GRÉGORY HUCHET

Nouvelles méthodes de codage vidéo distribué

Thèse présentée
à la Faculté des études supérieures de l'Université Laval
dans le cadre du programme de doctorat en Génie Électrique
pour l'obtention du grade de *Philosophiæ Doctor* (Ph.D.)

Faculté des Sciences et de Génie
UNIVERSITÉ LAVAL
QUÉBEC

2009

©Grégory Huchet, 2009

Résumé

De nos jours, certaines applications vidéo, de par des limitations de mémoire et de capacités de calculs, requièrent un système où l'encodage d'une séquence vidéo doit être le plus simple possible. Le codage vidéo distribué, nouveau paradigme de la compression vidéo, semble prometteur pour ce genre de demande. Il permet une compression peu complexe de séquences animées. Son principe repose sur les travaux de D. Slepian et J. K. Wolf [1] ainsi que ceux de A. D. Wyner et J. Ziv [2] menés dans les années 70. La présente thèse énonce trois différentes contributions relatives au codage vidéo conventionnel et distribué. La première est un nouveau procédé de filtrage vidéo basé sur l'utilisation d'arbres de décision. Afin de réduire les artefacts visuels issus d'une trop forte compression, les arbres de décision identifient et appliquent les meilleurs coefficients de filtrage en fonction des données à filtrer. La seconde contribution concerne un schéma de codage vidéo distribué où l'on évite de transmettre au décodeur l'information relative aux blocs de l'image les moins significatifs. De cette manière, on engendre une diminution importante du débit binaire et une réduction de la complexité de décodage. Ce schéma de codage repose sur une séparation en couche des coefficients de transformée en cosinus discret de l'image. Les coefficients DC sont les premiers à être transmis au décodeur. De là, ils sont analysés par ce dernier afin de trouver les blocs les plus significatifs de l'image. Le décodeur indique alors à l'encodeur, via un canal de retour, quels sont les coefficients AC des blocs à transmettre. Enfin, la dernière contribution consiste en une méthode de représentation binaire adaptative des images dans les procédés de codage vidéo distribué. Cette représentation permet de réduire efficacement le débit binaire et tenant compte uniquement de l'information source la plus pertinente. Ici encore cette méthode repose sur une utilisation plus efficace d'un canal de retour.

Abstract

Nowadays, because of computational complexity and limited memory, some video applications are required to have a very simple video coding process. Distributed video coding, a new paradigm of video compression, seems like a very good candidate for this. It allows for low computational complexity encoding of video sequences. The principle is based on the research of D. Slepian and J. K. Wolf [1] and also of A. D. Wyner and J. Ziv [2], carried out in the seventies. This thesis presents three different contributions in the conventional and distributed video coding fields. The first one is a new image filter using decision trees, a machine learning algorithm. To reduce the visual impact of artifacts resulting from a high compression, the decision trees identify and apply the best filtering coefficients according to the input data. The second contribution is a new distributed video coding scheme where the transmission of the quantized AC coefficients of non-significant blocks in the source image is avoided. This leads to a reduction of the bitrate as well as a decrease in the decoding complexity. The coding scheme is based on the transform coefficients being separated into layers. The entire image's DC coefficients are the first transmitted to the decoder. Then, they are analyzed by the decoder to find the image's most significant blocs. Afterwards, the decoder informs the encoder, via a return channel, which quantized AC coefficients have to be transmitted. Finally, the last contribution consists of an adaptive binary representation of images in distributed video coding systems. This representation allows for an efficient reduction of the bitrate by taking into account the most significant information provided by the information source. Once again, this method requires a return channel.

Remerciements

Au cours de ce doctorat, j'ai été amené à effectuer plusieurs séjours au Centre de Recherche sur les Communications (CRC) à Ottawa. La collaboration, faite avec le groupe de recherche sur les systèmes vidéo de pointe, m'a permis d'avoir une approche plus directe et plus intéressante sur les techniques de compression vidéo.

Pour cela, je tiens tout d'abord à remercier mon directeur de recherche, le professeur Jean-Yves Chouinard, sans qui cette collaboration n'aurait pu aboutir. Je tiens aussi à le remercier pour sa confiance, son aide, ses conseils, sa disponibilité et sa sympathie dont il a toujours fait preuve à mon égard.

Je remercie également André Vincent, gestionnaire du groupe de recherche, pour avoir également permis cette collaboration et pour son accueil au sein du groupe. Ses conseils, sa personnalité très ouverte et sa présence toujours amicale était pour moi très importante.

Je tiens ensuite à exprimer ma reconnaissance envers Demin Wang, qui est en fait la personne originaire de ce projet. L'écoute, l'aide et la patience dont il m'a souvent fait part m'ont permis d'acquérir une réflexion et une méthode de travail plus juste et plus pertinente.

Enfin je remercie tous les membres du groupe et toutes les personnes que j'ai pu côtoyer au CRC avec qui j'ai passé d'agréables moments.

Table des matières

Résumé ii

Abstract iii

Remerciements iv

Table des matières v

Table des figures viii

Liste des tableaux xii

1 Introduction 1

2 Élements de théorie de l'information 3
- 2.1 Notions préliminaires . . . 3
  - 2.1.1 Quantité d'information . . . 3
  - 2.1.2 L'entropie . . . 4
  - 2.1.3 Entropie conjointe et conditionnelle . . . 5
  - 2.1.4 Entropie relative et information mutuelle . . . 6
  - 2.1.5 Relations générales . . . 7
- 2.2 Fonction de débit-distorsion . . . 8
  - 2.2.1 Définitions . . . 8
  - 2.2.2 Fonction de distorsion . . . 10
  - 2.2.3 Théorème de débit-distorsion . . . 11
- 2.3 Le codage de source distribuées . . . 12
  - 2.3.1 Théorème de Slepian-Wolf . . . 12
  - 2.3.2 Théorème de Wyner-Ziv . . . 13
  - 2.3.3 Fonction de distorsion et information latérale . . . 14
- 2.4 Conclusion . . . 15

3 Le codage vidéo 16
- 3.1 La vidéo numérique . . . 16
- 3.2 La compression vidéo . . . 18

3.2.1 Estimation et compensation de mouvement 19
3.2.2 Limitations 21
3.2.3 La transformation DCT 22
3.2.4 La quantification 24
3.2.5 La norme H.264/AVC 29
3.3 Le codage vidéo distribué 33
3.3.1 Principes élémentaires 33
3.3.2 Étude d'un système de référence 35
3.3.3 Revue de littérature 39

4 Filtrage vidéo par arbres de décision **42**
4.1 Avant-propos 42
4.1.1 Rappels sur la norme H.264/AVC 42
4.1.2 Les méthodes d'apprentissage 43
4.2 Principe du filtre proposé 44
4.3 Base de données d'entraînement 46
4.4 Sélection de l'algorithme de classification 47
4.5 Les arbres de décision 48
4.5.1 Principe 48
4.5.2 L'algorithme *M5'* 50
4.6 Exemple de filtrage 53
4.7 Résultats expérimentaux 55
4.8 Conclusion 59

5 Schéma de compression vidéo distribuée guidée par le coefficient DC **61**
5.1 Étude statistique des différences entre les coefficients de la DCT 61
5.2 Architecture du codec proposé 65
5.2.1 Encodeur 68
5.2.2 Décodeur 70
5.3 Conditions expérimentales et résultats 75
5.3.1 Paramétrage de la valeur D_C 75
5.3.2 Génération de l'information latérale 76
5.3.3 Résultats expérimentaux 76
5.4 Conclusion 80

6 Représentation binaire adaptative **81**
6.1 Rappels sur le codage vidéo distribué 81
6.2 Principe de la méthode 82
6.2.1 Énoncé du problème 82
6.2.2 Solution proposée 83
6.2.3 Représentation *sub-bit* 87

6.3 Étude d'exemples 89
6.4 Conditions expérimentales et résultats 94
6.5 Conclusion 97

7 Conclusion générale **98**
7.1 Introduction 98
7.2 Contributions 98
7.3 Suggestion de travaux futurs 99

A Codage de source et de canal **101**
A.1 Codage de source 101
A.1.1 Classes de codes 101
A.1.2 Séquences typiques et atypiques 102
A.1.3 Théorème de Shannon-McMillan 103
A.1.4 Théorème du codage de source 103
A.2 Codage de canal 105
A.2.1 Capacité d'un canal discret sans mémoire 106
A.2.2 Capacité d'un canal discret avec mémoire 106
A.2.3 Séquences conjointement typiques 106
A.2.4 Théorème du codage de canal 107

B Codes à faible densité de parité (LDPC) **109**
B.1 Présentation 109
B.2 Généralités 110
B.3 Encodage 111
B.4 Décodage 111
B.5 Application au codage distribué 114
B.5.1 Système 115
B.5.2 Codes LDPCA 116

C Codes Gray et codage vidéo distribué **120**

D CRC-FRC **122**

E Reconstruction optimale des coefficients de source quantifiés **123**
E.1 Reconstruction standard 123
E.2 Reconstruction optimale 124

Bibliographie **126**

Table des figures

2.1 Entropie d'une variable aléatoire binaire. 5
2.2 Relation entre l'entropie et l'information mutuelle. 7
2.3 Diagramme de Venn à 3 variables aléatoires. 8
2.4 Chaîne de codage et décodage. 9
2.5 Mesures et probabilités de distorsion. 10
2.6 Fonction de distorsion d'une variable aléatoire binaire. 12
2.7 Illustration du théorème de Slepian-Wolf pour deux sources. 13
2.8 Codage de source avec information latérale. 14

3.1 Calcul des vecteurs de mouvement. 20
3.2 Exemples de quantification uniforme avec $\Delta = 1$: *a)* quantificateur de type *midtread* ; *b)* quantificateur de type *midrise*. 26
3.3 Erreur de quantification pour un quantificateur uniforme *midtread* avec $\Delta = 1$. 26
3.4 Quantificateur uniforme à zone morte avec $\Delta = 1$ et $f = 1/4$. 27
3.5 Profil typique d'application du filtre [3]. 31
3.6 H.264 : schéma de principe de l'encodeur [4]. 32
3.7 H.264 : schéma de principe du décodeur [4]. 32
3.8 Principe du codeur/décodeur de Wyner-Ziv. 33
3.9 Canal binaire symétrique avec ses probabilités de transition $\Pr(x|y)$. . 34
3.10 Système de codage vidéo distribué basé sur la transformation DCT. . . 36
3.11 Matrices de quantifications. 38

4.1 Sélection des attributs horizontaux en vue de la création des exemples d'entraînements. 44
4.2 Diagramme en bloc du filtre par arbres de décision. 45
4.3 Système de codage vidéo incluant le filtre par arbres de décision : (a) encodeur, (b) décodeur. 45
4.4 Création des instances de la base de données. 46
4.5 Structure typique d'un arbre de décision. 50
4.6 Exemple d'un pixel devant être filtré. 53

4.7 Extrait de l'arbre de décision associé à p_1 (*Foreman* - QCIF - 30 fps - $QPI = 35$). LM signifie "Linear Model". Les lignes en pointillés ici permettent uniquement de s'affranchir de certaines étapes dans l'arbre afin de simplifier la structure. 54
4.8 Résultat du filtrage de p_1. 55
4.9 Évolution du PSNR sur les 60 premières images de *Foreman* avec $QP = 35$. 57
4.10 Évolution du gain sur les 60 premières images de *Foreman* avec $QP = 35$. Le gain le plus élevé est donné pour l'image d'entraînement. 57
4.11 Image n°20 de la séquence vidéo *Foreman* avec $QP = 35$. *Gauche* : avant filtrage par arbres de décision (31.588 dB). *Droite* : après filtrage (32.231 dB). 58
4.12 Portion de l'image n°20 de la séquence vidéo *Foreman* avec $QP = 35$. *Gauche* : avant filtrage par arbres de décision (31.588 dB). *Droite* : après filtrage (32.231 dB). 58
4.13 Comparaison entre *M5'* et l'algorithme de régression linéaire pour la séquence *Foreman*. 59
4.14 Courbe de débit-distorsion obtenue avec le filtre par arbres de décision sur les 60 premières images de la séquence *Foreman* (QCIF - 30 fps). . 60

5.1 Balayage en zigzag des coefficients issus de la DCT. 62
5.2 Résultats obtenus pour la différence du coefficient AC AD$(0,1)$ avec $QPW = 0$. *Haut* : Nombre de blocs correspondant à chaque valeur de AD$(0,0)$; *Centre* : Moyenne de AD$(0,1)$ en fonction de AD$(0,0)$; *Bas* : Ecart-type de AD$(0,1)$ en fonction de AD$(0,0)$. 63
5.3 Répartition des différences des coefficients AC AD(0,1) en fonction des différences des coefficients DC AD(0,0) pour la première image WZ de la séquence *Foreman*. 64
5.4 Diagramme en bloc du codec proposé. 66
5.5 Diagramme en bloc de l'encodage et du décodage de Slepian-Wolf à double couche. 67
5.6 Valeurs de r selon la position du coefficient dans le bloc. 69
5.7 Exemple de concaténation des plans de bits : chaque plan de bits possède une longueur initiale G inférieure à L. Après la concaténation, si les bits restants ont une longueur D plus grande que D_C, ils seront concaténés avec des bits de valeur nulle afin de produire un nouveau message de longueur L. Dans le cas contraire, ils seront transmis directement au décodeur. 71

5.8 Courbes de débit-distorsion pour la séquence vidéo *Foreman* (QCIF, $QPI = 0$, GOP : IW, images impaires WZ à 15 fps). *Haut :* information latérale générée avec le CRC-FRC. *Bas :* information latérale générée par moyennage de deux images K consécutives. 77
5.9 Courbes de débit-distorsion pour la séquence vidéo *Carphone* (QCIF, $QPI = 0$, GOP : IW, images impaires WZ à 15 fps). *Haut :* information latérale générée avec le CRC-FRC. *Bas :* information latérale générée par moyennage de deux images K consécutives. 78
5.10 Reconstruction de l'image 4 de la séquence *Foreman* (QCIF - 30fps) : $QPI = 0$, $QPW = 30$, $P = 10$. *Gauche* : information latérale générée avec le CRC-FRC (36.727 dB). *Droite* : image décodée (38.296 dB) : 0.37 bpp. 79
5.11 Reconstruction de l'image 4 de la séquence *Foreman* (QCIF - 30fps) : $QPI = 0$, $QPW = 30$, $P = 10$. *Gauche* : information latérale générée par simple moyennage (32.007 dB). *Droite* : image décodée (38.206 dB) : 0.43 bpp. 79

6.1 Schéma bloc proposé de l'encodeur et décodeur de Slepian-Wolf utilisant la représentation adaptative. 84
6.2 Distribution Laplacienne des différences entre X et Y. E_{max} représente la valeur maximale absolue de l'erreur. 85
6.3 Distribution des valeurs $X^Q_{u,v}$, des différences $X^Q_{u,v} - Y^Q_{u,v}$ et des valeurs $X^{u,v}_{tr}$ pour la bande DC $(0,0)$: *Foreman* (QCIF - 30 fps), image 2, $QPI = 0$, $QPW = 20$. L'information latérale est générée via l'algorithme CRC-FRC (36.86 dB). 91
6.4 Distribution des valeurs $X^Q_{u,v}$, des différences $X^Q_{u,v} - Y^Q_{u,v}$ et des valeurs $X^{u,v}_{tr}$ pour la bande AC $(2,0)$: *Foreman* (QCIF - 30 fps), image 2, $QPI = 0$, $QPW = 20$. L'information latérale est générée via l'algorithme CRC-FRC (36.86 dB). 92
6.5 Courbes de débit-distorsion pour le système de compression guidé par le coefficient DC avec la séquence *Foreman*. 95
6.6 Courbes de débit-distorsion pour le système de compression guidé par le coefficient DC avec la séquence *Carphone*. 95
6.7 Courbes de débit-distorsion du système de Brites et al. [5] avec la séquence *Foreman*. 96
6.8 Courbes de débit-distorsion du système de Brites et al. [5] avec la séquence *Carphone*. 96

A.1 Les différentes classes de codes. 102
A.2 Système de communication. 106

B.1 Graphe de Tanner d'un code LDPC. 110
B.2 Principe de fonctionnement d'un code LDPCA [6]. 117
B.3 Syndrome réduit : avec et sans accumulateur [6]. 118
B.4 Performances des codes LDPCA de longueur $L = 396$. Débit d'encodage normalisé en fonction de l'entropie $H(X|Y) = -\epsilon \log_2(\epsilon) - (1-\epsilon) \log_2(1-\epsilon)$ puis en fonction de la probabilité d'entrecroisement ϵ du canal binaire symétrique. 119

Liste des tableaux

3.1 Paramètres d'applications vidéo typiques [7]. 18

4.1 Performances des différents algorithmes de classification avec les données de l'image n°20 de la séquence Foreman (QCIF - 30 fps). 49

