

V.1 Quantification scalaire

La quantification scalaire est l'approximation de chaque valeur du signal aléatoire x(t), par une valeur qui appartient à un ensemble fini de codes$\{y_1, y_2, \dots, y_l\}$. Pour toute amplitude x comprise dans l'intervalle $[x_{l-1}, x_l]$, on fait correspondre une valeur quantifiée y_l située dans cet intervalle [21].



(a) : quantification uniforme (b) : quantification non uniforme

Figure 2.3 : Exemples de quantification scalaire .

Une quantification scalaire est caractérisée par :

• Nombre de niveaux quantifiés L+1=2^b (b bits).

• Pas de quantification : $\Delta = x_i - x_{i-1}$ (2.4)

• Erreur de quantification : $e = y - x$ (2.5)

• Erreur de granulation : $|e| \leq \frac{\Delta}{2}$ (2.6)

V.2 Quantification vectorielle

Les techniques de compression d'images exploitent généralement la redondance statistique présente dans l'image. La quantification scalaire qui associe à une variable continue une variable discrète pouvant prendre un nombre plus faible, et fini de valeurs. Ces valeurs ne sont jamais totalement décorrélées, ou indépendantes. Shannon a montré qu'il était toujours possible d'améliorer la compression de données en codant des vecteurs plutôt que des scalaires [14].

La Quantification Vectorielle (QV), développée par Gersho et Gray a pris une place très importante dans le domaine de la compression d'image que ce soit dans le but de transmission ou d'archivage.

- **Principe de la quantification vectorielle**

La quantification vectorielle, dans son sens le plus général, est l'approximation d'un signal d'amplitude continue par un signal d'amplitude discrète . Elle peut être vue comme une application Q associant à chaque vecteur d'entrée x de dimension K, un vecteur $y = Q(x)$ de même dimension appartenant à un ensemble fini Y appelé DICTIONNAIRE de taille finie $N, Y = (y_j , j = 1, \dots, N)$. Elle se décompose en deux applications : codeur, décodeur [14].

- **Codeur**

Le rôle du codeur consiste, pour tout vecteur x du signal en entrée, à rechercher dans le dictionnaire Y le code vecteur y_j le plus proche du vecteur source x. C'est uniquement l'adresse du code vecteur y_j ainsi sélectionnée qui sera transmise ou stockée. C'est à ce niveau donc que s'effectue la compression.

- **Décodeur**

Il dispose d'une réplique du dictionnaire et consulte celui-ci pour fournir le code vecteur d'indice correspondant à l'adresse reçue. Le décodeur réalise l'opération de décompression.

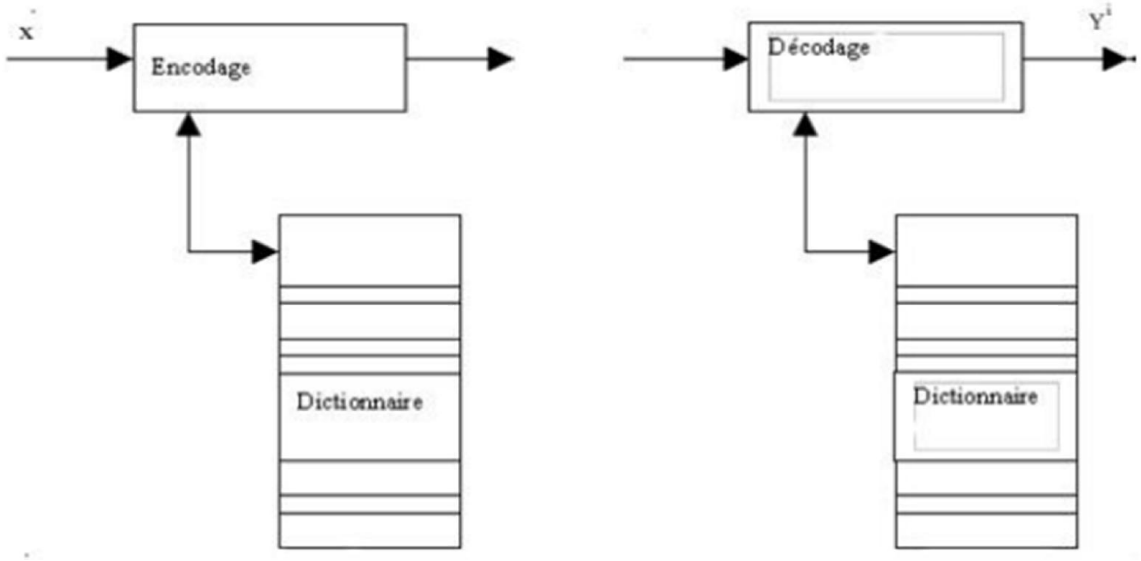


Figure 2.4 : Schéma synoptique d'un quantificateur vectoriel.

Conclusion

La compression des données est appelée à prendre un rôle encore plus important en raison du développement des réseaux et du multimédia. Les méthodes déjà utilisées couramment sont efficaces et sophistiquées (Huffman, LZW,…,etc.).

On a présenté au sein du chapitre les techniques de compression des données (méthodes directes): une introduction aux concepts de base sur la compression des données et les techniques de compression des données sans et avec perte.

Le prochain chapitre donne une description simplifiée de l'une des transformées à savoir la transformation DCT (Discrete Cosine Transform).

CHAPITRE 3

TRANSFORMEE DCT

Introduction

Dans ce chapitre, nous allons basé sur la DCT (*Discrete Cosine Transform*). Nous présentons son principe ainsi que ces propriétés.

I. Transformation DCT

Ahmed, Natarajan et Rao ont été les premiers à mettre en équation DCT en 1974. C'est une transformation mathématique qui transforme un ensemble de données d'un domaine spatial en spectre de fréquence et inversement par la IDCT (*Inverse Discret Cosine Transforme*).

le transforme en cosinus discrète est une technique pour convertir un signal élémentaire en les composantes de fréquence. Il représente une image en tant que somme de sinusoïdes de différentes grandeurs et les fréquences.

Le DCT à base de bloc populaire transformée segments une image non-chevauchement des blocs et applique DCT chaque bloc. Il en résulte un préavis de trois fréquences sous-bandes: sous-bande basse fréquence, mi-frequency sub-band et de sous-bande haute fréquence. filigranage base-DCT est basé sur deux faits. Le premier fait est qu'une grande partie de l'énergie du signal se trouve à basses fréquences sous-bande qui contient le plus important visuelle les parties de l'image. Le second fait est que la haute les composantes de fréquence de l'image sont habituellement éliminé par compression et de bruit attaques. Le filigrane est donc incorporé en modifiant la coefficients de la sous-bande de fréquence moyenne de sorte que la visibilité de l'image ne sera pas affectée et le filigrane ne sera pas éliminé par compression[22].

II. Pourquoi la DCT?

On a vu que la DCT était dans la même classe d'outils mathématiques que la Transformée de Fourier. Alors pourquoi les membres du groupe JPEG ont-ils fait le choix de la DCT? Ces deux méthodes permettent une décomposition de l'information dans une autre base : Une base de cosinus, ou la base de Fourier. Cependant, La décomposition dans la base de Fourier soulève plusieurs problèmes : si l'image présente des discontinuités, alors la

décroissance des coefficients de la transformée de Fourier n'est qu'en $\frac{1}{k}$, K étant l'indice du coefficient.

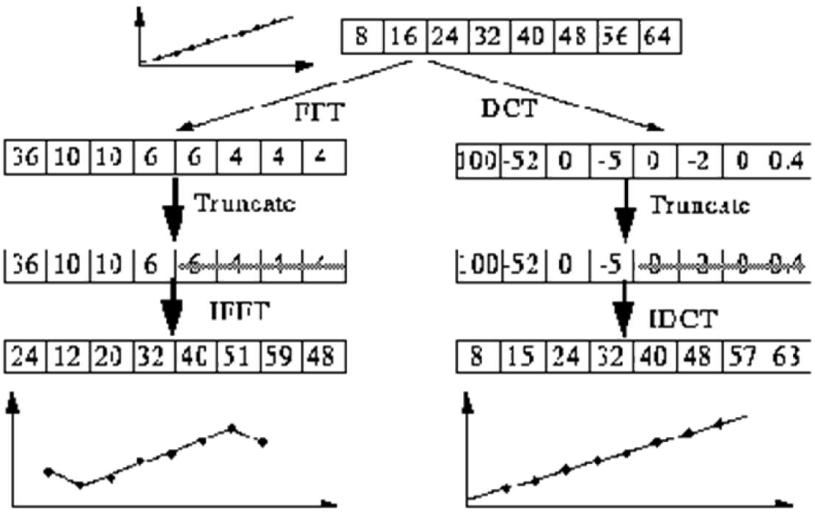


Figure 3.1 : Décroissance des coefficients

Dans cette figure, on se rend compte que pour restituer convenablement l'information, on a besoin de beaucoup plus de coefficients que pour une DCT. La décroissance des coefficients n'étant pas suffisante pour pouvoir négliger rapidement les coefficients de grands indices. De plus en tronquant les derniers coefficients, on risque de voir se produire à la fin le phénomène de Gibbs, qui se traduit par une oscillation au niveau des discontinuités.