4.2 Gains moyens du PSNR obtenus sur les 60 premières images des séquences vidéo utilisées. 56

5.1 Coefficient de corrélation entre la différence absolue AD$(0,0)$ (DC) et les autres différences absolues AD(u,v) (AC). 64

5.2 Nombre de coefficients transmis par bloc et par valeur du paramètre QPW. 76

6.1 Exemple de conversion de la valeur $X = 63$ avec $E_{max} = 3$. D'une manière équivalente, différentes valeurs autour de X sont exposées. Elles correspondent à différentes valeurs possibles de Y où aucune ambiguïté n'existe pour la reconstruction. 86

6.2 Représentation des valeurs de la source dans la base $B = 12$. Il est aisé de voir que les 4 premiers mots de codes Gray ont un radical de 3 bits différent des autres ($S = 3$). 88

6.3 Tableau comparatif pour le système de compression guidé par le coefficient DC : $QPI = 0$, $QPW = 20$ et $P = 15$. La longueur moyenne des mot-codes $\bar{\ell}$ (en bits) après représentation adaptative est indiquée pour chaque bande de fréquence transmise. Les données sont fournies d'après l'image 2 de la séquence *Foreman* (QCIF - 30 fps) et l'information latérale est générée via l'algorithme CRC-FRC (36.86 dB). 90

6.4 Tableau comparatif pour le système du système de Brites et al. [5] avec la matrice de quantification M5 (cf. figure 3.11). La longueur moyenne des mot-codes $\bar{\ell}$ (en bits) après représentation adaptative est indiquée pour chaque bande de fréquence transmise. Les données sont fournies d'après l'image 2 de la séquence *Carphone* (QCIF - 30 fps) et l'information latérale est générée par l'algorithme CRC-FRC (31.99 dB). 93

Chapitre 1

Introduction

Avec l'avènement de nouveaux services de téléphonie mobile, le codage vidéo distribué semble être le nouveau paradigme de la compression vidéo. Les standards actuels de compression vidéo tels que ISO MPEG ou H.263 et H.264 de l'ITU-T présentent une asymétrie de complexité [8]. En effet, la complexité calculatoire est beaucoup plus importante au niveau de l'encodage que pour le décodage. Cette asymétrie reste profitable dans certaines applications comme la diffusion vidéo où le décodage s'effectue au niveau de chaque unité de réception tandis que le processus d'encodage ne s'opère qu'une seule fois. Certaines applications requièrent néanmoins un système où cette fois la complexité de l'encodage se veut très faible. C'est le cas par exemple des caméras intégrées aux téléphones portables où la mémoire et les calculs doivent être peu élevés. Pour ce type d'application, le codage vidéo distribué semble être particulièrement viable.

Le codage vidéo distribué permet la compression d'images issues d'une séquence animée et comportant de fortes corrélations. Son principe consiste à exploiter au mieux les redondances temporelles d'information au niveau du décodage. En effet, pour deux images successives, la deuxième peut s'apparenter à une version bruitée de la première. La différence majeure avec un codeur vidéo standard est que la compression s'effectue indépendamment pour les deux images sous la condition d'une opération de décodage conjointe.

Ce procédé de codage possède deux fondements : (i) le théorème de Slepian-Wolf [1] établi en 1973 ainsi que (ii) le théorème de Wyner-Ziv établi en 1976 [2]. Le théorème de Slepian-Wolf stipule qu'il est possible et ce, pour plusieurs sources d'information corrélées, d'encoder indépendamment ces dernières et de les transmettre à des débits de sorte qu'il n'y ait aucune perte d'information à la réception. L'opération de décodage doit cependant s'effectuer de manière conjointe. Wyner et Ziv ont plus tard étendu ces travaux pour établir les limites théoriques d'une compression avec perte [2].

Bien que ces principes ont été énoncés depuis plus de trente ans, ce n'est que très récemment que des dispositifs permettent d'atteindre au plus près les débits théoriques du codage distribué. Ceux-ci reposent pour la plupart sur des techniques de codage de canal dont les plus performants utilisent les turbo-codes et les codes à faible densité

de parité (LDPC : *Low-Density Parity-Check Codes*). Les systèmes de codage vidéo distribué possèdent donc intrinsèquement un codage conjoint source-canal.

Nous débuterons cette thèse par un chapitre énonçant les concepts de base de la théorie de l'information. Ces concepts sont importants pour bien décrire les principes fondamentaux du codage vidéo distribué. Le chapitre 3 présentera quant à lui, des notions générales sur la compression vidéo conventionnelle ainsi que sur le codage vidéo distribué. Les chapitres qui suivront apportent les contributions originales de cette thèse. Ces contributions, au nombre de trois, se veulent simples dans leurs fondements. La première, au chapitre 4, consiste en un filtre permettant de réduire les effets de bloc d'une image. Ce filtre utilise pour cela des arbres de décision en vue d'appliquer le meilleur jeu de filtrage dépendamment de la classe des données à filtrer. La seconde contribution, au chapitre 5, concerne un nouveau procédé de codage vidéo distribué. Celui-ci permet de ne transmettre au décodeur que l'information relative aux blocs de l'image les plus significatifs, engendrant ainsi une réduction importante du débit binaire. Enfin, la dernière contribution, présentée au chapitre 6, est une nouvelle méthode de représentation binaire des images dans les procédés de codage vidéo distribué. Cette représentation permet de réduire très efficacement la quantité de données à transmettre au décodeur. Elle fonctionne d'après l'information relative à la différence entre la source et l'information latérale et nécessite pour cela l'utilisation d'un canal de retour.

Chapitre 2

Élements de théorie de l'information

La théorie de l'information représente le fondement du codage distribué. Cette partie introduit des concepts liés à cette théorie et n'y présente que les éléments nécessaires pour la compréhension du reste du document. Bien que les premières tentatives de mesure de l'information datent des années vingt, ce n'est qu'à partir de 1948, grâce aux travaux effectués par Claude Shannon que la théorie de l'information a pris sa forme actuelle. C'est une discipline fondamentale du domaine des communications qui consiste à déterminer les limites (imposées par les lois physiques) de la compression et de la transmission d'une source d'information. Elle est par ailleurs en constante évolution en regard des exigences contemporaines où l'information à stocker et à transmettre devient de plus en plus importante.

2.1 Notions préliminaires

2.1.1 Quantité d'information

Nous allons introduire le concept de mesure d'information en supposant une source d'information discrète et d'alphabet fini. Le point de vue de la théorie de l'information conduit à nous intéresser à des sources et des canaux aléatoires. De ce fait, le destinataire, qui ne dispose que du message reçu, peut seulement faire des hypothèses quant au message émis et choisir le plus vraisemblable. Il traite donc le message délivré par la source comme un processus aléatoire. Puisque son caractère imprévisible apparaît comme un attribut essentiel de l'information, on est conduit à identifier la mesure quantitative de l'information à la mesure de l'imprévisibilité. On mesurera ainsi la quantité d'information $I(x)$ apportée par la réalisation d'un événement x de probabilité $p(x)$ par une certaine fonction croissante de son improbabilité soit :

$$I(x) = f\left[\frac{1}{p(x)}\right], \tag{2.1}$$

où f est une fonction croissante et continue.

Il vient assez logiquement qu'un événement certain apportera une quantité d'information nulle, soit $f(1) = 0$. De même, la réalisation de deux événements indépendants x et y apportera la somme de leurs quantités d'information individuelles, soit :

$$\begin{aligned} I(x,y) &= f\left[1/p(x,y)\right] = f\left[1/p(x)p(y)\right] \\ &= f\left[1/p(x)\right] + f\left[1/p(y)\right] \\ &= I(x) + I(y). \end{aligned} \tag{2.2}$$

Puisque pour des événements indépendants, $p(x,y) = p(x) \cdot p(y)$, il peut être prouvé que seule la fonction logarithme remplit les conditions décrites ci-dessus. La réalisation d'un événement x est donc associée à la quantité d'information :

$$I(x) = \log_b \frac{1}{p(x)} = -\log_b p(x). \tag{2.3}$$

Remarque : Le choix de la base b du logarithme ne définit que l'unité dans laquelle l'information est mesurée :

- $b = 2$: *shannons (Sh) ou bit (binary units)*,
- $b = e$: *logons ou nats (natural units)*,
- $b = 10$: *hartleys*.

2.1.2 L'entropie

L'entropie représente la moyenne pondérée de la quantité d'information de chaque symbole délivré par la source. En d'autres termes, c'est l'espérance mathématique de l'information élémentaire donnée par chaque réalisation possible $X = x_i$ de la source, c'est-à-dire [9] :

$$H(X) = E\left[I_X(x_i)\right] = -\sum_{i=1}^{N} p(x_i) \log_b p(x_i). \tag{2.4}$$

On considère dans cette définition que $0 \times \log_b 0 = 0$.

Remarque : Exprimée en shannons, $H(X)$[1] représente le nombre moyen de bits nécessaires à la codification binaire des différentes réalisations de X.

L'entropie maximale réalisable est atteinte lorsque les événements sont équiprobables. Ainsi pour N événements possibles de l'espace d'épreuves $\mathcal{X}$ nous avons :

$$H(X) \leq \log_b N. \tag{2.5}$$

1. $H_b(X) = (\log_b a) H_a(X)$ puisque $\log_a(x) = \log_b(x)/\log_b(a)$.

Exemple : Pour une variable aléatoire binaire valant 1 avec la probabilité p et 0 avec la probabilité $(1-p)$ l'entropie correspondante (cf. Figure 2.1) exprimée en unité binaire vaut [10] :

$$H(X) = -p\log_b p - (1-p)\log_b(1-p) \stackrel{def}{=} H(p). \tag{2.6}$$

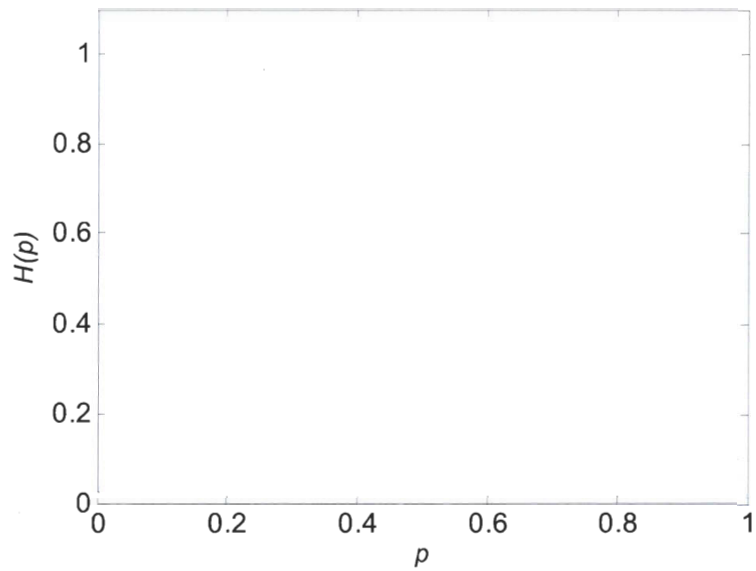


FIGURE 2.1 – Entropie d'une variable aléatoire binaire.

2.1.3 Entropie conjointe et conditionnelle

Considérons maintenant deux variables aléatoires X et Y à valeurs tirées de deux espaces d'épreuves $(x_1, x_2, \ldots, x_N)$ et $(y_1, y_2, \ldots, y_M)$. La définition précédente appliquée aux éléments (x_i, x_j) du produit cartésien de ces espaces, conduit à l'entropie conjointe [9] :

$$\begin{aligned} H(X,Y) &= -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x,y)\log_b p(x,y) && (2.7)\\ &= -\sum_{i=1}^{N}\sum_{j=1}^{M} p(x_i,y_i)\log_b p(x_i,y_j), && (2.8) \end{aligned}$$

où $\mathcal{X}$ et $\mathcal{Y}$ sont respectivement les espaces d'épreuves de X et Y.

D'une manière analogue, on peut définir l'entropie conditionnelle $H(X|Y)$ (ou ambiguité) comme l'incertitude qui reste sur X lorsque Y est connue [9] :

$$H(X|Y) = -\sum_{i=1}^{N}\sum_{j=1}^{M} p(x_i,y_i)\log_b p(x_i|y_j). \tag{2.9}$$

Ici le conditionnement diminue l'incertitude, et donc :

$$H(X) \geq H(X|Y). \tag{2.10}$$

L'égalité est vérifiée quand X et Y sont indépendantes.

L'entropie conjointe d'une paire de variables X et Y est la somme de l'entropie de X (resp. Y) et de l'entropie de Y (resp. X) conditionnellement à l'observation X (resp. Y) :

$$H(X,Y) = H(X) + H(Y|X) = H(Y) + H(X|Y). \tag{2.11}$$

2.1.4 Entropie relative et information mutuelle

L'entropie relative peut être interprétée comme une mesure de distance entre deux distributions $p(x)$ et $q(x)$. Ce n'est néanmoins pas une distance au sens mathématique puisque sa relation n'est pas symétrique [10] :

$$D(p\,\|q) = \sum_{x\in\mathcal{X}} p(x)\log_b \frac{p(x)}{q(x)} = E_p\left[\log_b \frac{p(X)}{q(X)}\right]. \tag{2.12}$$

On considère dans cette définition que $0 \times \log_b \frac{0}{q} = 0$ et $p\log_b \frac{p}{0} = \infty$. Il est également important de noter que $D(p\,\|q) \neq D(q\,\|p)$. L'entropie relative $D(p\,\|q)$ s'interprète aussi comme une mesure de l'inefficacité lorsque l'on suppose la distribution q alors que la vrai distribution est p.

Considérons maintenant deux variables aléatoires X et Y avec une probabilité conjointe $p(x,y)$ et des probabilités marginales $p(x)$ et $p(y)$. L'information mutuelle moyenne $I(X;Y)$ est l'entropie relative entre la distribution conjointe et le produit des distributions $p(x)p(y)$, c'est-à-dire [9] :

$$\begin{aligned} I(X;Y) &= \sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x,y)\log_b \frac{p(x,y)}{p(x)p(y)} && (2.13)\\ &= D(p(x,y)\,\|p(x)q(x)) && (2.14)\\ &= E_{p(x,y)}\left[\log_b \frac{p(X,Y)}{p(X)p(Y)}\right]. && (2.15) \end{aligned}$$

$I(X;Y)$ est la mesure de la quantité d'information que la variable aléatoire X contient par rapport à la variable aléatoire Y. C'est donc ainsi la réduction de l'incertitude sur X par la connaissance de Y.

L'information mutuelle est liée à l'entropie par les relations ci-dessous :

$$\begin{aligned} I(X;Y) &= H(X) - H(X|Y) && (2.16)\\ &= H(Y) - H(Y|X) && (2.17)\\ &= H(X) + H(Y) - H(X,Y). && (2.18) \end{aligned}$$

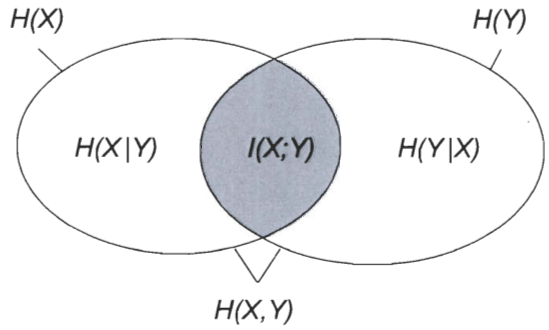


FIGURE 2.2 – Relation entre l'entropie et l'information mutuelle.

Ces différentes relations peuvent d'ailleurs être représentées à l'aide du diagramme de Venn de la figure 2.2. On note également que $I(X;Y) = I(Y;X)$.

L'information mutuelle moyenne $I(X;Y)$ entre les deux variables aléatoires X et Y est positive ou nulle :

$$I(X;Y) \geq 0. \tag{2.19}$$

L'égalité se vérifie lorsque X et Y sont statistiquement indépendantes.