D'autre part, la fonction doit être périodique pour la transformée de Fourier, sinon on se retrouve avec une discontinuité au bord. Là encore se posera le phénomène de Gibbs qui risque fort de dégrader l'image.

Le fait de décomposer la fonction sur une base de cosinus fait que la fonction sera symétrique par rapport à $-\frac{1}{2}$. Cependant la DCT pose un problème d'optimisation. En effet le calcul d'un coefficient nécessite N^2 multiplications, or il y a N^2 coefficients à calculer. Le coût d'une telle décomposition devient alors démeusuré si notre image est de taille 512x512. Ainsi, au lieu de traiter toute l'image, on découpe celle-ci en blocs 8x8. Ce choix représente un compromis performance qualité : en effet, en augmentant la taille de ces blocs, la compression serait meilleure, mais le coût en temps a été jugé trop grand. Sur chacun de ces blocs, on procède ensuite à une DCT.

Cela permet d'avoir un algorithme rapide. Cependant la DCT par blocs 8x8 est justement un des facteurs limitant de la compression JPEG : en effet lorsqu'on augmente la compression, on voit apparaitre ces blocs.

III. Transformée en cosinus discrète unidimensionnelle

La définition la plus courante (DCT unidimensionnelle(1-D)) d'une séquence $f(i)$ de N échantillons est donnée par l'équation (3.1):

$$F(k) = \alpha(k) \sum_{i=0}^{N-1} f(i) \cos\left[\frac{\pi(2i+1)k}{2N}\right] \quad (3.1)$$

Avec $k = 0,1,2,\dots,N-1$, et $\alpha(k)$ est définit comme :

$$\begin{cases} \sqrt{\frac{1}{N}} & pour\ k=0 \\ \sqrt{\frac{2}{N}} & pour\ k\neq 0 \end{cases} \quad (3.2)$$

La transformée inverse de la DCT-1D a pou équation :

$$f(i) = \sum_{i=0}^{N-1} \alpha(k) F(k) \cos\left[\frac{\pi(2i+1)k}{2N}\right] \quad (3.3)$$

Il est clair d'après (3.1) et (3.2) que pour $k = 0$,

$$F(k=0) = \sqrt{\frac{1}{N}} \sum_{i=0}^{N-1} f(i) \quad (3.4)$$

Ainsi, le coefficient de transformation est d'abord la valeur moyenne de la séquence d'échantillons. Dans la littérature, cette valeur est désignée sous le nom de Coefficient DC (pour Direct Component) ou coefficient continu. Tous les autres coefficients de transformation sont appelés les coefficients AC (Alternative Component) qui représentent les amplitudes des fréquences spatiales.

Le terme de $\sum_{i=0}^{N-1} \cos\left[\frac{\pi(2i+1)k}{2N}\right]$ pour $N = 8$ et en variant de i est montré sur la figure (3.2). Suivant l'observation précédente, la première forme d'onde en dessus-gauche $k = 0$ rend une valeur constante(DC), alors que, toutes les autres formes d'onde ($k = 1,2,\dots,7$) donner des formes d'onde à l'augmentation progressive des fréquences [23] .

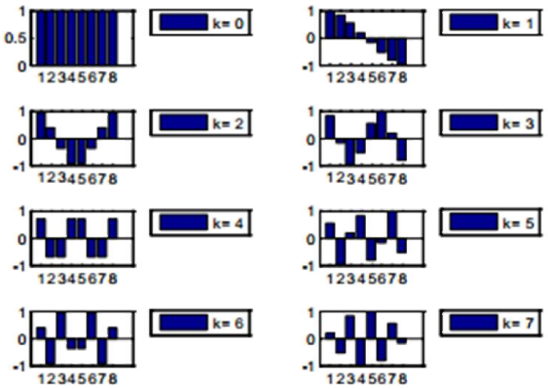


Figure 3.2 : Fonctions de base de la DCT-1D

IV. Transformée en cosinus discret bidimensionnelle

La DCT-1D est utilisée en traitement des signaux unidimensionnels tels que les signaux de la parole. Pour l'analyse d'un signal bidimensionnel (2D) comme les images, on a besoin d'une version 2D de la DCT.

La DCT-2D est effectuée sur une matrice carrée $N \times N$ de pixels et donne une matrice carrée $N \times N$ de coefficients fréquentiels. Comme pour la DCT-1D, l'élément (0,0) est appelé la composante DC et les autres éléments sont les composantes AC.

Par convention, les 64 valeurs transformées (de chaque bloc de 8×8) sont positionnées d'une certaine manière, ainsi la valeur moyenne de tous ces coefficients est placée en haut à gauche de ce bloc. Plus on s'éloigne des coefficients continues plus leurs grandeurs tendent à diminuer, ce qui signifie que la DCT concentre l'énergie d'image en haut à gauche de la matrice transformée, les coefficients en bas et à droite de cette matrice contiennent moins d'information utile.

On peut représenter la distribution des fréquences de la DCT d'une matrice de 8×8 éléments par la figure (3.3).

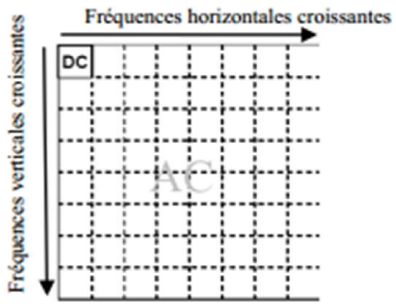


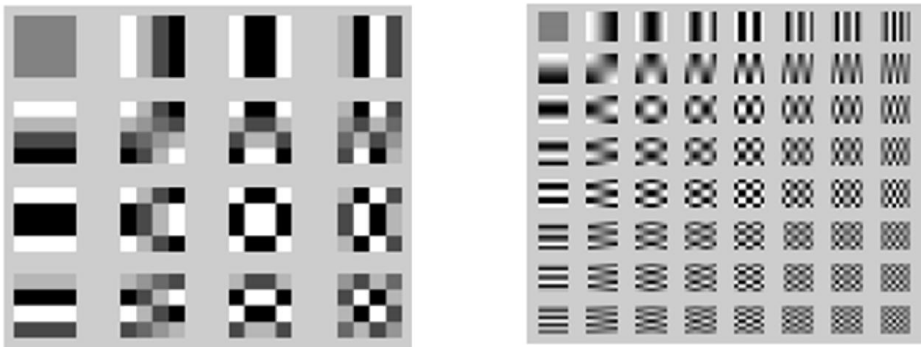
Figure 3.3 : distribution des fréquences de la DCT

Les deux équation qui suivent donnent, respectivement, la DCT-2D directe et inverse[24] :

$$y(u,v) = \sqrt{\frac{2}{M}}\sqrt{\frac{2}{N}}\alpha_u\alpha_v \sum_{u=0}^{M-l}\sum_{v=0}^{N-l} x(m,n)cos\frac{(2m+1)u\Pi}{2M}cos\frac{(2n+1)v\Pi}{2N} \quad (3.5)$$

Avec $k, m = 0,1,2,\dots,N-1$

$$x(m,n) = \sqrt{\frac{2}{M}}\sqrt{\frac{2}{N}}\sum_{u=0}^{M-l}\sum_{v=0}^{N-l}\alpha_u\alpha_v y(u,v)cos\frac{(2m+1)u\Pi}{2M}cos\frac{(2n+1)v\Pi}{2N} \quad (3.6)$$



(a) (b)

(a) : Pour $N = 4$ (b) : Pour $N = 8$

Figure 3.4 : Fonction de base de la DCT-2D

V. Propriétés de DCT

Dans les sections précédentes, ont mis au point une base mathématique pour DCT. Cependant, la vision intuitive dans son application de traitement d'image n'a pas été présentée. Cette section décrit (avec des exemples) quelques propriétés du DCT qui ont une valeur particulière pour les applications de traitement d'image.

IV.1. Décorrélation de DCT

Comme indiqué précédemment, l'avantage principal de la transformation de l'image est la suppression de la redondance entre les pixels voisins. Cela conduit à des coefficients de transformation non-corrélés qui peuvent être codés de manière indépendante.

La figure (3.5) pour décrire les caractéristiques de décorrélation de la 2-D DCT. L'auto-corrélation normalisée des images avant et après DCT est représenté sur la figure (3.5) Il est clair que l'amplitude de l'auto-corrélation après l'opération de TCD est très faible à tous les retards. Par conséquent, on peut en déduire que DCT présente d'excellentes propriétés de décorrélation.

Chaque point représente un pixel dans l'image, avec la coordonnée 'x' présente son niveau de gris et la coordonnée 'y' présente le niveau de gris de son voisin droit. La relation

diagonale forte pour la ligne x=y figure (3.6(a)) montre clairement la corrélation forte entre les pixels voisins et lorsqu'on applique la transformation DCT sur l'image entière, le résultat est montré surla figure (3.6(b)), les deux pixels sont décorrélés, i.e. savoir la valeur du premier pixel n'aide pas l'estimation de la valeur du premier pixel n'aide pas l'estimation de la valeur du second. Par conséquent, la DCT présente une excellente propriété de décorrélation.