2.1.5 Relations générales

L'entropie d'un ensemble de variables aléatoires est la somme des entropies conditionnelles. L'ensemble des variables aléatoires $X_1, X_2, \ldots, X_N$ suit la probabilité conjointe $p(x_1, x_2, \ldots, x_n)$, et donc [9] :

$$H(X_1, X_2, \ldots, X_N) = \sum_{i=1}^{n} H(X_i|X_{i-1}, \ldots, X_1). \tag{2.20}$$

De même nous avons pour l'entropie conditionnelle :

$$H(X_1, X_2, \ldots, X_N|Y) = \sum_{i=1}^{n} H(X_i|X_{i-1}, \ldots, X_1, Y). \tag{2.21}$$

L'information mutuelle moyenne s'exprime également comme suit [9] :

$$\begin{aligned} I(X_1, \ldots, X_N; Y) &= \sum_{i=1}^{n} H(X_i|X_{i-1}, \ldots, X_1) - \sum_{i=1}^{n} H(X_i|X_{i-1}, \ldots, X_1, Y) \\ &= \sum_{i=1}^{n} I(X_i; Y|X_1, \ldots, X_{i-1}). \end{aligned} \tag{2.22}$$

Exemple : Considérons les variables aléatoires X, Y et Z rattachées à l'espace d'épreuves conjoint $\mathcal{XYZ}$. Nous pouvons dès lors obtenir les relations suivantes :

$$I(X;Y|Z) = H(X|Z) - H(X|Y,Z), \tag{2.23}$$
$$I(X;Y,Z) = I(X;Z) + I(X;Y|Z), \tag{2.24}$$
$$I(X;Y;Z) = I(X;Y) - I(X;Y|Z). \tag{2.25}$$

$I(X;Y;Z)$ représente l'information commune aux trois variables aléatoires. Beaucoup de relations sont envisageables ici et toutes les écrire serait inutile. Elles peuvent être néanmoins déduites par le diagramme de Venn à trois variables suivant :

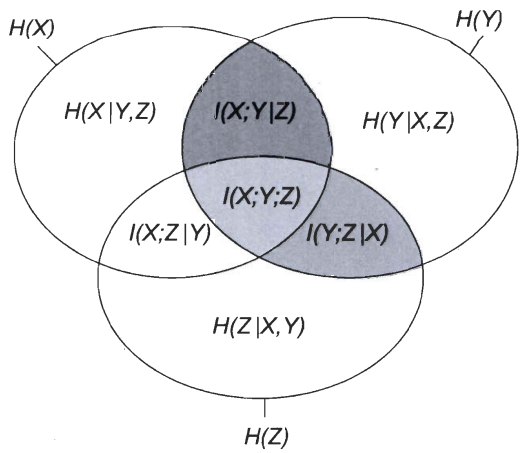


FIGURE 2.3 – Diagramme de Venn à 3 variables aléatoires.

2.2 Fonction de débit-distorsion

Pour les codes à longueur fixe, le théorème du codage de source (cf. Annexe A) énonce le fait qu'il est possible de compresser des données sans perte d'information tant que le débit R en bit/symbole vérifie $R > H(X)$, avec $H(X)$ l'entropie de la source et ceci dans un cas théorique et optimal. Néanmoins il existe des cas où $H(X)$ peut être trop grand par rapport au débit R que le système de transmission est prêt à supporter. La question est alors de savoir quel niveau de distorsion on est prêt à accepter entre l'information brute en sortie de la source et l'information en sortie du codeur.

2.2.1 Définitions

Supposons que nous disposions d'une source produisant des séquences de variables aléatoires i.i.d. $X^n = \{X_1, X_2, \ldots, X_n\}$ suivant $p(x)$ et $x \in \mathcal{X}$. L'encodeur décrit alors la séquence X^n par un mot-code d'indice $f_n(X^n) \in \{1, 2, \ldots, J^{nR}\}$ où J est la taille

de l'alphabet servant à la représentation d'un composant élémentaire du mot-code. Le décodeur calcule ensuite une estimée $\hat{X}^n \in \hat{\mathcal{X}}^n$ de la séquence initiale comme nous l'expose la figure 2.4.

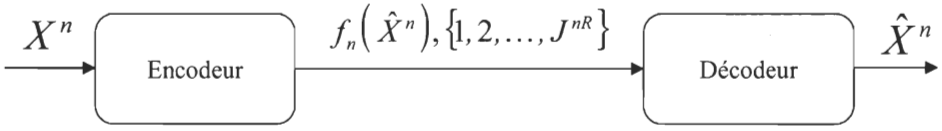


FIGURE 2.4 – Chaîne de codage et décodage.

La mesure de distorsion[2] $d(x, \hat{x})$ se résume en définitive à une mesure de la distance entre le signal reçu (décodé) $\hat{X}^n$ et le signal initial X^n. La distorsion est définie symbole par symbole et fournit la mesure du coût de représentation du symbole x par le symbole $\hat{x}$. Cette mesure est bornée si la valeur maximale de distorsion est finie [9] :

$$d_{max} \stackrel{def}{=} \max_{x \in \mathcal{X}, \hat{x} \in \hat{\mathcal{X}}} d(x, \hat{x}) < \infty. \tag{2.26}$$

Parmi les divers critères de distorsion, nous observons principalement [10] :

- **la distortion de Hamming** où :

$$d(x, \hat{x}) = \begin{cases} 1 & \text{si } x \neq \hat{x}, \\ 0 & \text{si } x = \hat{x}; \end{cases} \tag{2.27}$$

- **l'erreur quadratique** qui est la plus usuelle de par son caractère assez proche de l'écart-type et de son utilisation adaptée pour les alphabets continus :

$$d(x, \hat{x}) = (x - \hat{x})^2. \tag{2.28}$$

Il est aussi admis que la position de l'*erreur* de codage n'est pas importante et qu'elle est indépendante du contexte. En étendant la définition à une séquence entière, la distorsion entre les séquences x^n et $\hat{x}^n$ vaut alors :

$$d(x^n, \hat{x}^n) = \frac{1}{n} \sum_{i=1}^{n} d(x_i, \hat{x}_i). \tag{2.29}$$

La distorsion d'une séquence est donc la moyenne de la distorsion par symbole. Supposons maintenant que nous voudrions mesurer la distorsion moyenne[3] D associée à la source. Il suffit pour cela de calculer l'espérance de la distorsion par séquence. En effet, en considérant le message délivré par la source comme un processus aléatoire,

2. Toujours supposée positive ou nulle : $d(x, \hat{x}) \geq 0$.
3. $D \geq 0$.

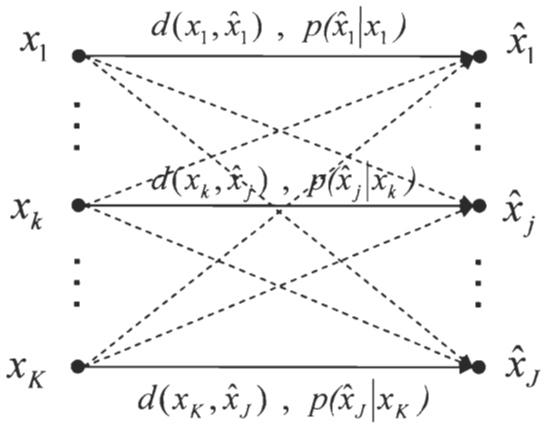


FIGURE 2.5 – Mesures et probabilités de distorsion.

$d(X^n, \hat{X}^n)$ sera aussi une variable aléatoire [9]. On définit donc la distorsion de la source comme l'espérance de cette variable aléatoire :

$$D = E\left[d(X^n, \hat{X}^n)\right] \tag{2.30}$$

$$= \sum_{x^n}\sum_{\hat{x}^n} p(x^n, \hat{x}^n) d(x^n, \hat{x}^n) \tag{2.31}$$

$$= \sum_{x^n}\sum_{\hat{x}^n} p(x^n) p(\hat{x}^n|x^n) d(x^n, \hat{x}^n). \tag{2.32}$$

La paire débit-distorsion (R, D) est dite réalisable si il existe des séquences du code (J^{nR}, n) telles que :

$$\lim_{n\to\infty} E\left[d(X^n, \hat{X}^n)\right] \leq D. \tag{2.33}$$

La région débit-distorsion d'une source se désigne comme l'ensemble clos des paires (R, D) réalisables. De même, la fonction de distorsion [4] $R(D)$ représente l'ensemble des débits R tel que le couple (R, D) fait partie de cette région et ce pour une distorsion donnée.

2.2.2 Fonction de distorsion

Le débit minimum R bit/symbole nécessaire pour reproduire l'information avec une distorsion égale ou inférieure à D est donné par la fonction de distorsion [9]. Pour une source X présentant les mesures de distorsion $d(x_l, \hat{x}_m)$, cette fonction se détermine comme suit :

$$R(D) \equiv \min_{\mathcal{P}_D} I(X; \hat{X}), \tag{2.34}$$

4. On définit également la fonction de débit $D(R)$ comme l'ensemble des distorsions D où, pour un débit R donné, les couples (R, D) appartiennent à la région débit-distorsion.

où $\mathcal{P}_D$ est l'ensemble des matrices de probabilités de transition D-admissibles :

$$\mathcal{P}_D = \left\{ \mathbf{P} = \{p(\hat{x}_l|x_m\} : \sum_{l=1}^{L} \sum_{m=1}^{M} p(x_l)p(\hat{x}_m|x_l)d(x_l, \hat{x}_m) \leq D \right\}. \tag{2.35}$$

Ici il est démontré que le critère de distorsion D est borné par une valeur minimum et maximum :

$$d_{min} \leq D \leq d_{max}, \tag{2.36}$$

avec

$$d_{min} = \sum_{k=1}^{K} p(x_k) \cdot \min_{j=1,\ldots,J} d(x_k, \hat{x}_j). \tag{2.37}$$

De même, on caractérise d_{max} par la relation :

$$d_{max} = \min_{j=1,\ldots,J} \sum_{k=1}^{K} p(x_k) d(x_k, \hat{x}_j). \tag{2.38}$$

Exemple d'une source binaire

Pour une source binaire sans mémoire avec $P(X_i = 1) = 1 - P(X_i = 0) = p$, et en considérant la distorsion de Hamming, il peut être démontré que la fonction de distorsion $R(D)$ est égale à :

$$R(D) = \begin{cases} H(p) - H(D), & 0 \leq D \leq \min\{p, 1-p\}, \\ 0, & D \geq \min\{p, 1-p\}, \end{cases} \tag{2.39}$$

où $H(p)$ est l'entropie d'une valeur aléatoire binaire.

La figure 2.6 représente les courbes de distorsion pour les valeurs $p = 0.5$ et $p = 0.05$. Ainsi pour un symbole ayant une probabilité d'apparition de $p = 0.5$, pour $D = 0.11$ on a $R(D) = 0.5$. C'est-à-dire que pour un codeur fournissant un débit de 0.5 bit/symbole (donc un taux de compression de 2), celui-ci engendrera au minimum 11% de distorsion.

2.2.3 Théorème de débit-distorsion

Ce paragraphe énonce le troisième théorème donné par Claude Shannon. Il stipule que pour une source $\mathbf{X}$ i.i.d. avec une distribution de probabilité $\mathbf{p} = \{p(\mathbf{x})\}$ et présentant les mesures de distorsion $d(\mathbf{x}, \hat{\mathbf{x}})$, il est possible de déterminer un code de compression de source $\mathcal{C}$ et de débit R tel que la distorsion moyenne par symbole est inférieure au critère de distorsion $D + \epsilon_1$ [10].

Le débit R doit être alors supérieur à la fonction de distorsion $R(D) = \min_{\mathcal{P}_D} I(X; \hat{X})$ et la longueur des mot-codes suffisamment grande ($N \geq N_0$) :

$$R \geq R(D) + \epsilon_2. \tag{2.40}$$

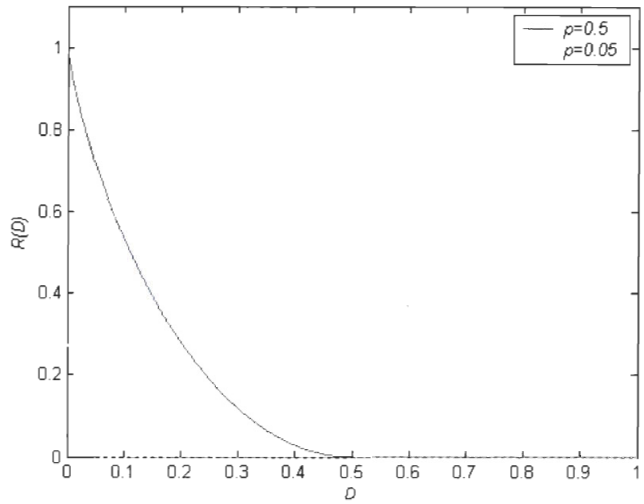


FIGURE 2.6 – Fonction de distorsion d'une variable aléatoire binaire.

Remarque : Le tracé de $R(D)$ pour $p = 0.05$ de la figure 2.6 est révélateur de l'intérêt du codage pour une source dont les symboles ne sont nullement équiprobables. En effet, on peut atteindre dans ce cas un taux de compression de $1/R(D) = 1/0.2423 \approx 4$ sans engendrer de distorsion entre l'information portée par les symboles émis par la source et ceux fournis par le codeur.

2.3 Le codage de source distribuées

2.3.1 Théorème de Slepian-Wolf

Nous savons que le codage d'une source X requiert un débit minimum $R \geq H(X)$. De même, pour deux sources d'information corrélées (X, Y) suivant la distribution de probabilités $p(x, y)$, un débit $R \geq H(X, Y)$ est suffisant si celles-ci sont encodées conjointement. En revanche, pour des sources encodées séparément, Slepian et Wolf ont démontré qu'un débit $R_X + R_Y \geq H(X, Y)$ est aussi suffisant sous une opération de décodage conjointe [1]. Dans le cas d'une compression sans perte, le théorème de Slepian-Wolf stipule alors que le débit optimal, obtenu lors d'un codage et décodage conjoint de plusieurs sources corrélées, peut théoriquement être atteint par un encodage indépendant et un décodage conjoint des sources. Ce théorème peut être considéré comme un aspect *distribué* du codage entropique [5] [8].

Soient les sources corrélées $(X_{1i}, X_{2i}, \ldots, X_{mi})$ suivant $p(x_1, x_2, \ldots, x_m)$. L'ensemble des vecteurs de débits réalisables pour le codage de sources distribuées avec des enco-

5. Le codage entropique est une autre appellation du codage de source sans perte d'information. Le codage de Huffman en est un exemple.

deurs séparés et un décodeur commun est alors défini comme suit [10] :

$$R(\mathcal{S}) > H\big(X(\mathcal{S})|X(\mathcal{S}^c)\big) \text{ pour tout } \mathcal{S} \subseteq \{1, 2, \ldots, m\}\,, \tag{2.41}$$

avec

$$R(\mathcal{S}) = \sum_{i \in \mathcal{S}} R_i \quad \text{et} \quad X(\mathcal{S}) = \{X_j : j \in \mathcal{S}\}\,.$$

Cas de deux sources

La figure 2.7 donne une illustration du théorème de Slepian-Wolf dans le cas de deux sources discrètes. Ainsi pour deux sources corrélées X et Y, il est possible de les transmettre à des débits tels que $R_X + R_Y \geq H(X,Y)$, $R_X \geq H(X|Y)$ et $R_Y \geq H(Y|X)$ sans qu'aucune perte d'information ne soit à remarquer. Les séquences d'informations doivent tout de même être suffisamment grandes.

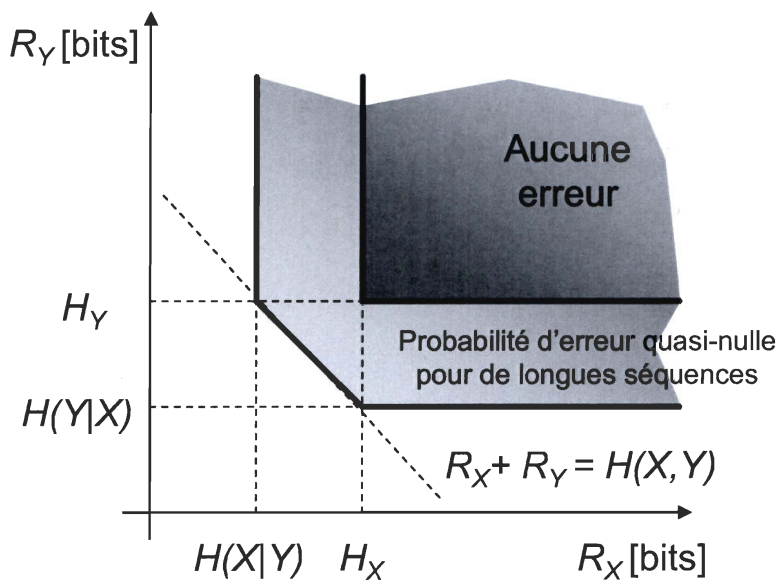


FIGURE 2.7 – Illustration du théorème de Slepian-Wolf pour deux sources.

La somme des débits peut donc atteindre l'entropie conjointe $H(X,Y) = H(X) + H(Y|X)$ comme pour un encodage conjoint de X et Y et ce malgré des encodeurs séparés. Supposons que $R_X = H(X)$ et $R_Y = H(Y|X)$, alors pour la séquence codée X^n de $nH(X)$ bits, il est possible d'encoder efficacement de telle sorte que l'on obtienne une faible probabilité d'erreur au décodage.

2.3.2 Théorème de Wyner-Ziv

En 1976, les travaux de Slepian-Wolf ont été repris par Wyner et Ziv [2]. Ces derniers ont permis d'établir les limites théoriques de débit-distorsion dans le cas d'un système

de codage de sources distribuées où un processus à perte d'information était pris en compte. Le principe du codage/décodage de Wyner-Ziv est donné par la figure 2.8. Dans ce cas-ci on suppose que la source Y a été transmise à un débit $R_Y = H(Y)$.

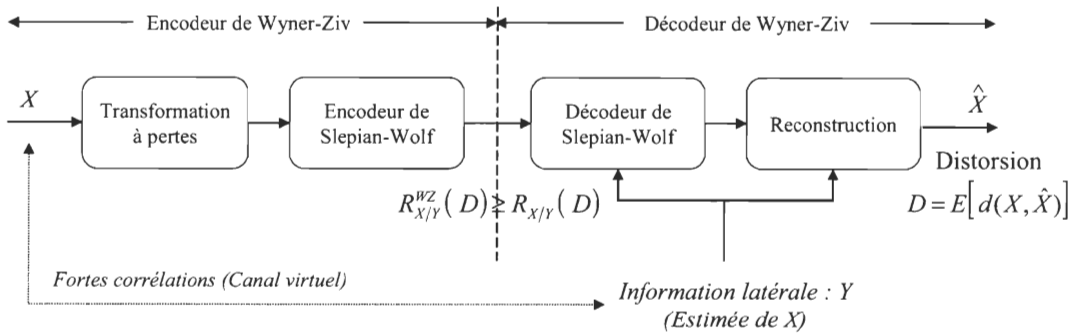


FIGURE 2.8 – Codage de source avec information latérale.

En règle générale, un codeur de Wyner-Ziv résulte de la concaténation d'un système à perte d'information et d'un codeur de Slepian-Wolf. Comme nous venons de le voir, ce dernier n'engendre aucune perte.

L'objectif du codage de sources distribuées est de compresser des signaux fortement corrélés codés au préalable séparément mais devant être décodés conjointement. Plusieurs articles traitent de ce problème pour essayer d'atteindre la limite de Slepian-Wolf. Les techniques actuelles les plus performantes reposent sur l'utilisation des turbo-codes et des codes LDPC [11]. Ces derniers sont apparemment les plus performants pour cette application [11].

2.3.3 Fonction de distorsion et information latérale

Nous savons que pour décrire une source X avec une certaine propension de distorsion D, $R(D)$ bits sont suffisants. Le théorème énoncé dans cette partie permet cette fois de calculer les bits nécessaires pour décrire une source X sachant qu'une information latérale Y est disponible au niveau du décodeur (cf. Figure 2.8).

La fonction de distorsion avec information latérale $R_Y(D)$ est définie comme le débit minimum suffisant pour permettre une distorsion D si l'information latérale Y est disponible au niveau du décodeur [10]. Soient deux sources corrélées $(X, Y) \sim p(x, y)$, la fonction de distorsion prenant en compte l'information latérale se détermine par :

$$R_Y(D) = \min_{p(w|x)} \min_{f} \big(I(X;W) - I(Y;W)\big). \tag{2.42}$$

où W est la source issue du système à perte d'information. Ce dernier peut être modélisé par un canal avec les probabilités de transition $p(w|x)$. La minimisation de la distorsion s'effectue à la fois à travers toutes les fonctions $f : \mathcal{Y} \times \mathcal{W} \rightarrow \hat{\mathcal{X}}$ et toutes les probabilités

conditionnelles $p(w|x)$, $|\mathcal{W}| \leq |\mathcal{X}| + 1$, telles que :

$$\sum_x \sum_y \sum_z p(x,y)p(w|x)d(x, f(y,w)) \leq D. \tag{2.43}$$

La fonction f correspond ici au décodage des symboles transformés W par l'intermédiaire de l'information latérale Y : $\hat{X} = f(Y, W)$. On minimise alors sur l'ensemble des probabilités conditionnelles en rapport à W et toutes les fonctions f pour obtenir ainsi une distorsion inférieure à D.

2.4 Conclusion

Ce chapitre nous a permis d'introduire certains fondements de la théorie de l'information. Sans pour autant recourir aux démonstrations des théorèmes énoncés, il regroupe l'ensemble des aspects importants qui ont été utiles durant ce travail de doctorat. En effet, la dernière section qui portait plus particulièrement sur l'aspect distribué de l'information, relate des travaux menées par Slepian et Wolf [1] ainsi que ceux de Wyner et Ziv [12, 2] sur le codage de source distribué.

Chapitre 3

Le codage vidéo

3.1 La vidéo numérique

La vidéo numérique vient dans le cas où une image originale est générée sous la forme d'une série de pixels (*picture elements*). Il semble intuitivement qu'une image produite par ce moyen soit de qualité moindre qu'une image analogique [7]. Cette dernière semble posséder une résolution infinie. En pratique en revanche, sa haute résolution n'est aucunement un avantage dès qu'il s'agit de la regarder sur un écran où la résolution est fixée. De ce fait, la vidéo numérique possède bon nombre d'avantages :

- elle est aisément modifiable ;
- elle peut-être facilement stockée sur un support numérique. Des codes correcteurs d'erreurs peuvent ainsi venir en addition pour augmenter la fiabilité. Des copies peuvent alors être produites sans pertes de qualité ;
- elle peut-être compressée afin de permettre un stockage plus important et/ou une transmission plus rapide.

En principe, la vidéo numérique est une succession d'images, appelées plus communément trames, affichées selon un certain nombre d'images par seconde (*frame rate* exprimé en fps) pour créer l'illusion d'une animation [7]. Chaque pixel d'une image est préalablement codé suivant un modèle de représentation des couleurs. Le plus connu est l'espace colorimétrique RGB (*Red-Green-Blue*) correspondant aux trois couleurs primaires [1] caractéristiques de la perception humaine [13]. Cet espace est basé sur une synthèse additive des couleurs, c'est-à-dire que le mélange des trois composantes dans des proportions diverses permet de reproduire à l'écran une part importante du spectre visible sans avoir à spécifier une multitude de fréquences lumineuses. Pour les applica-

1. Le rouge, vert et bleu sont les couleurs primaire de la synthèse additive (leur addition donne du blanc). Par compte, les couleurs primaires pour la synthèse soustractive sont le cyan, le majenta et le jaune (leur addition fournit du noir).

tions les plus courantes, les composantes sont codées selon une échelle de 256 valeurs (0 à 255 soit 8 bits) correspondant à leur niveau d'intensité. Ceci détermine aussi la profondeur du pixel qui représente le nombre total de bits sur lequel il est codé, soit 24 bits pour ce cas ci. Le nombre de couleurs pouvant être ainsi représenté est de $256 \times 256 \times 256$ possibilités, soit près de 16 millions de couleurs.

Bien que le système RGB soit très utilisé dans l'acquisition et l'affichage de l'image, pour ce qui est du traitement des couleurs et de la compression en revanche, le système de luminance-chrominance[2] est plus efficace et donc plus largement utilisé. Ceci est étroitement lié à la perception des couleurs du système visuel humain. En effet il est connu que ce dernier est plus sensible au vert qu'au rouge et moins sensible au bleu. Ainsi une égale représentation du rouge, vert et bleu mène à une représentation inefficace des données. Les normes internationales de compression vidéo utilisent préférablement la représentation colorimétrique YCrCb où Y correspond à la luminance et Cr-Cb aux chrominances rouge et bleu respectivement [13]. La relation reliant les deux espaces RGB et YCrCb est simplement linéaire :

$$\begin{pmatrix} Y \\ Cb \\ Cr \end{pmatrix} = \begin{pmatrix} 0.257 & 0.504 & 0.098 \\ -0.148 & -0.291 & 0.439 \\ 0.439 & -0.369 & -0.071 \end{pmatrix} . \begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} 16 \\ 128 \\ 128 \end{pmatrix}.$$

Des expériences psychovisuelles ont montré que le système de vision humaine est plus sensible aux hautes fréquences spatiales de la luminance Y que celles des chrominances Cr et Cb. Ceci a permis par la suite le sous-échantillonnage spatial de ces dernières sans apporter de détérioration notable au contenu visuel. Trois types de sous-échantillonnage des couleurs ont été normalisés :

- 4:4:4 $\Longrightarrow$ aucun sous-échantillonnage des chrominances.
- 4:2:2 $\Longrightarrow$ pour chaque 4 échantillons de luminance Y seulement deux de chrominance Cr et deux de chrominance Cb sont encodés.
- 4:2:0 $\Longrightarrow$ pour chaque 4 échantillons de luminance Y on encode seulement un échantillon de chrominance Cr et un de chrominance Cb.

Le nombre d'images par seconde ainsi que la profondeur des pixels dépendent intrinsèquement du type d'application. Le tableau 3.1 expose quelques applications typiques avec les paramètres y étant associés.

Remarque : Un calcul simple permet de montrer l'intérêt de la compression vidéo. Par exemple, pour une séquence vidéo en HDTV progressif 1080 lignes (1080p), la vidéo à elle seule requerrait sans compression $30 \times 1920 \times 1080 \times 24$ bits par seconde

2. On recense les modèles tels que : HSI, YUV, YCbCr et YDbDr.

Application	Débit des trames (fps)	Dimension	Profondeur (bits)
Surveillance	5	640×480	12
Téléphonie vidéo	10	320×240	12
Multimédia	15	320×240	16
TV	25	640×480	24
HDTV (720p)	60	1280×720	24
HDTV (1080i)	60	1920×1080	24
HDTV (1080p)	30	1920×1080	24

TABLE 3.1 – Paramètres d'applications vidéo typiques [7].

(bps), soit approximativement 1.493 Gbps ($\approx$ 187 Mo/s). À titre de comparaison, les derniers lecteurs *Blu-ray* possèdent une vitesse de lecture en $\times 2$ correspondant à un débit maximal de 72 Mbps ($\approx$ 9 Mo/s). C'est dire l'intérêt primordial de la compression dans ce genre de cas.

3.2 La compression vidéo

La compression vidéo repose sur deux principes de base [7]. Le premier exploite les redondances spatiales qui existent dans chaque image. Le second repose sur la caractéristique qu'une image peut-être très similaire aux images voisines. C'est ce que l'on appelle la redondance temporelle. Une technique typique de compression vidéo devrait donc débuter par encoder la première image par une méthode classique de compression d'image, puis encoder chacune des images successives en identifiant les différences avec les images antérieures, et enfin d'encoder ces différences. Si l'image est très différente de ses voisines, elle devra être encodée indépendamment. Dans la littérature, cette image sera dite *intra* alors qu'une image codée par l'intermédiaire de ses prédécesseurs sera identifiée comme une *inter* image. Nous verrons plus tard que ce qui distingue cette compression traditionnelle de la compression vidéo dite *distribuée* est que pour cette dernière les images subissent un encodage indépendant les unes des autres. Le processus de décodage tiendra cependant compte de la corrélation entre les images.

La compression vidéo se fait généralement avec perte. Encoder une image T_i à l'aide d'une image antérieure T_{i-1} introduit de la distorsion. Il est aussi montré qu'encoder l'image T_{i+1} à l'aide de l'image T_i augmente cette distorsion. Pour éviter qu'une distorsion trop importante n'apparaisse et ne dégrade le rendu vidéo, les intra-images[3] sont régulièrement utilisées dans la séquence pour maintenir une certaine cohérence dans le rendu vidéo. En définitive, trois types d'images sont nécessaires pour effectuer le codage différentiel et le codage bidirectionnel procurant un minimum d'erreurs de propagation :

3. L'appellation images clés K (*Key frames*) est également employée.

- **Image I ou K** (*Intra*) : intra-codée et ne nécessite pas d'information additionnelle pour être décodée.
- **Image P** (*Predictive*) : déduite d'une image antérieure qui peut être de type I ou P.
- **Image B** (*Bidirectional*) : se calcule de manière bi-directionnelle depuis les images antérieures ou postérieures de type I ou P. L'estimation de mouvement (cf. § 3.2.1) résulte dans ce cas-ci en deux vecteurs de mouvement ; un pour chacune des images références, passé et futur, pointant respectivement sur les meilleurs blocs $M_{t'}^{-}$ et $M_{t'}^{+}$. Le bloc d'erreur résiduelle (nécessaire à l'étape suivante de codage) est calculé comme suit [14] :

$$E_t \triangleq B_t - \frac{1}{2}\left(M_{t'}^{-} + M_{t'}^{+}\right) \ , \text{ avec } t' < t. \tag{3.1}$$

Usuellement, une séquence vidéo est segmentée selon des groupes d'images consécutifs (GOP : *Group of Pictures*) débutant par une image I et suivis d'images de type P puis B dans des positions bien spécifiées (exemple d'ordre d'affichage : I P P I P...). Les images constituant le GOP peuvent avoir des ordonnancements différents suivant l'encodage ou l'affichage. En effet, les images B sont encodées après le calcul des images P bien qu'elles apparaissent antérieurement à ces dernières lors de l'affichage. D'après l'exemple précédent la séquence d'encodage serait [14] : IP P P IP ...

En règle générale, la rapidité d'un encodeur importe peu dès l'instant que le décodage (et donc l'affichage) soit rapide [7]. Pour la compression vidéo distribuée en revanche, l'asymétrie de complexité se trouve inversée : l'encodeur est simple mais le décodeur complexe.

3.2.1 Estimation et compensation de mouvement

Comme nous venons de le voir, une séquence animée est constituée d'une suite d'images ayant comme particularité une différence entre deux images successives faible. Cette caractéristique peut être exploitée afin d'obtenir une compression. Si l'encodeur distingue qu'une partie de l'image précédente a été déplacée vers une position différente de l'image courante, cette partie pourra être compressée en écrivant dans le flux de données [7] : (i) sa position antérieure ; (ii) sa position actuelle ; (iii) les informations permettant d'identifier ses délimitations.

En principe, une telle partie peut posséder n'importe quelle forme. En pratique néanmoins, l'image est découpée selon des blocs de taille égale [7]. Ils peuvent être rectangulaires mais sont plus souvent carrés ($N \times N$). Les tailles habituellement rencontrées sont : 4×4, 8×8, 16×16.

Pour chaque bloc B_t de l'image courante, l'encodeur cherche dans l'image *précédente* un bloc M_{t-1} minimisant un certain critère. Une fois le bloc trouvé, l'encodeur calcule la différence entre sa position dans l'image courante et sa position dans l'image précédente,

formant ainsi le vecteur de mouvement [4] associé au bloc. Cette estimation de mouvement est efficace si les objets subissent une translation et non un changement d'échelle ou une rotation [7]. En général, ce processus s'ensuit d'une perte d'information.

L'algorithme *Full Search*

Bien qu'il soit le plus simple des algorithmes de recherche de vecteurs de mouvement, celui-ci requiert énormément de calcul. Pour un bloc donné B_t d'une image au temps t l'algorithme cherche un bloc identique ou très proche M_{t-1} dans l'image du temps $t-1$. Afin de limiter les calculs, cette recherche est restreinte dans une *zone de recherche* définie par un déplacement maximum dx et dy. Il va de soi que ces déplacements doivent être en adéquation avec l'amplitude globale du mouvement si l'on veut limiter la dissimilitude. Pour un bloc $N \times N$, la zone de recherche contiendra en fin de compte $(N+2dx) \times (N+2dy)$ pixels et donc $(2dx+1) \times (2dy+1)$ blocs distincts (cf. Fig. 3.1) [7]. Le nombre de blocs candidats est alors proportionnel à $dx \times dy$. D'autres algorithmes existent permettant de limiter la complexité et d'accélérer la recherche ; ils ne seront pas expliqués ici.

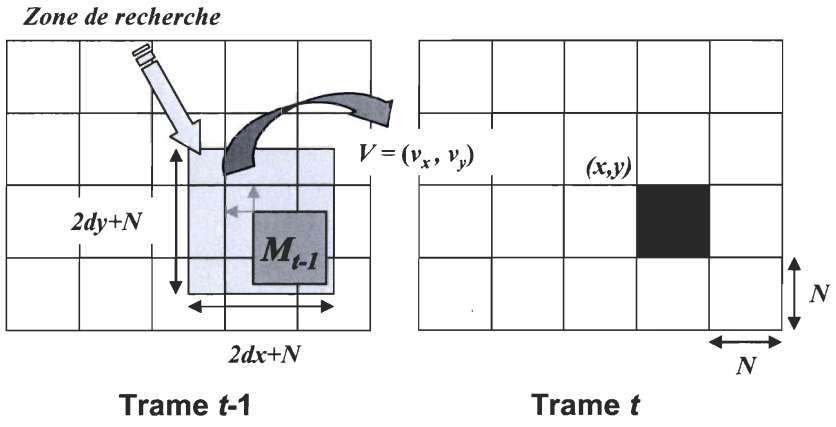


FIGURE 3.1 – Calcul des vecteurs de mouvement.

Remarque : pour certains blocs proches des bords, il arrive que la zone de recherche vienne en partie hors de l'image. Pour palier à cela, une technique simple est le redimensionnement de l'image en incluant une bande de largeur dx sur les cotés gauche et droite et une bande de hauteur dy sur le haut et bas. L'intérieur de ces bandes correspondra

4. Dans les normes de compression vidéo, la recherche des vecteurs de mouvement se fait plutôt pour des *macroblocs* (groupement de 4 blocs 8×8) pour la luminance et pour des blocs 8×8 pour les chrominances. Ceci est en fait compatible avec le format de sous-échantillonnage 4:2:0. Si les formats 4:4:4 et 4:2:2 sont utilisés, le groupement sera modifié en accord avec ces derniers.