Figure 3.5 : Image aux niveaux de gris, 'Lena.bmp'.

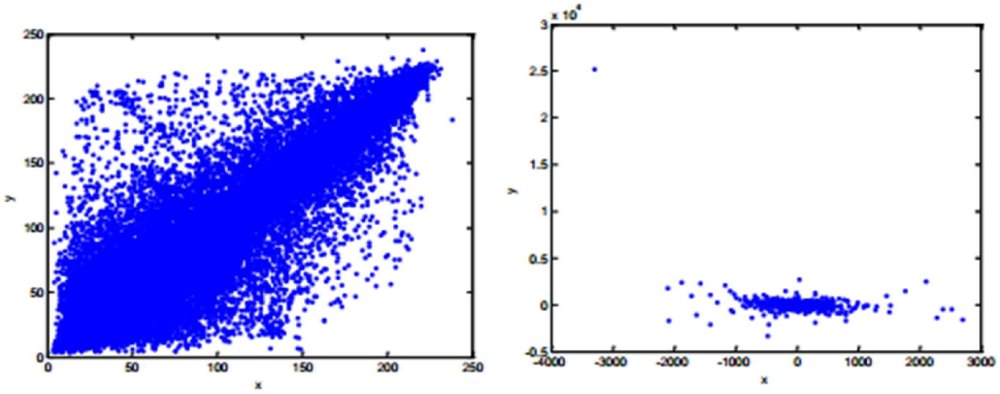
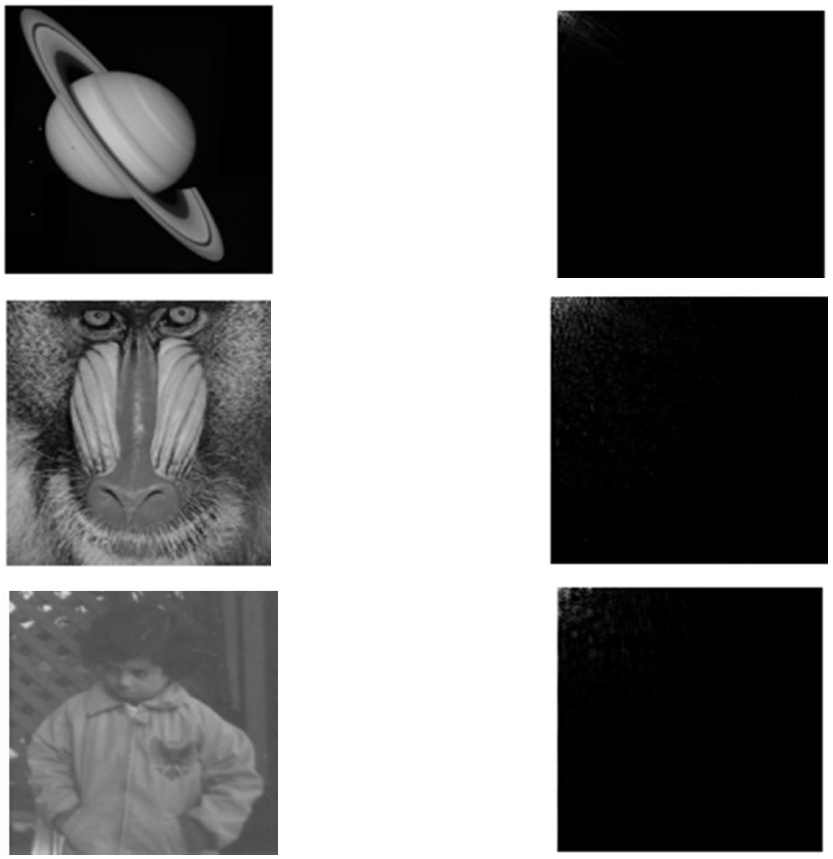


Figure 3.6 : Les valeurs de niveau de gris des paires de pixels adjacents
(a) : Avant la DCT (b) : Après la DCT.

V.2. Comptage d'énergie

L'efficacité d'une transformation peut être directement mesurée par sa capacité à emballer des données d'entrée en tant que coefficients peu que possible. Cela permet au quantificateur de se défaire des coefficients avec relativement faibles amplitudes sans introduire de distorsion visuelle dans l'image reconstruite. DCT présente une excellente compactage d'énergie pour les images fortement corrélées.

Considérons à nouveau les deux exemples d'images de Figure (3.6) (a) et (b). En plus de leurs propriétés de corrélation respectives discutées dans les sections précédentes, l'image décorrélées a plus de variations d'intensité vives que l'image corrélée. Par conséquent, le premier a plus de contenu à haute fréquence de celui-ci. La figure (3.7) montre le DCT des deux images. Il est clair que l'image non corrélés a son énergie étalée, tandis que l'énergie de l'image corrélative est emballé dans la région à basse fréquence.



(a) (b)

(a) Saturn et sa DCT, (b) Baboon et sa DCT, (c) child et sa DCT.

Figure 3.7 : Compactage d'énergie de la DCT

V.3 Séparabilité

L'équation (3.6) de la DCT peut être exprimée comme[24] :

$$F(k,m) = \alpha(k)\alpha(m)\sum_{i=0}^{N-1} cos\left[\frac{\pi(2i+1)k}{2N}\right]\sum_{j=0}^{N-1} f(i,j)cos\left[\frac{\pi(2j+1)m}{2N}\right] \quad (3.7)$$

$$F(k,m) = \alpha(k)\sum_{i=0}^{N-1} G(i,j)cos\left[\frac{\pi(2i+1)k}{2N}\right] \quad (3.8)$$

$$F(k) = \alpha(m)\sum_{j=0}^{N-1} f(i,j)cos\left[\frac{\pi(2j+1)m}{2M}\right] \quad (3.9)$$

Cette propriété, connue sous le nom séparabilité, présente l'avantage de principe C (u, v) est calculé en deux étapes successives, par les opérations 1-D sur les lignes et les colonnes d'une image. Cette idée est illustrée graphiquement sur la figure (3.8). Les arguments présentés peuvent être identiques et appliqués pour le calcul inverse DCT.

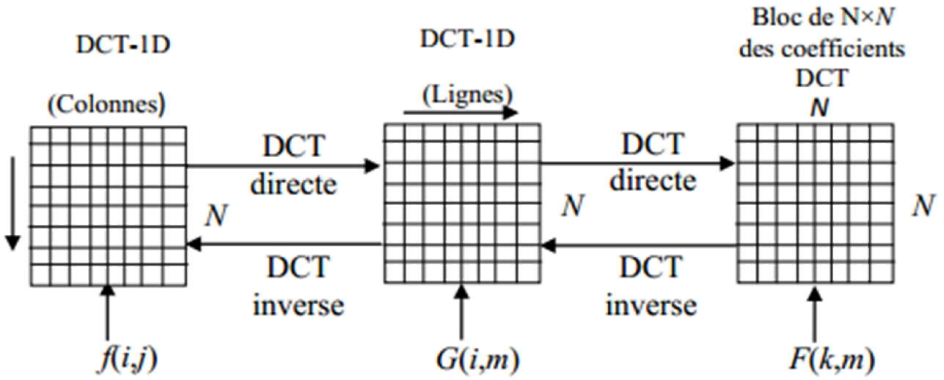


Figure 3.8 : Calcul de la DCT-2D en utilisant la propriété de séparabilité.

V.4. Symétrie

Un autre regard sur les opérations de ligne et de colonne dans l'équation (3.6) réveils que ces opérations sont fonctionnellement identiques. Une telle transformation est appelée une transformation symétrique. Un séparable et symétrique transformation peuvent être exprimés sous la forme :

$$F = T_N \times f \times T_N^t \tag{3.10}$$

Ou **F** est la matrices transformée, **f** est la matrice d'image original et T_N est la matrice de la transformée de taille $N \times N$ avec des élément $T_N(i,j)$ donné par l'équation (3.11)[24].

$$T_N = \begin{cases} \frac{1}{\sqrt{N}} \\ \sqrt{\frac{2}{N}} \cos\left[\frac{(2j+1)i\pi}{2N}\right] \end{cases} \tag{3.11}$$

Ceci est une propriété extrêmement utile car elle implique que la matrice de transformation peut être déconnecté pré-calculé puis appliqué à l'image fournissant ainsi des ordres de grandeur amélioration de l'efficacité de calcul.

V.5 Orthogonalité

Afin d'étendre les idées présentées dans la section précédente, notons la transformation inverse de l'équation (3.10).

$$T_N \times T_N^t = I_N \tag{3.12}$$

Comme indiqué précédemment, les fonctions de base DCT sont orthogonales. Ainsi, l'inverse matrice d'Ais de transformation égale à sa transposée à savoir. Par conséquent, et en plus de ses caractéristiques de décorrélation, cette propriété rend une certaine réduction dans la complexité du pré-calcul.

Conclusion

On a vu dans ce chapitre l'intérêt et les différents types de transformation DCT. Ainsi, on a constaté que la DCT exploite les redondance inter-pixels pour produire une excellent décorrélation. En outre, la DCT

CHAPITRE 4

SIMULATIONS ET RESULTAT

Introduction

Dans ce chapitre, on va évaluer les performances des techniques classiques de compression et de décompression dans divers applications : textes, signaux et images. Ceci pour conclure le domaine d'application de chacune d'elles.

Notre étude consiste à développer chacune des techniques et ensuite l'applications de ces dernières pour différents types d'informations à savoir : un texte bien choisi, un signal médical (Electrocardiogramme : ECG) et des images bien connus dans le domaine de la compression.

Le critère d'évaluation utilisé pour le texte est le taux de compression seulement (information qui exige l'utilisation d'une compression sans perte). Pour le signal ECG les critères sont le taux de compression et le PRD. Pour les image le taux de compression et le PSNR.

IV.1 Critères de mesures des performances

IV.1.1 Taux de compression

Une mesure courante pour déterminer le degré de compression obtenu est le taux de compression CR . Il est défini par:

$$CR = \frac{Nomber\ de\ bits\ de ficher\ originale}{Nomber\ de\ bits\ de ficher\ compressé} \tag{4.1}$$

La théorie de l'information donne une limite théorique au CR maximal qu'il est possible d'atteindre sans distorsion pour n'importe quelle méthode de compression. Les propriétés statistiques des informations originales jouent un rôle prépondérant dans le résultat obtenu. Par exemple avec les images où il existe une forte redondance, il est facile d'obtenir des taux de compression élevés.

IV.1.2 Mesures de distorsion

La distorsion (D) est l'erreur introduite par l'opération de compression, due au fait qu'éventuellement l'image reconstruite n'est pas exactement identique à l'image originale. La mesure de distorsion utilisée généralement en compression d'image est l'erreur quadratique

moyenne (EQM : MSE (Mean Square Error)). Cette grandeur est définie par la moyenne des écarts au carré entre le pixel (i,j) de l'image originale$I_o(i,j)$, et le pixel (i,j) de l'image reconstruite $I_r(i,j)$ [25]. Elle définit par l'équation (4.2) :

$$EQM = \frac{1}{M \cdot N} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} (I_o(i,j) - I_r(i,j))^2 \quad (4.2)$$

IV.1.3 Rapport signal à bruit en pic

Un rapport signal sur bruit crête pour une image dont le maximum est $2^R - 1$ dénoté PSNR (*Pic Signal to Noise Ratio*)[25], il se mesure en décibel (dB) :

$$PSNR = 10 \cdot \log_{10} \frac{(2^R-1)^2}{MSE} \quad (4.3)$$

Où R représente le nombre de bits désignés pour un pixel. Quelques auteurs définissent le PSNR en fonction de RMSE (*Root Mean Square Error)* comme [26] :

$$PSNR = 20 \cdot \log_{10} \frac{(2^R-1)}{RMSE} \quad (4.4)$$

La valeur typique de PSNR varie entre 20 db et 40 db [26].

Sachant que pour une même méthode de compression et un même CR réalisés sur des images distinctes, la qualité obtenue (PSNR) peut être très variable d'une image à l'autre.

IV.1.4 Taux de distorsion

La distorsion est l'erreur introduite par l'opération de compression, due au fait que le signal reconstruit n'est pas exactement identique au signal original. La mesure de la fidélité de la reconstruction PRD est définie comme suit [27]:

$$PRD = \sqrt{\frac{\sum(x(n)-y(n))^2}{\sum x(n)^2}} \times 100\% \quad (4.4)$$

Où $x(n)$ et $y(n)$ représentent respectivement le signal original et le signal reconstruit avec N échantillons.

IV.2 Applications utilisées

II.2.1 Présentation du texte

Les techniques classiques Huffman, arithmétique, LZW et RLE vont être appliqués au fichier texte de 242 caractères suivant :

"La distorsion est l'erreur introduite par l'opération de compression, due au fait que le signal reconstruit n'est pas exactement identique au signal original. La mesure de la fidélité de la reconstruction PRD est définie comme suit"

II.2.2 Présentation du signal ECG

L'électrocardiographie (ECG) est la représentation graphique du potentiel électrique qui commande l'activité musculaire du cœur. Ce potentiel est recueilli par des électrodes mises à la surface de la peau. L'électrocardiographe est constitué en général de :

- Un ensemble d'électrodes destinées à être appliquées en contacte directe du patient.
- Un système d'amplification des signaux issus des électrodes.
- Un appareil enregistreur.
- Un système d'enregistrement graphique [28].

Le signal ECG est le signal 100.dat extrait de la base de données arrythmia [29]. C'est un signal échantillonné à 360 échantillons par seconde et de résolution 12 bits/échantillon. La figure 4.2 montre l'allure de 4 secondes de ce signal.

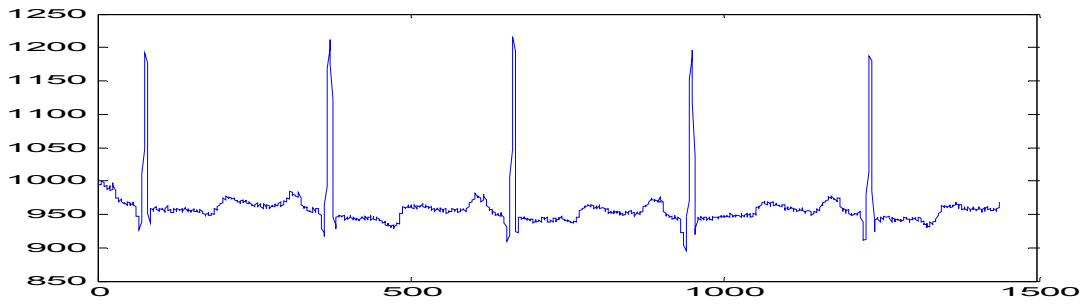
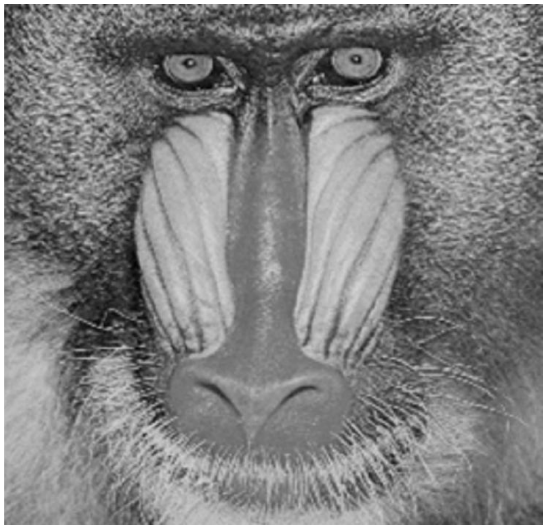


Figure 4.1 : Allure du signal de 100.dat

II.2.3 Présentation des images utilisées

Trois images de test aux niveaux de gris (25x256, 8 bits/pixel) sont utilisées pour montrer l'efficacité des techniques proposées : Lena.bmp, House.bmp qui sont affichées dans la figure (4.2).

(a)

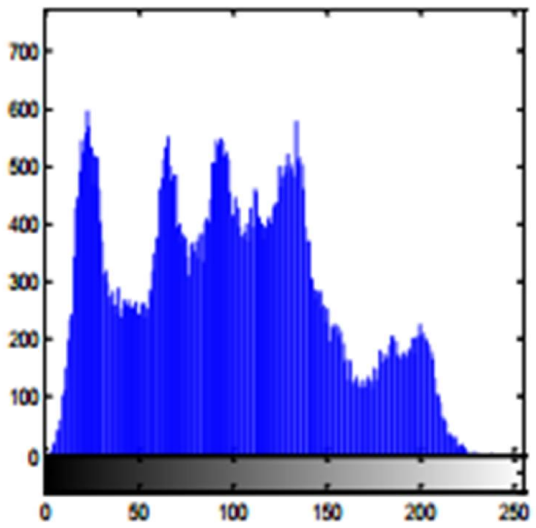


(b)

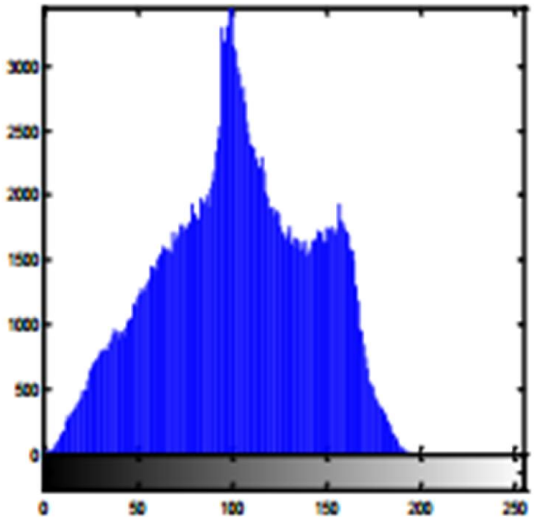
(a) : Lena 256x256 (b) : Baboon256x256

Figure 4.2 : Images de testes utilisées

Les histogrammes correspondants aux images de teste utilisées sont représentés sur la figure (4.3).