à la copie exacte des pixels des bords de l'image, un pixel étant recopié dx ou dy fois. Les quatre pixels des coins de l'image seront en outre recopiés $(dx \times dy) + dx + dy$ fois.

Critère de dissemblance

Il a été mentionné précédemment que le choix du meilleur bloc M_{t-1} s'effectue d'après la minimisation d'un certain critère évaluant la dissemblance entre les blocs. Dans la littérature, il en existe une multitude tel que la distance euclidienne. Le critère de dissemblance le plus couramment employé se base néanmoins sur le calcul de la différence moyenne absolue (MAE : *Mean Absolute Error*) :

$$MAE_{(x,y)_t} = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} \left| f_t\left(x+i, y+j\right) - f_{t-1}\left(x+i-v_x, y+j-v_y\right) \right|, \tag{3.2}$$

où $V = (v_x, v_y)$ est un vecteur de mouvement potentiel du bloc B_t. Dans ce cas-ci, on dénombre alors N^2 soustractions pour chacun des $(2dx+1) \times (2dy+1)$ blocs candidats.

Erreur résiduelle

Une fois les vecteurs de mouvement calculés, l'erreur résiduelle du bloc (ou erreur de prédiction) sera finalement codée [14]. Ce principe forme ce qu'on appelle la compensation de mouvement. Ceci sera préférable au codage des valeurs initiales des pixels puisque que l'erreur résiduelle présente une dynamique plus faible. La quantification (cf. § 3.2.4) se fera alors sur un nombre de bits moins important procurant ainsi une compression.

$$E_t \triangleq B_t - M_{t-1}. \tag{3.3}$$

3.2.2 Limitations

Les techniques de compensation de mouvement basées sur le partitionnement en blocs sont largement utilisées en codage vidéo. Cependant bien qu'elles soient très simples et efficaces, elles présentent quelques limitations [13]. Elles possèdent tout d'abord un champ de vecteurs de mouvement peu représentatif du mouvement réel, en particulier pour les zones proches des limites d'objets en mouvement. Deuxièmement, ces techniques causent des discontinuités très visibles au niveau des jonctions entre les blocs.

Tous ces désavantages sont essentiellement dus au fait que chaque bloc est supposé subir une translation uniforme et que les vecteurs de mouvement sont estimés indépendamment des uns des autres. De plus, l'estimation de mouvement au niveau des limites d'objets en mouvement est à l'origine d'une forte proportion d'erreurs de prédiction, réduisant ainsi l'efficacité du codage.

Les artéfacts visuels ne causent pourtant pas de dégradation perceptible pour le système visuel humain lorsque le débit d'encodage est relativement élevé. En effet, une quantité d'information suffisante de l'erreur résiduelle est transmise, augmentant de ce fait l'effet subjectif. En revanche, lors d'un débit binaire plus faible (< 64 kbps), les artéfacts visuels peuvent devenir déplaisants.

3.2.3 La transformation DCT

Les techniques de compensation de mouvement sont utilisées afin de retirer la corrélation temporelle. En revanche, la transformation en cosinus discret (DCT : *Discrete Cosine Transform*) permet de diminuer la corrélation spatiale au sein d'une image [14]. C'est une transformation réversible qui est utilisée indifféremment sur des blocs $N \times N$ de luminance et de chrominance. De plus, elle s'applique directement sur les valeurs originales des pixels pour les images I ainsi que sur les blocs d'erreurs de prédiction pour les images P et B. Soit $\mathbf{X}$, un bloc $N \times N$, le résultat $\mathbf{Y}$ issu de la transformée s'obtient comme suit [4] :

$$\mathbf{Y} = \mathbf{F}\mathbf{X}\mathbf{F}^T, \tag{3.4}$$

où la matrice de transformation $\mathbf{F}$ de taille $N \times N$ est de la forme :

$$F_{k\ell} = \begin{cases} \sqrt{\frac{2}{N}} \cos\left(\frac{\Pi}{N} k \left(\ell + \frac{1}{2}\right)\right) & , \forall\, k = 1 \rightarrow N-1,\ \forall\, \ell = 0 \rightarrow N-1, \\ \frac{1}{\sqrt{N}} & , k = 0,\ \forall\, \ell = 0 \rightarrow N-1. \end{cases} \tag{3.5}$$

De la même manière, la transformée inverse s'obtient par la relation :

$$\mathbf{X} = \mathbf{F}^T\mathbf{Y}\mathbf{F}. \tag{3.6}$$

Les indices k et l représentent les coordonnées discrètes à l'intérieur de la matrice de transformation. Le résultat de (3.4) est une matrice $N \times N$ comprenant les coefficients dans le domaine transformé. L'élément du coin supérieur gauche (0,0) représente le coefficient DC (composante énergétique moyenne du signal) tandis que les autres éléments correspondent aux coefficients AC[5] : composantes des fréquences horizontales et verticales plus élevées. Ces fréquences augmentent en fonction de l'incrémentation des indices de coordonnées. Étant donné les valeurs possibles d'un pixel (0 à 255), les coefficients AC peuvent être négatifs tandis que le coefficient DC est toujours positif. Leurs représentations binaires peuvent alors différer.

Il est question ici de savoir combien de bits sont nécessaires pour représenter les coefficients du domaine transformé. Si l'on appelle V la valeur maximale absolue de $\mathbf{X}$ (habituellement de valeur 255) alors la valeur maximale de la transformation $\mathbf{FX}$ est donc $G \times V$ où G est le facteur multiplicatif maximal de cette transformation. Il

5. Les termes DC et AC trouvent leur origine du domaine du génie électrique où ils correspondent respectivement à *direct current* et *alternative current* [7].

est possible de montrer que $G = N \times F_{00} = \sqrt{N}$. Le calcul de l'amplitude maximale A_{max} de $\mathbf{Y}$ vaut alors : $A_{max} = G^2 \times V$. La représentation binaire peut par conséquent s'opérer sur un nombre de bits équivalent à $\lceil \log_2(A_{max}) \rceil$. En fait, A_{max} correspond plus particulièrement à l'amplitude maximale du coefficient DC. Les coefficients AC ont eux une amplitude deux fois moindre mais une dynamique semblable. Par exemple, dans le cas de blocs 4×4, les coefficients se représentent au maximum sur 10 bits . L'étendue dynamique s'étend ainsi de 0 à 1023 pour les coefficients DC et de -512 à 511 pour les coefficients AC afin de tenir compte du bit de signe.

Les propriétés de décorrélation de la transformation DCT ont été prouvées par de multiples expérimentations sur des images naturelles [14]. Elle affiche d'excellentes propriétés de compaction d'énergie. En pratique cela signifie que les coefficients résultants de la transformée, porteurs de la majorité de l'information, sont positionnés dans le coin supérieur gauche du bloc transformé. Ceux-ci correspondent en fin de compte aux basses fréquences, pour lesquelles l'oeil est plus sensible. L'information portée par les hautes fréquences peut ne pas être prise en compte pour opérer une compression. L'approximation de l'image originale est alors calculée par transformation inverse. Bien évidemment plus les coefficients de hautes fréquences seront écartés plus la dégradation sera perceptible. La transformée en cosinus discret joue un rôle très important dans le domaine de la compression vidéo. Elle est la transformée la plus utilisée à cause du nombre important d'implémentations rapides qu'il est possible de trouver.

La transformée en cosinus discret est cependant une transformation sous-optimale. La transformation de Karhunen-Loeve [15] offre quant à elle (aussi appelée transformation de Hotelling pour le mode discret) une décorrélation optimale des coefficients pour un signal markovien, et permet ainsi un meilleur stockage des données sous forme compacte [16]. Une image naturelle est un signal bidimensionnel qui peut être modélisée par un système de Markov du premier ordre (l'état présent ne dépend que de l'état précédent) de par la corrélation inter-pixel très forte. Plus particulièrement les pixels voisins ne sont pas indépendants des uns des autres. Cette transformation s'opère par la diagonalisation et le calcul des vecteurs propres de la matrice de covariance des coefficients. Ce procédé requiert énormément de calculs. Néanmoins dans le cas d'un processus de Markov, il existe une solution analytique approximant les vecteurs et les valeurs propres [15]. Les vecteurs de cette base sont calculés d'après les pixels originaux de l'image et sont donc dépendants des données. D'un point de vue pratique, ces vecteurs doivent être inclus dans le flux de données compressées pour permettre au décodeur la décompression. Si l'on combine cela au fait qu'aucune méthode rapide de calcul (du moins plus rapide que la DCT) n'a été découverte, alors tout ceci fait que la transformée de Karhunen-Loeve n'est pas très appropriée pour le cas d'applications pratiques.

6. Dans la norme JPEG ou MPEG, les coefficients de transformée sont représentés habituellement par des valeurs différentielles. L'étendue dynamique des coefficients est alors différente.

Parmi les différentes transformations sous-optimales existantes, il a été prouvé que la transformée en cosinus discret est la meilleure. Ses performances sont non seulement très proches de la transformée de Karhunen-Loeve mais de plus il existe plusieurs algorithmes rendant son implémentation très rapide. Comme nous l'avons vu, la DCT possède cette caractéristique très utile pour la compression permettant la décorrelation des données et la concentration de l'essentiel de l'énergie dans quelques coefficients (ceux notamment correspondant aux basses fréquences de l'image). La décompression ne nécessite quant à elle que des coefficients issus de la transformée, la rendant ainsi indépendante des données. La DCT reste cependant plus performante lorsque les données à compresser sont corrélées.

Remarque : En codage vidéo, le codage prédictif des images veut que la DCT utilise les différences entre les images. Dans un tel cas, il est vrai que la DCT ne contribue pas majoritairement à la compression puisque ces différences sont déjà décorrélées. En effet, la décomposition de ces différences en une suite de cosinus (variant en fréquence et en amplitude) requiert un nombre de coefficients plus important afin les caractériser (beaucoup de discontinuités et donc de hautes fréquences). Par compte, cette transformation reste très utile puisque qu'elle est suivie du processus de quantification (processus à perte d'information) qui pour le cas du codage prédictif peut être très important par rapport à l'amplitude des données (par exemple dans la norme MPEG-1, la table de quantification est uniforme et égale à 16) [7].

Une raison supplémentaire de l'utilisation de la DCT pour le codage prédictif est qu'ici les coefficients DC et AC ne sont pas substantiellement différents. Il n'y a donc aucune nécessité de les séparer durant l'encodage (c'est le cas pour le codage Intra des images).

3.2.4 La quantification

La quantification est un processus à perte d'information. Son principe consiste en la représentation d'un ensemble de valeurs de source (continues ou discrètes) par un ensemble moins important, c'est donc une application surjective. Elle est un des fondements de la compression à perte et engendre par la même occasion un impact direct sur le débit binaire et la distorsion des images reconstruites [13]. Diverses techniques de quantification existent, incluant la quantification uniforme, non-uniforme, vectorielle, optimale et adaptative...

Dans ce qui va suivre, nous allons discuter de la quantification uniforme. C'est en effet la plus adaptée à la compression vidéo [7]. Ce sera par ailleurs le pré-requis pour une bonne compréhension de la suite de ce document. Nous invitons néanmoins le lecteur qui désire étendre ses connaissances sur le sujet à se reporter à différents ouvrages portant sur la compression tels que [13, 4] et [7].

7. La quantification uniforme est optimale pour une source uniforme.

L'idée générale d'une quantification uniforme est d'arrondir un nombre fractionnaire vers une valeur entière, tout en conservant des intervalles de quantification constants. Elle dépend ainsi d'un pas de quantification unique Δ sur l'ensemble des valeurs à quantifier. La relation de quantification s'exprime de la manière suivante :

$$x_q = Q\left(x, \Delta\right). \tag{3.7}$$

Dans cette expression, la fonction de quantification Q peut agir différemment selon que l'on souhaite :

1. arrondir vers l'entier le plus proche : $\mathrm{round}(x, \Delta) = \mathrm{sign}(x) \cdot \lfloor |x| / \Delta + 0.5 \rfloor$,
2. arrondir vers l'entier le plus proche vers plus l'infini : $\mathrm{ceil}(x, \Delta) = \lceil x/\Delta \rceil$,
3. arrondir vers l'entier le plus proche vers 0 : $\mathrm{fix}(x, \Delta) = \mathrm{sign}(x) \cdot \lfloor |x| / \Delta \rfloor$,
4. arrondir vers l'entier le plus proche vers moins l'infini : $\mathrm{floor}(x, \Delta) = \lfloor x/\Delta \rfloor$.

(sign() est la fonction permettant de déterminer le signe de son argument.)

Dans le cas d'une quantification uniforme il est possible de différencier deux types de quantificateurs selon la position de la valeur nulle :

1. le type *midtread* où la valeur nulle est au centre d'un intervalle de quantification : $\Delta = V_{dyn}/(N-1)$ avec N le nombre total d'intervalles de quantification souhaité et V_{dyn} la dynamique du signal à quantifier ;
2. le type *midrise* où la valeur nulle sépare deux intervalles de quantification, ici $\Delta = V_{dyn}/N$.

La figure 3.2 montre un exemple de la caractéristique de ces quantificateurs pour un pas de quantification unitaire. Pareillement, la formule de reconstruction est :

$$\hat{x} = x_q \cdot \Delta. \tag{3.8}$$

D'où une erreur de reconstruction équivalente à (cf. figure 3.3) :

$$e(x, \hat{x}) = x - \hat{x}. \tag{3.9}$$

Il existe également un cas particulier de quantification uniforme. C'est le cas où les intervalles de quantification sont constants exceptés dans un intervalle plus large centré autour de 0. Cette zone morte (*dead zone*) permet aux valeurs de source très petites d'avoir une seule et même valeur quantifiée (habituellement 0). La figure 3.4 en expose le principe. Dans ce cas là, la formule de quantification devient [17] :

$$x_q = \mathrm{sign}\left(x\right) \cdot \left\lfloor \frac{|x| + f}{\Delta} \right\rfloor, \tag{3.10}$$

8. Les fonctions énoncées (*round, ceil, fix, floor*) sont celles fournies dans MATLAB.

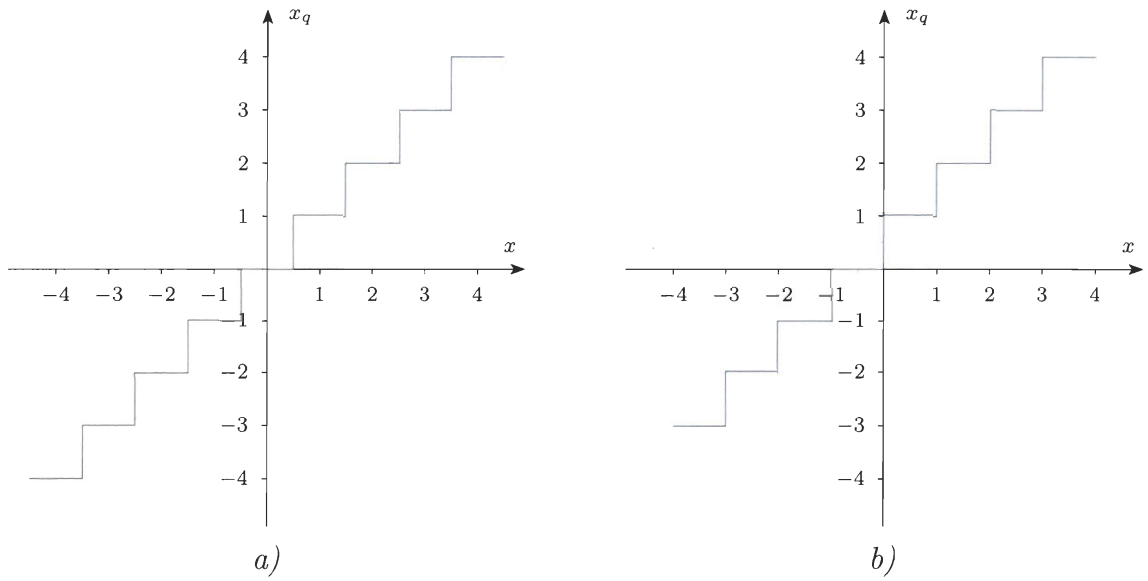


FIGURE 3.2 – Exemples de quantification uniforme avec $\Delta = 1$: *a)* quantificateur de type *midtread* ; *b)* quantificateur de type *midrise*.