(a)



(b)

(a) : Lena (b) : Baboon

Figure 4.3 : Histogrammes correspondants aux images de tests utilisées

IV.3 Résultats expérimentaux

Pour évaluer les performance des techniques de compression proposées, les opérations de codage et de décodage dans cette section ont été effectués. La figure (4.4) décrit la chaine de compression et décompression utilisée.

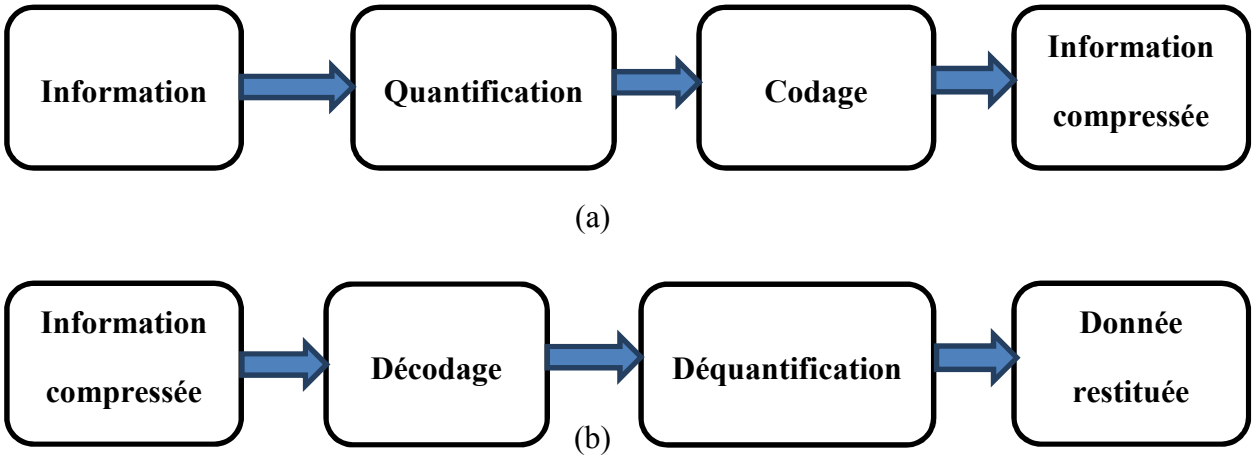


Figure 4.4 : Chaine de compression (a) et décompression (b) utilisées

Comme il a été déjà mentionné, les méthodes développées sont résumés en ce qui suit:

Méthode de Huffman : La méthode de Huffman est une méthode qui va établir des statistiques relatives au fichier à compresser et, à partir de celles-ci, recréer la donnée de façon à réduire la longueur moyenne des bits nécessaires pour encoder l'entièreté de cette donnée.

Codage arithmétique : Le codage arithmétique traite l'ensemble des valeurs quantifiées comme une seule entité. Il fonctionne par la représentation d'un nombre par un intervalle de nombres réels compris entre 0 et 1, les valeurs quantifiées successives réduisent cet intervalle en concordance avec la probabilité d'apparition. Alors nous avons associé à chaque valeur quantifiée un domaine à l'intérieur de l'espace des probabilités compris entre 0 et 1.

Run Length Encoding : Connu aussi sous le nom de Run Length Coding, abrégé par RLE, Run Length Encoding est de loin l'algorithme de compression le plus simple et intuitif. Son principe est de détecter, par exemple dans un texte, les caractères qui se répètent de manière consécutive, et si leur répétition excède un seuil alors on compresse cette répétition c'est à-

dire qu'on remplace cette chaîne du même caractère par un triplé contenant : un caractère de répétition, le nombre de répétitions et le caractère.

LZW : on va lire des données caractères par caractère et émettre au fur et à mesure des codes. Pour que cela compresse, il faut bien sur produire moins de codes que de caractères lus. Un programme de décompression, va lire des codes et produire en sortie des caractères. On doit obtenir à la sortie le même fichier qu'à l'entrée.

Après la phase de compression, on calcule le rapport de compression CR et après la phase de décompression on calcule le PRD ou le PSNR.

IV.3.1 Résultats obtenus de la compression du texte

Le schéma bloc de la figure 4.4 est utiliser mais sans l'étape de quantification et de déquantification, le fait que la compression du texte est sans perte. Le tableau (4.1) expose les résultats obtenus pour les quatre méthodes utilisées.

Codage	Huffman	Arithmétique	RLE	LZW
CR	**1.85**	**1.85**	**0.50**	**0.61**

Tableau 4.1 : Résultats obtenus de la compression de texte

Le taux de compression CR moyen égal à 1.85. pour l'algorithme Huffman contre la même valeur pour le codage arithmétique. Le algorithmes RLE (CR=0.50) et LZW (CR=0.61) sont déconseillés pour les textes.

IV.3.2 Résultats obtenus de la compression du signal ECG

Le tableau (4.2) rapporte les résultats obtenus de la compression du signal 100.dat toujours pour les quatre algorithmes proposés.

D'abord le signal subira une quantification scalaire afin de réduire la résolutions de 11, 10, 9 jusqu'à 8 bits. Pour chaque résolution on applique les quatre algorithmes et on calcule le CR et le PRD.

Résolution (bits)	CR (Huffman)	CR (arithmétique)	CR (RLE)	CR (LZW)	PRD (%)
11	2.62	2.63	1.64	1.63	1.28
10	3.27	3.30	2.47	2.19	2.10
9	4.46	4.49	4.76	4.57	4.04
8	6.35	6.50	6.48	6.54	8.29
Moyenne	**4.18**	**4.23**	**3.84**	**3.73**	**3.93**

Tableau 4.2 : Résultats numérique de notre algorithme de compression.

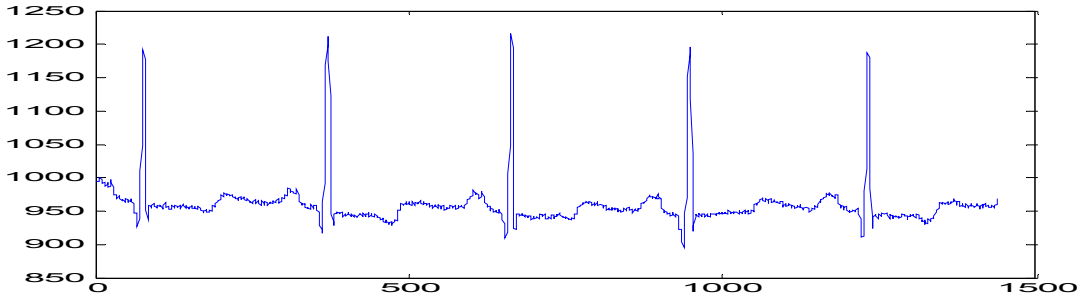
D'après la compression du signal ECG dans sa forme originale, en minimisant la distorsion (valeur moyenne du PRD=3.93%) pour l'enregistrements 100.dat. Le taux de compression CR moyen égal à 4.18 pour l'algorithme Huffman, 3.84 pour l'algorithme RLE, 3.73 pour l'algorithme LZW et 4.23 pour le codage arithmétique, alors l'algorithme le plus efficace et le plus adapté à notre application est codage arithmétique. Les approches RLE et LZW sont plus performants que dans le cas de la compression du texte

Les résultats de signal ECG de l'enregistrements 100 schématisés sur la figures (4.5) et (4.6) représentent le signal original et le signal restitué. Le signal erreur est aussi représenté dans cette figure. reconstruits a quelque quantification correspondants et l'erreur de compression entre les deux pour le cas le plus défavorable (résolutions 10 et 9 bits au lieu de 12 bits). Ces figures montrent que le signal ECG distordu est plus similaire au signal original pour la résolution 10 bits que de la résolution 9 bits.

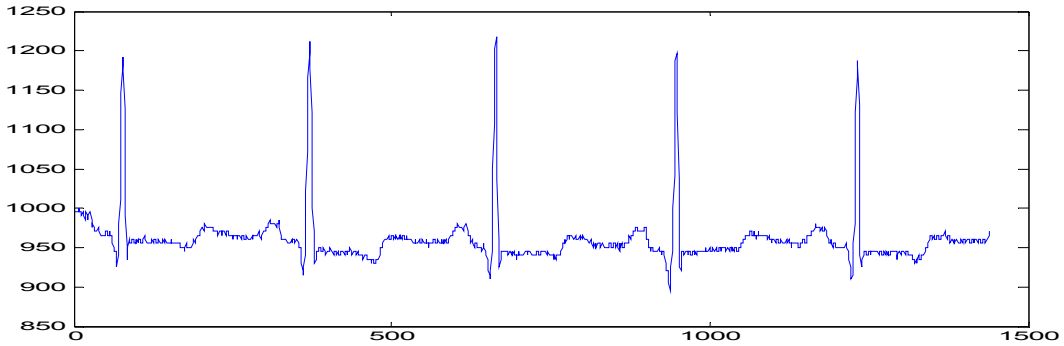
IV.3.3 Résultats obtenus de la compression d'image

La figure (4.7) montre l'allure de la courbe du PSNR (dB) en fonction de la résolution obtenue après quantification scalaire, de l'image Lena.bmp, de taille 256x256. La méthode RLE est moins sensible au pas de quantification que les autres approches.