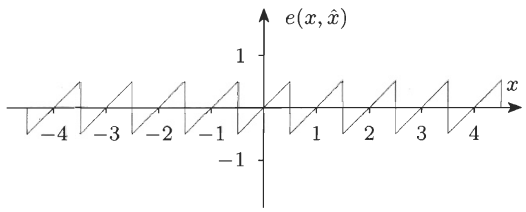


FIGURE 3.3 – Erreur de quantification pour un quantificateur uniforme *midtread* avec $\Delta = 1$.

où $f \in [0, 1/2]$ est un paramètre influençant directement la taille de la zone morte ainsi que les limites des intervalles de quantification. Ce type de quantificateur est souvent employé en compression d'image. En effet, l'application d'une transformation (DCT, ondelettes...) sur une image peut résulter en un nombre important de valeurs non significatives centrées à 0. La quantification de ces valeurs permet d'avoir un grand nombre de 0 et ainsi obtenir par la suite des compressions très efficaces avec le recours de techniques de codage entropique comme le codage par plages, aussi connu sous l'appellation RLE (*Run-Length Encoding*) [18].

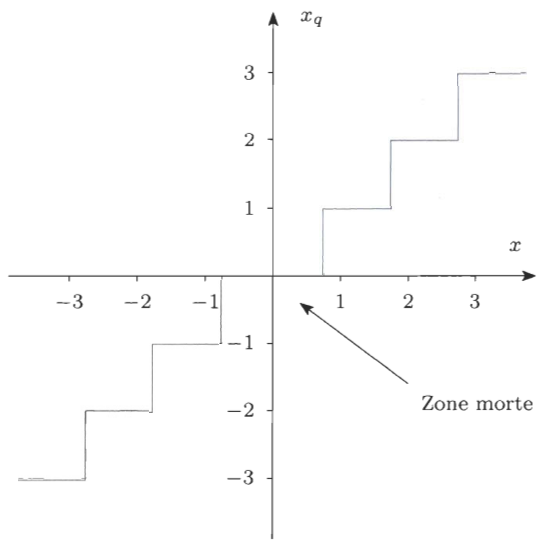


FIGURE 3.4 – Quantificateur uniforme à zone morte avec $\Delta = 1$ et $f = 1/4$.

Dans ce qui suit nous allons exprimer le rapport signal à bruit SNR (*Signal to Noise Ratio*) pour une quantification uniforme. Dans [13], le SNR est définit comme suit :

$$SNR_{dB} \triangleq 10 \log_{10} \frac{\sigma_x^2}{\sigma_e^2} , \tag{3.11}$$

avec σ_x^2 la variance du signal et σ_e^2 la moyenne du carré des erreurs. Pour un quantificateur uniforme, cette dernière s'exprime par la relation suivante :

$$\sigma_e^2 = \sum_{i=0}^{N-1} \int_{d_i}^{d_{i+1}} (x - \hat{x})^2 f_x(x) dx, \tag{3.12}$$

où N est le nombre total d'intervalles de quantification, et d_i et d_{i+1} les limites inférieure et supérieure respectivement de chaque intervalle. De même, si l'on suppose que les valeurs de source sont uniformément distribuées alors nous pouvons écrire que la densité de probabilité $f_x(x)$ vaut $1/N\Delta$. La valeur de σ_e^2 s'écrit dans ce cas [13] :

$$\sigma_e^2 = N \int_{d_0}^{d_1} (x - \hat{x})^2 \frac{1}{N\Delta} dx = \frac{1}{\Delta} \int_{-\Delta/2}^{\Delta/2} x^2 dx = \frac{\Delta}{12}^2 . \tag{3.13}$$

La moyenne du carré de l'erreur est ainsi directement proportionnelle au carré du pas de quantification. Il est d'ailleurs intéressant de remarquer que lorsque le pas de quantification est divisé par 2, la moyenne du carré de l'erreur est divisée par 4.

D'après la variance d'une variable aléatoire suivant une distribution uniforme de support borné[9], nous pouvons écrire [13] :

$$\sigma_x^2 = \frac{(N\Delta)^2}{12}. \tag{3.14}$$

Le rapport signal à bruit s'écrit alors :

$$SNR_{dB} = 10\log_{10}\frac{\sigma_x^2}{\sigma_e^2} = 10\log_{10} N^2. \tag{3.15}$$

Remarque : Si nous supposons que les intervalles de quantification sont codés à l'aide d'un code binaire naturel, nous pouvons écrire que N équivaut à 2^b où b est le nombre de bits utilisés pour la représentation binaire. Le rapport signal à bruit s'écrit dans ce cas :

$$SNR_{dB} = 20\log_{10} 2^b \approx 6.02b \text{ dB}. \tag{3.16}$$

Un bit supplémentaire permet ainsi de doubler le nombre d'intervalles de quantification et d'apporter un gain d'environ 6.02 dB sur le rapport signal à bruit.

En codage vidéo, il est plus courant d'utiliser le PSNR[10] plutôt que le SNR. Le PSNR est en quelque sorte un indicateur objectif de la distorsion présente entre deux images de taille $m \times n$ (une décodée I_D et une servant de référence I_R). Il s'exprime en dB et une valeur élevée signifie que peu de distorsion est présente dans l'image. Toutefois, une valeur élevée ne veut pas forcément signifier un bon rendu de l'image, la meilleure manière de percevoir la qualité restant subjective. Néanmoins, une valeur de 36 dB sera considérée comme une très bonne qualité d'image.

$$PSNR = 10\log_{10}\frac{MAX^2}{\frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n}\left(I_D(i,j) - I_R(i,j)\right)^2}. \tag{3.17}$$

Le dénominateur représente l'écart quadratique moyen EQM et MAX est la valeur maximale qu'un pixel peut avoir. Par exemple, si un pixel est codé sur 8 bits (non signé) sa valeur maximale possible sera alors 255. À titre de référence, un gain de même 0.1 dB sur le PSNR sera considéré comme significatif sur l'amélioration de la qualité.

9. $X \sim U(a,b) : E\left[(X - E[X])^2\right] = \frac{1}{b-a}\int_a^b \left(x - \frac{a+b}{2}\right)^2 dx = \frac{(b-a)^2}{12}$.

10. PSNR : *Peak Signal-to-Noise Ratio*

3.2.5 La norme H.264/AVC

H.264, ou MPEG-4 AVC, est une norme de codage vidéo issue du travail commun de normalisation de l'UIT-T *Video Coding Experts Group* (VCEG) ainsi que l'ISO/CEI *Moving Picture Experts Group* (MPEG) [19]. Elle est de plus le produit d'un effort de partenariat connu sous le nom *Joint Video Team* (JVT). Actuellement, la norme est devenue partie intégrante de la nouvelle génération de supports vidéo haute définition (HD-DVD, Blu-Ray) ainsi que d'autres applications comme la télévision sur téléphone portable. La norme (cf. [20]), publiée en 2003, définit cinq profils différents et globalement 15 niveaux répartis selon ces profils [14]. Chaque profil correspond à un sous-ensemble de syntaxe utilisé pour la représentation des données codées. Ceci permet de faire varier la complexité du codec selon les besoins d'une application particulière. Pour les mêmes raisons, les différents niveaux constituant chaque profil limitent les options à diverses valeurs de paramètres. Les profils et les niveaux sont ordonnés selon la qualité souhaitée. Pour indication, le niveau 1 du profil 1 est approprié pour l'encodage vidéo jusqu'à 64 kbps. De même, le niveau 5.1 (profil 5) permet des applications allant jusque 240 Mbps.

La norme de codage permet d'atteindre une bien meilleure compression vidéo que toutes les normes précédentes avec un débit deux ou trois fois moindre pour la même qualité au décodage [14]. Cependant cela se traduit par une complexité accrue à l'encodage. Par rapport aux normes antérieures, elle présente quelques innovations *majeures* [14] :

- La compensation de mouvement peut être effectuée avec plusieurs images de référence déjà codées. Il est ainsi possible d'avoir jusqu'à 32 images utilisées comme référence et jusqu'à 4 références différentes pour un même *macrobloc* [21]. Cette fonctionnalité permet une réduction significative du débit pour des vidéos comportant des scènes très rapides.

- La norme H.264 reconnaît qu'un *macrobloc* peut-être similaire à un autre dans la même image. De ce fait, l'estimation et la compensation de mouvement sont étendues à l'intra-codage (intraprédiction), recherchant ainsi les similitudes internes à l'image. Bien entendu, cette recherche est limitée à des portions de l'image restant disponibles avant le décodage des macroblocs.

- Les macroblocs peuvent être de forme rectangulaire (les formes permises sont ainsi 16×8, 8×16, 8×8). Dans le profil 1, les macroblocs sont découpés selon des blocs 4×4 pour la luminance et 2×2 pour les chrominances ; de plus larges blocs 8×8 sont permis dans les profils supérieurs. Ce degré de liberté supplémentaire permet une meilleure estimation de mouvement, spécialement sur des régions hautement détaillées où la correspondance avec des macroblocs fonctionne mal.

• La H.264 inclut une version modifiée de la DCT avec des valeurs entières dans la matrice de transformation **H** (cf. § 3.2.3). En fait, les valeurs de **H** sont légèrement différentes si la transformation est appliquée sur des intra-blocs, des blocs d'erreur résiduelle ou bien des blocs de luminance et de chrominance. Les valeurs de la matrice **H** sont choisies de telle sorte que les transformations directe et inverse s'opèrent seulement par des décalages de bits ou des additions, permettant ainsi de réduire la complexité calculatoire.

• À la place d'un codage à longueur variable tels que le codage de Huffman ou le codage arithmétique conventionnel, la norme H.264 introduit deux autres techniques pour encoder l'erreur résiduelle transformée, les vecteurs de mouvement, etc., soient :

1. le codage arithmétique contextuel (CABAC : *Context-adaptive binary arithmetic coding*), une technique sophistiquée pour compresser sans perte des éléments syntaxiques dans le flux vidéo ;
2. le codage Golomb exponentiel est utilisé afin d'encoder des valeurs de paramètres uniques tandis qu'un codage adaptatif de type Huffman (CAVLC : *Context-adaptive Huffman variable-length coding*) est utilisé comme amélioration du codage à longueur variable traditionnel. Bien que moins complexe que CABAC, CAVLC est plus élaboré et plus efficace que les méthodes habituellement utilisées jusqu'à présent pour coder les coefficients issus de la transformation.

• La norme H.264 a également bénéficié d'une amélioration significative du rapport débit-distortion en partie par la présence d'un filtre adaptatif réducteur d'artéfacts [19]. Ce filtre fonctionne selon de simples opérations détectant et analysant les discontinuités sur les limites des blocs codés. Les artéfacts sont par la suite atténués par application de filtres spécifiques [3]. La figure 3.5 donne une représentation graphique d'une discontinuité à la jonction de deux blocs 4×4. C'est un exemple typique où le filtre doit s'appliquer.

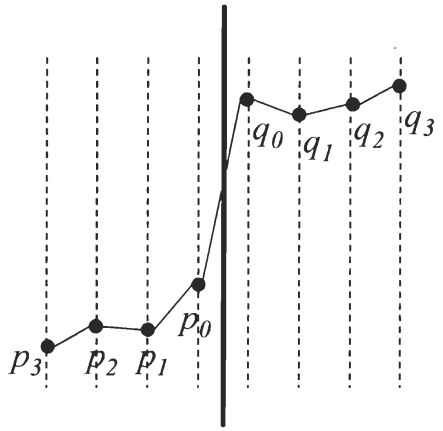


FIGURE 3.5 – Profil typique d'application du filtre [3].

L'idée générale est d'effectuer ici un filtrage passe-bas permettant de *lisser* les profils trop abruptes. Cependant le filtre ne s'applique pas partout. Il est nécessaire qu'il sache faire la différence entre l'information des détails (donc de hautes fréquences) et les discontinuités issues de l'effet de blocs. Pour faire cette part des choses, la norme utilise un indicateur *Bs* (*Boundary-Strength*) caractérisant chaque limite de bloc [3]. Sans trop entrer dans les détails, cet indicateur permet de caractériser de quelle manière le filtre doit s'appliquer en bordure du bloc. Des critères calculés ensuite d'après les valeurs des pixels p_0, p_1, p_2 ainsi que q_0, q_1, q_2 (voir figure 3.5) permettent d'appliquer certaines fonctions de filtrage et modifier par la même occasion les valeurs de ces pixels.

À titre d'information, les schémas de principe de l'encodeur ainsi que du décodeur de la norme H.264 sont représentés par la figure 3.6 et la figure 3.7 respectivement.À l'exception du filtre réducteur d'artéfact, la majeure partie des éléments fonctionnels (transformation, quantification, estimation et compensation de mouvement,...) du codec se retrouve dans les normes de codage vidéo antérieures (MPEG-1, MPEG-2, MPEG-4, H.261 et H.263). Les changements importants se trouvent plutôt dans les détails de chaque élément [4]. Par ailleurs, le lecteur désireux d'obtenir plus d'information relativement à cette nouvelle norme de codage vidéo, peut consulter l'ouvrage de Iain E. Richardson [4].

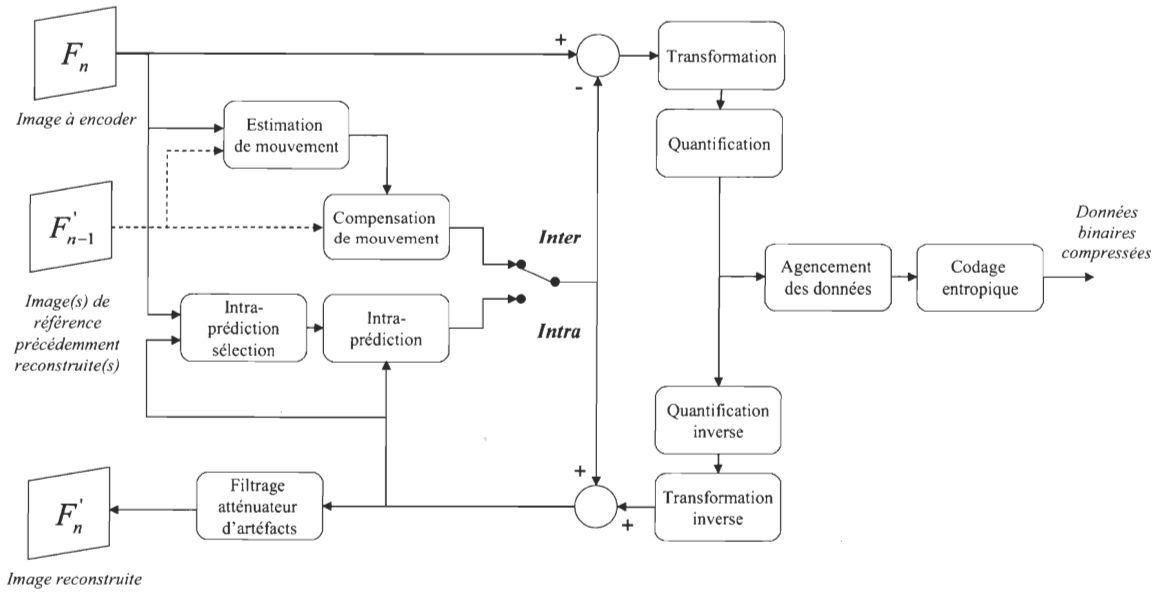


FIGURE 3.6 – H.264 : schéma de principe de l'encodeur [4].

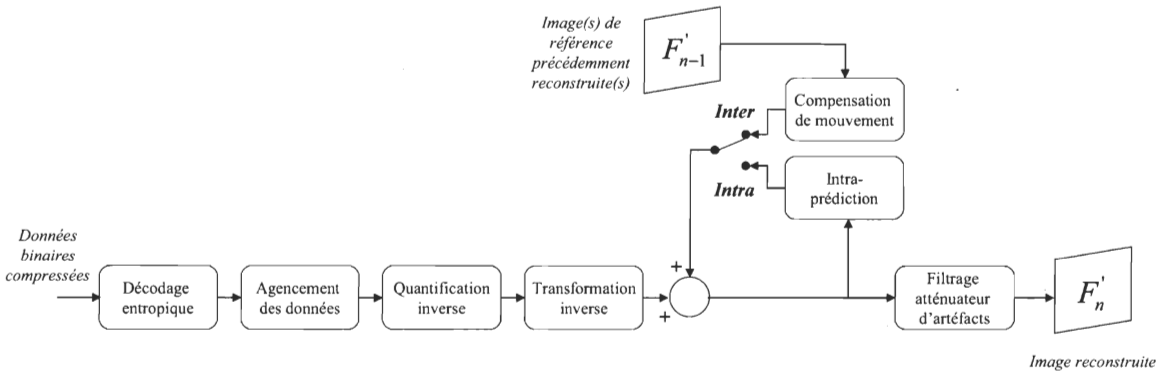


FIGURE 3.7 – H.264 : schéma de principe du décodeur [4].