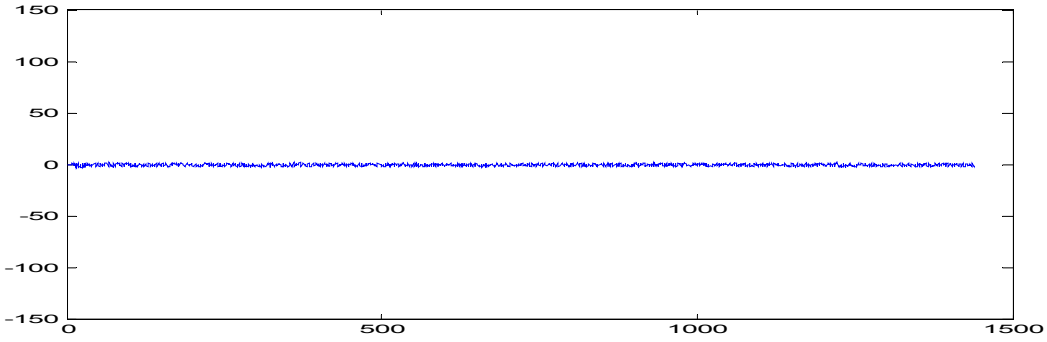
Les résultats de simulation pour les méthodes de compression développées sont présentés dans le tableau (4.3) en terme de PSNR en fonction du taux de compression CR et la résolution pour la même image.



(a)



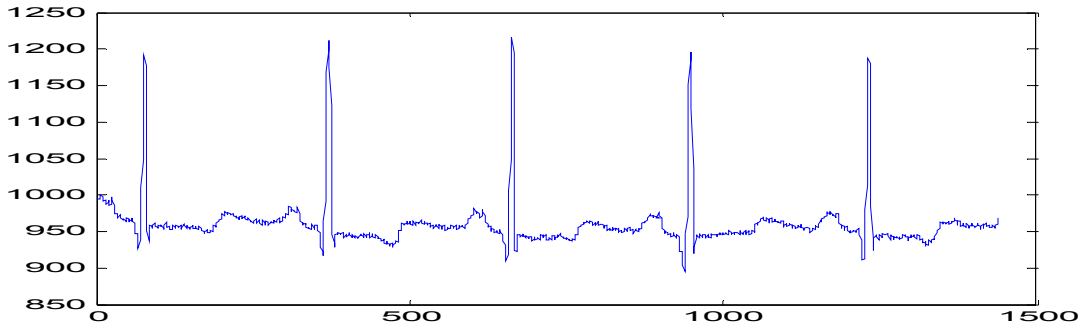
(b)



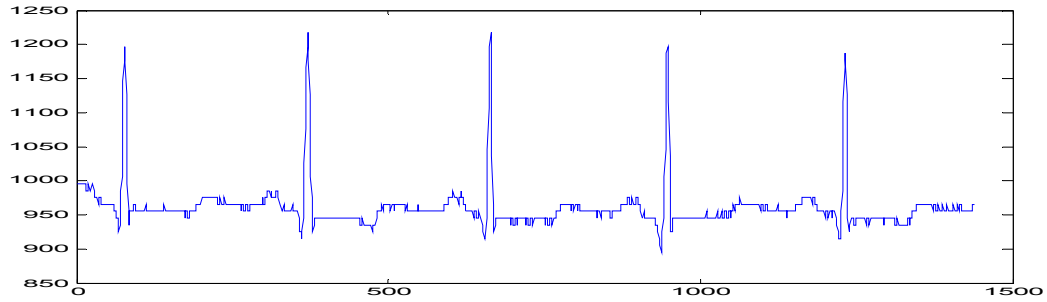
(c)

(a) ECG original, (b) ECG reconstitué, (c) Erreur de compression.

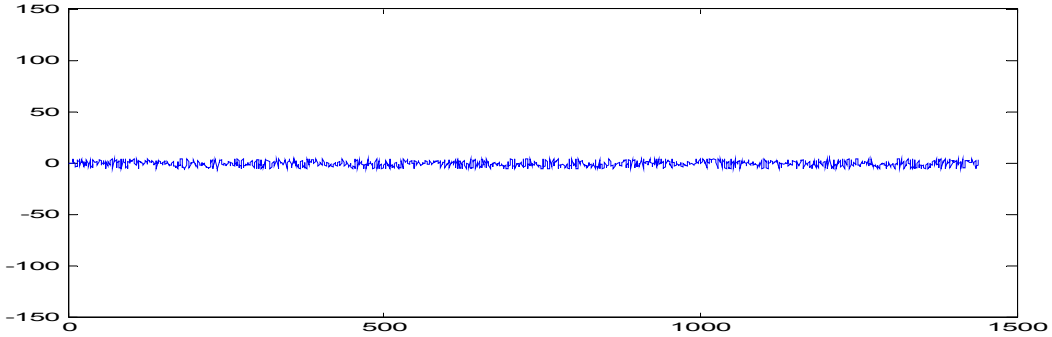
Figure 4.5 : Compression du signal ECG de l'enregistrement 100.dat pour résolution 10 bits.



(a)



(b)



(c)

(a) ECG original, (b) ECG reconstitué, (c) Erreur de compression.

Figure 4.6 : Compression du signal ECG de l'enregistrement 100.dat pour résolution 9 bits.