3.3 Le codage vidéo distribué

3.3.1 Principes élémentaires

Le codage vidéo distribué est une nouvelle façon de voir la compression vidéo. C'est une technique récente dont l'essor se situe après l'an 2000. Son principe permet un encodage très simple et indépendant d'images vidéo tout en nécessitant un décodage complexe de celles-ci, inversant de cette façon l'asymétrie de complexité propre aux codecs vidéo conventionnels. Pour donner un ordre d'idée, l'encodage distribué est à la fois moins *couteux* en calculs, moins *couteux* en temps et requiert moins de mémoire qu'un encodage classique reposant sur l'estimation et la compensation de mouvement.

Dans les systèmes actuels de codage vidéo distribué, il est possible de répertorier deux types d'images : (i) les images clés (K) et (ii) les images Wyner-Ziv (WZ). Les images K sont encodées/décodées par l'intermédiaire d'un codec conventionnel d'intra-images comme celui rencontré dans H.264/AVC. En outre, les images WZ sont encodées/décodées via un codec de Wyner-Ziv. Leur décodage requiert l'utilisation de l'information latérale propre à chacune d'entre elles. Cette information latérale consiste, pour chaque image WZ, en une image générée au décodeur par un mécanisme d'interpolation où d'extrapolation de mouvement utilisant les images K ou/et WZ précédemment décodées. C'est une estimation de l'image WZ encodée que le décodage de Wyner-Ziv va *corriger* avec un certain degré de distorsion. Pour rappel, le principe de l'encodeur/décodeur de Wyner-Ziv est présenté en figure 3.8.

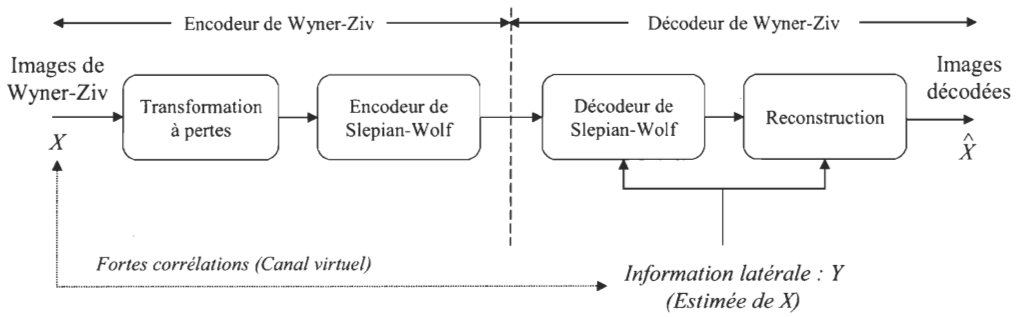


FIGURE 3.8 – Principe du codeur/décodeur de Wyner-Ziv.

Nous savons qu'un encodeur de Wyner-Ziv consiste en une transformation à perte d'information suivie d'un encodeur de Slepian-Wolf. La transformation à perte appliquée sur une image WZ (représentée par la source d'information X) est habituellement la transformation en cosinus discret couplée avec un processus de quantification des coefficients issus de la DCT. L'encodeur de Slepian-Wolf permet quant à lui la compaction (compression sans pertes) des bits issus du processus de quantification sous la condition que certains critères soient respectés [1].

Au niveau du décodeur de Wyner-Ziv, l'information latérale Y, une estimée de X, peut être également considérée comme une version bruitée de X. Les bits transmis au décodeur, permettent alors, d'un certain point de vue, de corriger Y. De cette manière, un canal virtuel entre X et Y existe. Il est habituellement modélisé par un canal binaire symétrique (BSC : *Binary Symmetric Channel*) ayant une probabilité d'entrecroisement ϵ. La figure 3.9 en est une représentation. Finalement, la version corrigée de Y correspond à l'image reconstruite $\hat{X}$.

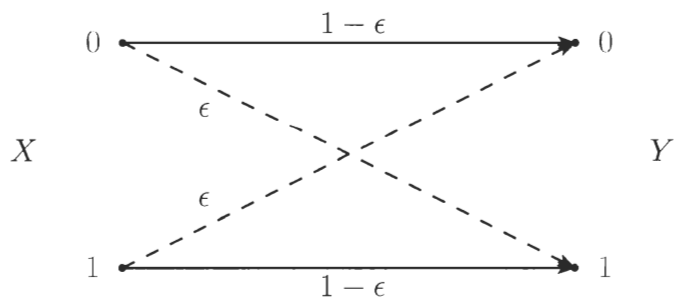


FIGURE 3.9 – Canal binaire symétrique avec ses probabilités de transition $\Pr(x|\,y)$.

La corrélation qui existe intrinsèquement entre X et Y est un paramètre important. Elle joue un rôle majeur dans les performances de compression du système. Une forte corrélation signifie en général que peu de bits seront nécessaires à la correction de Y. Pour cette raison, les performances des systèmes de codage vidéo distribué dépendent de beaucoup du contenu de l'information à coder. La transformée en cosinus discret, la transformée en ondelettes, les codes Gray, les algorithmes à haute performance de génération de l'information latérale, etc..., sont autant de moyens permettant d'augmenter la corrélation entre l'image source à coder et l'information latérale.

Le codec de Slepian-Wolf repose habituellement sur l'utilisation de codes correcteurs d'erreurs très efficaces. Ces codes permettent de corriger efficacement l'information latérale Y tout en étant d'un certain point de vue une manière de compresser l'image source X (l'image WZ). Ici, la compression est rendue possible justement par la présence d'un canal virtuel entre X et Y. Parmi les bits de parité (où des bits de syndromes [11]) formés par ces codes de canal, seulement un sous-ensemble sera transmis au décodeur. C'est uniquement de cette façon qu'une compression des données peut être atteinte.

11. Dans le cas de l'utilisation de codes en bloc comme les codes LDPC, il est usuel de transmettre des bits de syndrome plutôt que des bits de parité. Par définition, en codage de canal, le syndrome est le vecteur de bits issu de la multiplication du message reçu via le canal de transmission (mot de code plus motif d'erreurs) avec la matrice de contrôle de parité. Un syndrome nul signifie alors qu'il n'y a aucune erreur de transmission. Pour ce qui est de l'utilisation en codage vidéo distribué, le syndrome est obtenu directement par multiplication du message d'information avec la matrice de contrôle de parité (cf. Annexe B), engendrant ainsi des syndromes différents du vecteur nul.

Aujourd'hui, les systèmes de codage vidéo distribué les plus avancés reposent sur l'utilisation des turbo-codes et des codes LDPC. Ce sont les meilleurs candidats en vue d'approcher la limite de Slepian-Wolf [1]. L'utilisation de ces codes est également la principale cause d'une forte complexité au décodeur. C'est également pour cette raison que les systèmes de codage vidéo distribué diffèrent des codeurs vidéo conventionnels. En effet, dans ces derniers, la forte complexité est associée au processus d'estimation de mouvement situé dans l'encodeur. Celui-ci peut être jusqu'à 10 fois plus complexe que le décodeur. En codage vidéo distribué, l'asymétrie de complexité est inversée.

Un processus de contrôle de débit est effectué directement au décodeur par l'intermédiaire d'un canal de retour vers l'encodeur. Habituellement, lorsqu'un message n'est pas correctement décodé, un bit émis du décodeur vers l'encodeur permet la demande d'un nombre plus important de bits de parité. Cette procédure est répétée jusqu'à temps que le message soit décodé correctement. Naturellement, pour chaque message à décoder, l'encodeur commencera par envoyer un nombre minimal de bits de parité et augmentera progressivement celui-ci à chaque nouvelle requête du décodeur. Ce canal de retour ne devrait pas exister en théorie mais il est indispensable pour des raisons pratiques, liées entre autre à l'utilisation des codes correcteurs d'erreurs. En effet, il serait difficile sans son recourt de transmettre directement le nombre adéquate de bits de syndrome (ou de parité) pour un décodage correct. Pareillement la présence d'un tel canal rend impossible une séparation dans le temps de l'encodage et du décodage, restreignant par la même occasion l'éventail d'applications. Il fait actuellement l'objet de nombreuses études en vue de sa suppression.

3.3.2 Étude d'un système de référence

A titre d'exemple, nous allons présenter ici un système de codage vidéo distribué présent dans la littérature. Celui-ci fut initialement proposé par C. Brites, J. Ascenso et F. Pereira dans [5] et nous servira de référence pour la suite. C'est un système semblable à celui de Aaron et *al.* dans [22] mais dont certains composants ont évolué. Son fonctionnement repose sur l'utilisation de la DCT entière de H.264 et d'une quantification uniforme. La figure 3.10 en fournit une illustration. Nous allons maintenant en décrire quelques éléments essentiels.

Encodage

Les images K, $X_{2i\pm1}$, impaires, sont intra-codées via un codeur conventionnel d'intra images tandis que les images WZ, X_{2i}, paires, sont codées via le codeur de Wyner-Ziv. Chaque bloc 4×4 des images WZ est ensuite transformé à l'aide de la DCT entière propre à H.264 [4] :

$$X_{2i}^{DCT} = (\mathbf{C}X_{2i}\mathbf{C}^T) \otimes \mathbf{E}_f, \tag{3.18}$$

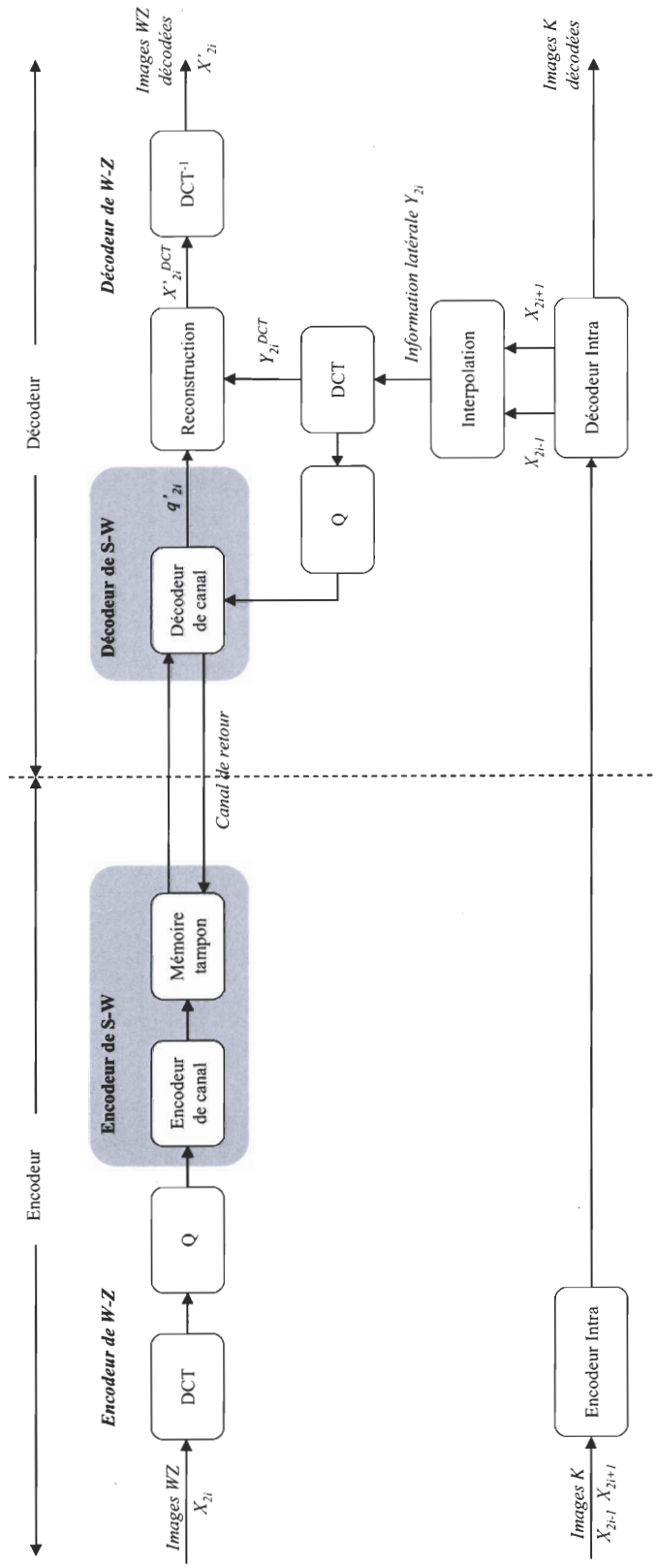


FIGURE 3.10 – Système de codage vidéo distribué basé sur la transformation DCT.

avec

$$\mathbf{C} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix}, \tag{3.19}$$

et $\mathbf{E}_f$ une matrice de facteurs d'échelle définie comme :

$$\mathbf{E}_f = \begin{bmatrix} a^2 & \frac{ab}{2} & a^2 & \frac{ab}{2} \\ \frac{ab}{2} & \frac{b^2}{4} & \frac{ab}{2} & \frac{b^2}{4} \\ a^2 & \frac{ab}{2} & a^2 & \frac{ab}{2} \\ \frac{ab}{2} & \frac{b^2}{4} & \frac{ab}{2} & \frac{b^2}{4} \end{bmatrix}. \tag{3.20}$$

où $a = 1/2$ et $b = \sqrt{2/5}$. L'opérateur $\otimes$ dans (3.18) représente le produit terme à terme de deux matrices.

Les coefficients issus de cette transformation, X_{2i}^{DCT}, se répartissent suivant 16 bandes de fréquence ($k = 1, \cdots, 16$). Ensuite selon le type de coefficient de transformée, deux types de quantification s'opèrent afin de produire les symboles q_{2i}. Le coefficient DC ($k = 1$) dont les valeurs sont comprises entre 0 et 1020 subissent une quantification uniforme de type *midtread*. Le pas de quantification s'écrit alors :

$$\Delta_{DC} = 1024/N_k, \tag{3.21}$$

avec N_k correspondant au nombre d'intervalles de quantification (généralement sous forme de puissance de 2). La valeur 1024 est utilisée puisque les coefficients DC se représentent sur 10 bits. Pour les coefficients AC ($k \neq 1$) en revanche, une quantification uniforme *midrise* s'applique. Néanmoins, le pas de quantification Δ_{AC} dépendra ici de la valeur maximale absolue du coefficient de la bande k dans l'image :

$$\Delta_{AC} = 2\,|V_k|\,/(N_k - 1). \tag{3.22}$$

Ceci permet de faire varier le pas de quantification pour chaque image et ainsi d'obtenir une meilleure qualité d'image à la reconstruction. Ainsi, lors de cette reconstruction, il est indispensable que le décodeur connaisse les valeurs maximales absolues de chaque bande de fréquence. Pour cette raison, l'encodeur doit transmettre ces valeurs au décodeur. Dans [5], cette transmission correspond à un maximum de 240 bits par images : 15 bandes de fréquence à raison de 16 bits par bande.

Des performances différentes s'obtiennent en changeant la valeur de N_k. La figure 3.11 montre huit matrices 4×4 de quantification distinctes. Chaque valeur à l'intérieur de l'une d'entre elles correspond au nombre d'intervalle de quantification N_k associé à la bande k. La valeur 0 signifie simplement qu'aucune information relative à la bande de fréquence correspondante n'est transmise.

Les coefficients quantifiés q_{2i} sont ensuite transmis à l'encodeur de Slepian-Wolf. De là, ils sont séparés en plan de bits puis transmis à l'encodeur de canal. Ce dernier utilise

16	8	0	0
8	0	0	0
0	0	0	0
0	0	0	0

M1

32	8	0	0
8	0	0	0
0	0	0	0
0	0	0	0

M2

32	8	4	0
8	4	0	0
4	0	0	0
0	0	0	0

M3

32	16	8	4
16	8	4	0
8	4	0	0
4	0	0	0

M4

32	16	8	4
16	8	4	4
8	4	4	0
4	4	0	0

M5

64	16	8	8
16	8	8	4
8	8	4	4
8	4	4	0

M6

64	32	16	8
32	16	8	4
16	8	4	4
8	4	4	0

M7

128	64	32	16
64	32	16	8
32	16	8	4
16	8	4	0

M8

FIGURE 3.11 – Matrices de quantifications.

des codes correcteurs d'erreurs très performants afin de compresser l'information. La séparation en plan de bits permet d'obtenir de meilleures performances de compression en pratique. Dans [5], les auteurs utilisent les turbo-codes [23]. Le principe général est d'utiliser des codes convolutifs récursifs systématiques et de ne transmettre que les bits de parité au décodeur. Un poinçonnage appliqué sur ces derniers permet d'atteindre différents débits. Le décodeur devra alors corriger non seulement les bits poinçonnés mais aussi les bits de l'information latérale (transformée et quantifiée) différents des bits de l'image source. Dans notre étude, notre choix s'est porté sur l'utilisation des codes LDPC plus performants.