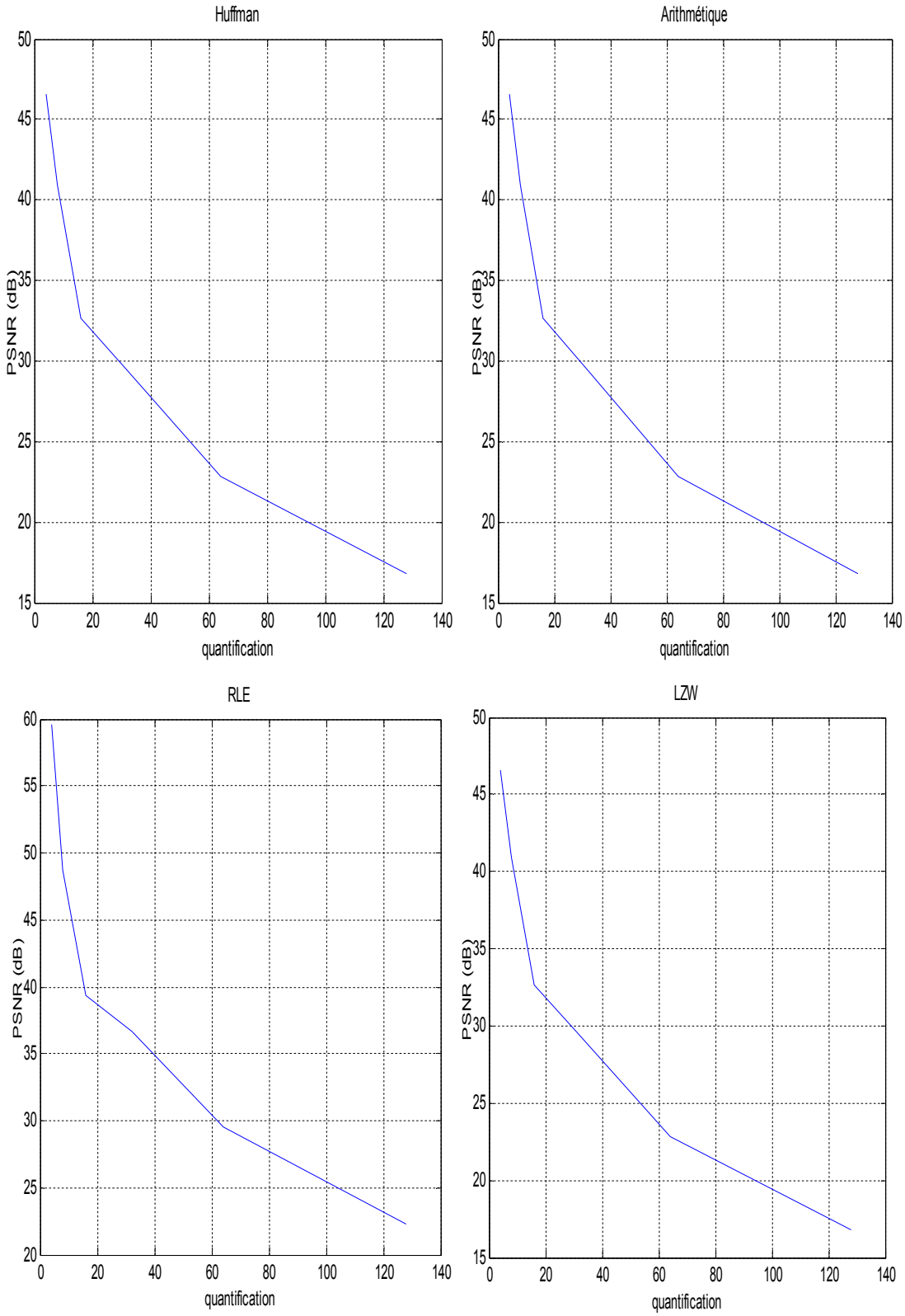


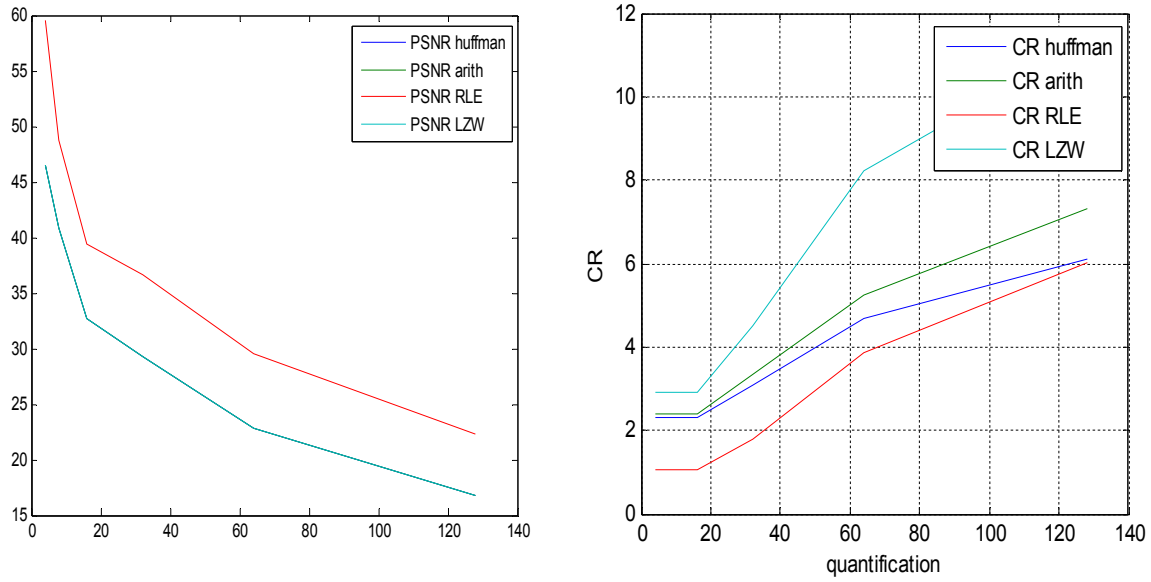
Figure 4.7 : PSNR de l'image Lena en fonction du pas de quantification pour les quatre algorithme

	Résolution (bits)	10	9	8	7	6	5
Huffman	PSNR	46.55	40.90	32.69	29.34	22.86	16.82
	CR	2.30	2.30	2.30	3.08	4.67	6.12
Arithmétique	PSNR	46.55	40.90	32.69	29.34	22.86	16.82
	CR	2.38	2.38	2.38	3.32	5.23	7.31
RLE	PSNR	59.60	48.70	39.37	36.61	29.58	22.30
	CR	1.05	1.05	1.05	1.80	3.87	6.01
LZW	PSNR	46.55	40.90	32.69	29.34	22.86	16.82
	CR	2.90	2.90	2.90	4.52	8.23	11.28

Tableau 4.3 : Comparaison objective en terme de PSNR et CR et la résolution entre les techniques de compression de l'image Lena.

D'après le tableau (4.3), on constate que le PSNR de Huffman, arithmétique et LZW sont identiques, en revanche RLE se manifeste différent de ces valeurs, logiquement , si on prend la valeur typique de PSNR qui varie entre 20 db et 40 db, les valeur du PSNR à la quantification qui correspond à la résolution 5 bits du codage de Huffman, arithmétique et LZW sont inférieurs à 20, en outre on remarque que la valeur de PSNR du codage RLE demeure la meilleure .

De plus le taux de compression CR moyen égal à 3.47 pour l'algorithme Huffman, 2.47 pour l'algorithme RLE, 5.45 pour l'algorithme LZW et 3.84 pour le codage arithmétique, il nous est possible de montrer que l'algorithme le plus efficace et le plus adapté à notre compression d'image est celui du codage LZW (figure 4.8).



(a) (b)

(a) Courbe du PSNR, (b) Courbe du CR

Figure 4.8 : Influence de la quantification sur le PSNR et le CR

La figure (4.9) affiche les images restituées pour quatre valeurs de PSNR. La qualité visuelle est confirmée par cette figure.

Pour confirmer ces résultats, on a répété la simulation sur une autre image "Baboon.bmp". Les résultats sont présentés dans le tableau (4.4).

	Résolution (bits)	10	9	8	7	6	5
Huffman	PSNR	46.37	40.73	34.70	28.80	23.11	17.31
	CR	1.51	1.86	2.42	3.35	4.82	7.23
Arithmétique	PSNR	46.37	40.73	34.70	28.80	23.11	17.31
	CR	1.52	1.88	2.44	3.32	5.42	16.57
RLE	PSNR	60.43	50.48	42.97	36.92	31.45	22.83
	CR	0.57	0.65	0.82	1.21	2.10	7.18
LZW	PSNR	46.37	40.73	34.70	28.80	23.11	17.31
	CR	1.27	1.67	2.30	3.44	5.52	13.11

Tableau 4.4 : Comparaison objective en terme de PSNR et CR et la résolution entre les technique de compression de l'image Baboon.

(a) PSNR= 46.55 db

(b) PSNR=32.69 db

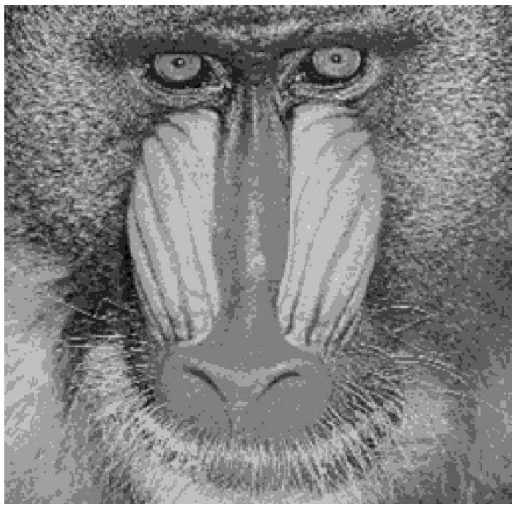
(d) PSNR=29.34 db

(e) PSNR=22.86 db

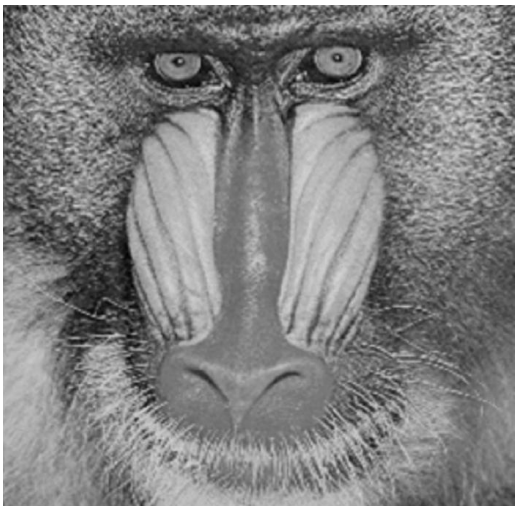
Figure 4.9 : Images Lena reconstruites pour différents PSNR

D'après le tableau (4.4), on a confirmé les résultat du tableau (4.3). Ce qui nous assure de dire que l'algorithme le plus efficace et le plus adapté dans la compression des images parmi les quatre méthodes est l'algorithme de LZW.

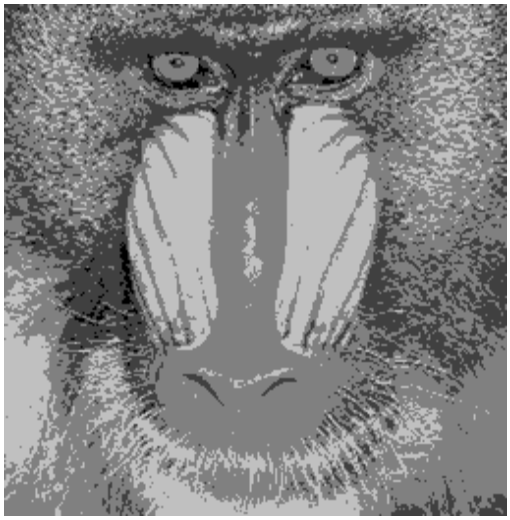
La figure (4.10) montre la qualité visuelle de l'image Baboon. Le PSNR est un critère mathématique qui reflète la qualité visuelle. L'examen du PSNR nous confirme que plus l'image est compressée, plus elle s'éloigne de l'image originale.



(a) PSNR= 40.73 db



(b) PSNR=28.80 db



(c) PSNR=34.70 db

(d) PSNR=17.31 db

Figure 4.10 : Images Baboon reconstruites pour différents PSNR

D'après toute les simulations on a constaté que :

- L'algorithme RLE n'est pas utilisée dans le fichier texte.
- Le codage Huffman et arithmétique sont toujours valables pour tous les formats de fichiers.
- Le temps d'exécution de l'algorithme LZW est très grand, mais l'algorithme est très performant.

Conclusion

Dans ce dernier chapitre, on a évalué et développé les techniques classiques de compression, on s'est servi de trois formes de données (texte, signal et image).

En ce qui concerne le texte qui a été compressé par les techniques proposées dont on a vu ce que le CR augmente tout en postulant que l'algorithme le plus efficace est celui de Huffman et arithmétique.

De même, le signal ECG, dont on a effectué une compression, il résulte un taux de compression élevé et un PRD faible par rapport au signal orignal, il est vraisemblablement que le codage arithmétique parait très efficace.

L'image, quant à elle, a été compressée en l'évaluant par le taux de compression et le PSNR , on a globalement retenu que l'algorithme LZW semble efficace .

Conclusion générale

La nécessité de comprimer des donnés de plus en plus volumineuses et difficiles à transmettre ou à stoker est accrue avec le dévelopment des techniques de communications. Les performances d'un système de compression sont évaluées par sa complexité calculatoire, son cout de stockage, le débit (ou le taux de compression) qu'il permet d'atteindre et la dégradation qu'il engendre sur les données compressées. Les techniques de compression ont fait l'obiet de nombreuses recherches pour tenter d'optimiser ces différents aspects. Il demeure, cependant, difficile d'optimiser tous ces critéres en meme temps.

Au cours de ce travail de recherche, on a vu trois types de compression de données , la compression du texte, du signal et d'image. Pour cela, on a structuré ce travail autour de quatres chapitres fondamentaux. Le début de ce travail a été consacré à quelques notions génerles sur les signaux et l'images. Ensuite, on a présenté les différentes méthodes de codage des données avec et sans perte. On a ensuite introduit des théories et propriétés de la transformée en cosins discrète. Finalement, des simulations et des extractions de résultats ont été réalisées pour permettre l'évaluation de quatre techniques Huffman, codage arithmétique, RLE et LZW par le biais des critères mathématiques: CR, PSNR et PRD.

On a commencé par la compression du texte et le calcul du taux de compression de chaque méthode de compression pour effectuer une comparaison entre elles. Dans la deuxième simulation, on a appliqué la compression au signal ECG et on a mesuré le taux de compression et sa valeur associée de PRD. Finalement, nous avons effectué la meme chose pour la compression de deux types d'images.

Les résultats obtenus ont montré que le codage d'Huffman et le codage arithmétique sont très efficaces et plus performants pour les fichiers textes et signaux unidimensionnels et l'algorithme LZW est très efficace et plus performant surtout pour les images volumineuses.

BIBLIOGRAPHIE

[1] G . Scorletti, “Traitement du Signal”, Ecole d’ingénieur, Ecole Centrale de Lyon, 2013.

[2] É. Tisserand J. Pautex P. Schweitzer, “Analyse et traitement des signaux : méthodes et applications au son et à l’image ”, 2e édition, DUNOD, Paris, 2008.

[3]M. Bellanger, “TRAITEMENT NUMÉRIQUE DU SIGNAL ”, 8e édition, DUNOD, Paris, 2002.

[4] Steven W. Smith, “Digital Signal Processing”, Second Edition, California Technical Publishing, 1999.

[5] F. Cottet, “AIDE-MÉMOIRE TRAITEMENT DU SIGNAL”, Dunod, Paris, 2000.

[6] M. ANDRE, “Introduction aux techniques de traitement d’images”, Eyrolles 1987.

[7] http://www.map.toulouse.archi.fr/works/panoformation/imagenum/imagenum.htm.

[8] http://fr.wikipedia.org/wiki/R%C3%A9solution_spatiale_des_images_matricielles.

[9]M . BENABDELLAH “Outils de compression et de cryptocompression : Applications aux images fixes et video”,UNIVERSITE MOHAMMED V-AGDAL, Juin 2007

[10] R . Gonzales, P. Wintz “Digital Image Processing”, Addison Wessley, 1977.

[11] M. Sharma, “Compression Using Huffman Coding”, International Journal of Computer Science and Network Security, VOL.10 No.5, May 2010.

[12] M. Benabdellah, “Outils de compression et de crypto compression : Applications aux images fixes et vidéo”, Thèse Doctorat en Télécommunications et Multimédia, Université de Mohammed Agdal, 2007.

[13] A. Isar, C . Andrei, N. Miranda, “Algorithmes et techniques de compression ,” Editura ORIZONTURI POLITEHNICE, 2002.

[14] F. Davoine, “Compression d’images par fractales basée sur la triangulation de Delaunay,” Institut National Polytechnique de Grenoble - INPG, Françe 1995.

[15] V. Beaudoin, “Développement de nouvelles techniques de compression de données sans perte”, Thèse Maître ès sciences en Sciences et génie, Université Laval, 2008.

[16] M. Sharma, “Compression Using Huffman Coding”, IJCSNS International Journal of Computer Science and Network Security, VOL.10 No.5, May 2010.

[17] S. Sarika, S. Srilali, “Improved Run Length Encoding Scheme For Efficient Compression Data Rate,” Nov-Dec 2013.

[18] F. LUBIN, "La compression audio-numerique," Enseignement des Métiers de la Communication.

[19] Jonathan Taquet, "Techniques avancées pour la compression d'images médicales," Réseaux et télécommunications [cs.NI]. Université Rennes 1, 2011. Français.

[20] S. Pigeon, "Contributions à la compression de données," Thèse présentée à la Faculté des arts et sciences en vue de l'obtention du grade Philosophiæ Doctor (Ph. D.) en Informatique, Décembre 2001.

[21] Z. E. Baarir, A. Ouafi, "Etude de la transformée en ondelettes dans la compression d'images fixes," Courrier du Savoir, N°05, pp. 69-74, 2004.

[22] A. Al-Haj, "Combined DWT-DCT Digital Image Watermarking," Journal of Computer Science 3, Amman, Jordan, 2007.

[23] W. B. Pennebaker, J. L. Mitchell, "JPEG – Still Image Data Compression Standard," Newyork: International Thomsan Publishing, 1993.

[24] S. A Khayam, "The discrete cosine transform (DCT) : theory and application," Departement of Electrical and computer Engineering, Michigan State University, March, 2003.

[25] P. Beaurepaire, "compression d'image appliquée aux angiographies cardiaques : aspects algorithmiques, évaluation de la qualité diagnostique", thèse doctorat, d'ordre 07 ISAL0107, Lyon, France, 1997.

[26] D. Saloman, "Data compression : the complet reference," London, Fourth Edition, Springer-Verlag London Limited, 2007.

[27] K. Arbatni, "Réseaux de neurones appliqués à l'analyse et à la modélisation non linéaire du signal ECG," Mémoire de Magister en Traitement du Signal, Université de Constantine, Algérie, 2007.

[28] A. Hodni, "Analyse du signal ECG par réseaux de neurones récurrents", Mémoire de Magister en Traitement du Signal, Université de Constantine, Algérie, 2009.

[29] MIT-BIH Arrhythmia Database: www.physionet.org.

MEMOIRE DE FIN D'EUDES POUR L'OBTENTION DU DIPLOME DE MASTER EN ELECTRONIQUE

Option: Contrôle Industriel

Proposé et dirigé par : Dr. N. BOUKHENNOUFA

Etudié par : AHMED OUSSAMA

Thème : Développement et évaluation des techniques classiques de compression

Résumé

La compression de données, de façon simplifiée, c'est l'ensemble des méthodes que l'on utilise pour prendre un message long pour en faire un message court, sans perdre d'information importante. Nous trouvons de tels usages dans la vie de tous les jours et pas seulement dans les applications technologiques.

Ce travail s'inscrit dans la perspective de l'évaluation de techniques classiques de la compression de données (Huffman, Arithmétique, RLE et LZW) par les critères de performance qui sont: taux de compression, le rapport signal à bruit en pic (PSNR) pour les images et Taux de distorsion (PRD) pour les signaux.

Ainsi, une comparaison entre ces techniques permet de juger les méthodes les plus performantes ainsi que leurs domaines d'application.

Mots clés : Compression, PSNR, PRD, Taux de compression

Abstract

Data compression, in simplified way , it's all the methods that are used to take a long message into a short message, without losing important information. We find such uses in the life of every day and not just in the technology application.

This work is from the perspective of evaluating conventional data compression techniques (Huffman , Arithmetic, RLE and LZW) by the performance criteria which are: the compression ratio, the signal to noise ratio peak (PSNR) for images and distortion (PRD) for signals.

Thus, a comparison of these techniques to try the most effective methods and their application areas.

Keywords : Compression, PSNR, PRD, Compression ratio