Décodage

Deux images K décodées, X_{2i-1} et X_{2i+1}, permettent de produire l'information latérale Y_{2i} par un procédé d'interpolation. Le décodeur de Slepian-Wolf permet alors de corriger l'information latérale transformée et quantifiée à l'aide des bits de parité reçus de l'encodeur. Les coefficients quantifiés q'_{2i} sont ainsi formés. Ils correspondent aux coefficients q_{2i} de l'image source mais peuvent contenir une faible proportion de différences (cf. § 2.3.1). Un processus de reconstruction permet dès lors de reformer les coefficients de la DCT à partir de q'_{2i} et des coefficients de transformée Y_{2i}^{DCT}. Cette reconstruction se déroule ainsi [24] :

$$X_{2i}'^{DCT} = \begin{cases} I_{min} & \text{, si } Y_{2i}^{DCT} < I_{min}, \\ Y_{2i}^{DCT} & \text{, si } I_{min} \leq Y_{2i}^{DCT} < I_{max}, \\ I_{max} & \text{, si } Y_{2i}^{DCT} \geq I_{max}, \end{cases} \tag{3.23}$$

avec I_{min} et I_{max} les bornes de l'intervalle de quantification reliées au coefficient q'_{2i}. Ces bornes dépendent du type de coefficient. Ainsi pour les coefficients DC, elles se

calculent comme suit :

$$I_{min} = q'_{2i} \cdot \Delta_{DC}, \tag{3.24}$$
$$I_{max} = (q'_{2i} + 1) \cdot \Delta_{DC}. \tag{3.25}$$

Pour les coefficients AC en revanche, elles se définissent comme :

$$I_{min} = (q'_{2i} - 0.5) \cdot \Delta_{AC}, \tag{3.26}$$
$$I_{max} = (q'_{2i} + 0.5) \cdot \Delta_{AC}. \tag{3.27}$$

Finalement, une fois que les coefficients X'^{DCT}_{2i} ont été reconstruits, la transformation inverse DCT de H.264 s'applique afin calculer l'image reconstruite X'_{2i} [4] :

$$X'_{2i} = \mathbf{C}_{inv} \left(X'^{DCT}_{2i} \otimes \mathbf{E}_{inv} \right) \mathbf{C}^T_{inv}. \tag{3.28}$$

Ici $\mathbf{C}_{inv}$ est la matrice inverse de transformation :

$$\mathbf{C}_{inv} = \begin{bmatrix} 1 & 1 & 1 & 1/2 \\ 1 & 1/2 & -1 & -1 \\ 1 & -1/2 & -1 & 1 \\ 1 & -1 & 1 & -1/2 \end{bmatrix}. \tag{3.29}$$

$\mathbf{E}_{inv}$ est une matrice de facteurs d'échelle pour les coefficients de transformée. Elle s'écrit comme :

$$\mathbf{E}_{inv} = \begin{bmatrix} a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \\ a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \end{bmatrix}. \tag{3.30}$$

Globalement, les performances résultantes de ce système surpassent celles de celui proposé par Aaron et *al.* dans [22]. La raison majeure en est qu'elle provient de l'utilisation d'une quantification adaptative pour les coefficients AC. Ici la transformation entière de H.264 n'influe que très peu sur les résultats. En revanche, elle permet de réduire la complexité de l'encodeur.

3.3.3 Revue de littérature

Avant 2002, les recherches étaient axées principalement sur le codage de source distribuée. Prahdan et Ramchandran proposèrent l'idée de transmettre le syndrome d'un ensemble de mots de code pour effectuer une compression [25]. Depuis des concepts similaires ont été étendus à des codes plus avancés. Garcia-Frias et Zhao [26, 27], Bajcsy et Mitran [28, 29] ainsi que Aaron et *al.* dans [30] ont montré que les turbo-codes permettaient d'obtenir des performances plus proches de la borne théorique de Slepian-Wolf. Liveris et *al.* montrèrent quant à eux que les codes à faible densité de parité (LDPC) sont plus adaptés et plus performants que les turbo-codes [11].

Dans [6], publié en 2005, Varodayan et *al.* exposent une méthode *clé* pour l'utilisation des codes LDPC dans le codage vidéo distribué. Les auteurs proposèrent en effet l'idée de coupler un code LDPC avec un accumulateur. Ces codes LDPCA, de leur appellation, permettent dès lors d'éviter toute retransmission des bits de syndromes déjà transmis lors d'un décodage incorrect et procure une adaptation plus efficace des débits de transmission. Des explications plus détaillées sont fournis en annexe B. Il est également possible d'y voir une certaine analogie avec les codes fontaines [31] tels que les codes Raptor [32].

En 2002, le groupe d'Aaron fut le premier à appliquer les concepts de compression distribuée à la vidéo [24]. Depuis plusieurs améliorations et techniques furent développées. Elles possèdent le même objectif : améliorer la corrélation existante entre la source et l'information latérale en vue d'atteindre des débits binaires plus faibles. Dans [33, 34, 22], le groupe s'oriente sur l'utilisation d'une transformation en cosinus discret afin d'exploiter les redondances spatiales de l'image. D'autres groupes de recherches commencent à l'utiliser également [35, 5]. Dès 2005, certains chercheurs utilisent la transformée en ondelettes [36, 37, 38]. Cette transformée également très utilisée en compression d'image procure à la fois une distribution spatiale et fréquentielle de l'énergie. Utilisée pour le codage vidéo distribué, elle engendre cependant une complexité accrue à l'encodage.

D'autres travaux de recherche s'orientent aussi sur de nouvelles méthodes de construction de l'information latérale par diverses techniques de compensation de mouvement [39, 40, 41, 42, 43]. Ces méthodes permettent un gain significatif du rapport débit-distorsion sans pour autant augmenter la complexité d'encodage. Ici encore l'objectif est l'obtention d'une information latérale de très haute qualité afin d'améliorer la corrélation existante avec l'image source.

Dans [44], He et *al.* montrèrent qu'il est possible d'utiliser les codes Gray afin d'augmenter la corrélation entre l'image source et l'information latérale. L'idée veut que deux valeurs consécutives sous forme binaire possèdent une distance de Hamming unitaire. C'est un article très important dont les principes seront repris plus tard au chapitre 6 de cette thèse. Des détails sont également fournis en annexe C.

L'amélioration du modèle de corrélation entre la source et l'information latérale est aussi d'actualité [45, 46, 47]. Les différences entre l'image source et l'information latérale suivent en effet une distribution de Laplace. De ce fait, une bonne estimation de ce modèle procure plusieurs améliorations : (i) au décodeur, il permet d'initialiser la valeur de l'information intrinsèque nécessaire au bon décodage des messages reçus de l'encodeur ; et (ii) à l'encodeur, sa connaissance permet de réduire le débit binaire en réduisant l'utilisation de la voie de retour. Cette voie de retour est en effet rencontrée dans la plupart des architectures de codage distribué. Elle permet de contrôler directement le débit binaire de l'encodeur en minimisant le nombre de bits de parité (ou de syndrome) à transmettre. Par ailleurs, c'est dans l'utilisation de cette voie de re-

tour que réside la faiblesse du codage distribué puisqu'elle rend la réalisation pratique de certaines architectures de compression distribuée très difficile. La suppression de cette voie de retour est une direction prometteuse qui commence à peine a être étudié [48, 49, 50, 51, 52].

Pareillement dans [53], Kubasov et *al.* proposent un procédé de reconstruction optimal au décodeur. Celui-ci repose sur la minimisation de l'écart-quadratique moyen. Ici encore une bonne estimation du paramètre de la distribution Laplacienne est requise au décodeur.

En 2007, dans l'article [54], les auteurs étudient l'influence du GOP. Il y est montré que les systèmes utilisant un GOP constitués de deux images uniquement, soient une image K et une image WZ, sont les plus performants. De même, cette étude porte sur un comparatif entre les codes LDPC et les turbo-codes. Il y est expliqué que les codes LDPC sont toujours les plus performants dans ce type d'application.

Un dispositif intéressant aussi à prendre en compte ici est celui proposé par Chien et *al.* dans [55]. Ce dispositif permet de ne corriger que les blocs présentant des erreurs au décodage. Le décodeur doit alors transmettre à l'encodeur les indices des blocs à corriger. Dans cette thèse, nous proposons au chapitre 5 un schéma de codage assez semblable dans le sens où seule l'information des blocs les plus significatifs est transmise au décodeur. Son fonctionnement reste néanmoins très différent.

Chapitre 4

Filtrage vidéo par arbres de décision

La plupart des normes de codage vidéo sont basées sur l'utilisation de techniques de compensation de mouvement ainsi que de la transformation DCT agissant sur des blocs de pixels. Pour de forts niveaux de quantification, la compensation de mouvement et la transformation produisent des artefacts visuels dans la séquence vidéo décodée. C'est une forme de distorsion à laquelle le système visuel humain est très sensible. La dernière norme de codage vidéo, H.264/AVC, introduit un filtrage réducteur d'artefacts. Malgré cela, de la distorsion reste visible après l'application de ce dernier. Dans ce chapitre, nous proposons une méthode de filtrage non-conventionnelle pour réduire la distorsion résiduelle et ainsi améliorer la qualité des images décodées. Cette technique de filtrage repose sur l'utilisation d'arbres de décision. Ces derniers sont utilisés afin de classifier les entrées du filtre et de sélectionner les meilleurs coefficients de filtrages pour ces entrées. Les résultats expérimentaux avec des blocs 4×4 *ont indiqué que l'utilisation d'un tel filtre permet d'améliorer la qualité des séquences vidéo encodées par la norme H.264/AVC.*

4.1 Avant-propos

4.1.1 Rappels sur la norme H.264/AVC

Le norme H.264/AVC [20], est la dernière et la plus efficace des normes de compression vidéo. Elle est une évolution directe des précédentes normes H.261, H.262 et H.263. Elle permet d'atteindre la même qualité que la norme MPEG-2 tout en ayant un débit binaire deux fois moindre. Actuellement, elle est devenue une composante principale des DVD haute définition et d'autres applications telles que la télévision sur téléphone portable. Une de ses principales caractéristiques est qu'elle repose sur une transformation DCT entière, opérant sur des blocs de pixels 4×4. Cette transformation peut être calculée uniquement avec une arithmétique entière [56] et mène en conséquence à une réduction de complexité tout en conservant un impact minime sur la qualité d'image. Les coefficients de transformée sont quantifiés et codés pour atteindre une compression.

Ici, la quantification est une opération à pertes où la quantité d'information perdue dépend du paramètre de quantification QP, lequel est relié à la largeur de l'intervalle de quantification.

Le codage vidéo basé sur des blocs de pixels introduit de la discontinuité aux limites de ceux-ci : ce sont les artefacts. Pour circonvenir à cela, la norme H.264/AVC introduit un filtrage adaptatif réducteur d'artefacts s'adaptant au type de discontinuité [3]. La *force* du filtre dépend du paramètre de quantification QP dans le sens où une forte quantification augmentera la force de filtrage. Tout d'abord, le filtre détecte les discontinuités présentes aux limites des blocs puis détermine la force de filtrage appropriée en fonction du paramètre QP, du type de discontinuité ainsi que du mode de codage des blocs voisins (macroblocs ou non) [4]. Un paramètre, Bs (*Boundary-Strength*) permet de choisir la force du filtre. Il comporte 5 valeurs distinctes de 0 à 4. Par exemple, une valeur de 4 signifie que la force de filtrage la plus élevée sera utilisé tandis qu'une valeur nulle signifie qu'aucune opération de filtrage ne sera effectuée. Cependant, même après ce type de filtrage, une distorsion visible reste en comparaison avec la séquence vidéo originale.

4.1.2 Les méthodes d'apprentissage

Le filtre proposé se fonde sur des techniques d'apprentissage. Le concept de méthodes d'apprentissage est relié à celui des techniques d'extraction de connaissance à partir de données : aussi connu sous l'appellation *data mining*. Celui-ci est un des maillons de la chaîne de traitement pour la découverte des connaissances à partir de données [57]. Il possède une approche très différente de la méthode statistique. Cette dernière exige en effet qu'on se fixe une hypothèse, que les données vont confirmer ou non. Le data mining fait quant à lui émerger des hypothèses à partir de données brutes.

Auparavant, les méthodes d'apprentissage n'ont jamais été utilisées dans des applications de filtrage vidéo. Notre motivation était donc de créer une nouvelle méthode de filtrage en exploitant les principes des méthodes d'apprentissage. Les algorithmes d'apprentissage aussi appelés algorithmes de classification (*classifiers* en anglais) sont utilisés ici dans le but d'analyser le comportement sous-jacent du codage par bloc, duquel les meilleures règles de filtrage seront générées afin de réduire la distorsion. Spécifiquement, la relation existante entre la distorsion d'un pixel et la valeur des pixels adjacents est analysée et caractérisée par des algorithmes de classification. Un filtrage adaptatif permettant de réduire la distorsion est ainsi obtenue de cette relation. Les algorithmes de classification les plus courants regroupent les arbres de décision, les réseaux de neurones et les algorithmes génétiques.

Pour notre étude, nous avons utilisé le logiciel Weka[1] qui regroupe en fait toute une

1. Weka : *Waikato Environment for Knowledge Analysis*. Logiciel développé par I. H. Witten et E. Frank de l'Université de Waikato en Nouvelle Zélande (http ://www.cs.waikato.ac.nz/ml/weka/).

collection d'algorithmes de classification. C'est un logiciel libre de Licence Publique Générale GNU[2] (GNU *General Public License*) qui fut un outil de choix nous ayant permis de sélectionner la méthode de classification la plus efficace pour notre application.

4.2 Principe du filtre proposé

Le filtre proposé est un filtre adaptatif unidimensionnel prenant en compte quatre pixels adjacents de manière transverse à la limite d'un bloc. La figure 4.1 montre un exemple de quatre pixels voisins horizontaux pris au travers de la délimitation verticale d'un bloc. Soient p_1, p_2, p_3 et p_4 les quatre pixels adjacents de l'image décodée et soit $\tilde{p}_i$ la sortie du filtre pour le pixel p_i ($1 \leq i \leq 4$). Le filtre consiste, pour chaque pixel p_i, en un arbre de décision (soit un ensemble de règles) et d'un nombre de groupes de coefficients de filtrages (soient les modèles de prédiction associés à chaque règle).

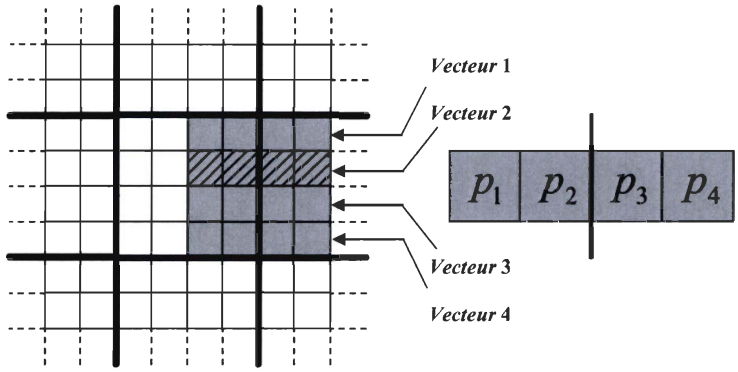


FIGURE 4.1 – Sélection des attributs horizontaux en vue de la création des exemples d'entraînements.

La figure 4.2 illustre le schéma bloc du filtre. L'arbre de décision est utilisé pour classifier chaque groupe d'entrées p_1, p_2, p_3 et p_4 puis sélectionne, en association avec ce groupe, les meilleurs coefficients du filtre a_{ij}, b_{ij}, c_{ij}, d_{ij} and K_{ij} qui minimisent la distorsion. j est l'indice de la règle de filtrage et à chacune d'elles correspond un groupe de coefficients de filtrage. La sortie $\tilde{p}_i$ du filtre est calculée par l'intermédiaire de la relation suivante :

$$\tilde{p}_i = a_{ij} \cdot p_1 + b_{ij} \cdot p_2 + c_{ij} \cdot p_3 + d_{ij} \cdot p_4 + K_{ij}. \tag{4.1}$$

Pour chaque pixel de l'image décodée, le filtre unidimensionnel est appliqué dans les deux directions : horizontale et verticale. Ces deux résultats de filtrage sont ensuite moyennés pour obtenir la valeur finale du pixel.

2. Acronyme récursif signifiant *GNU's Not Unix* .

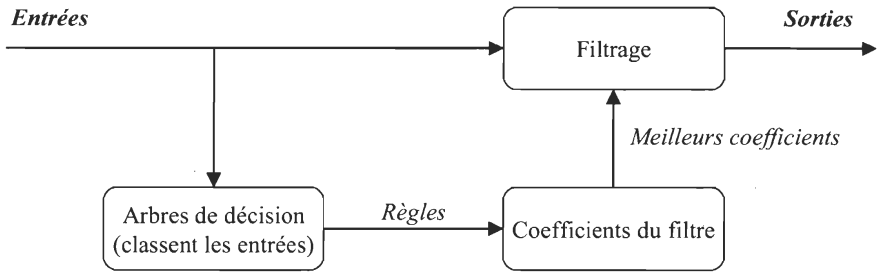


FIGURE 4.2 – Diagramme en bloc du filtre par arbres de décision.

La figure 4.3 expose une application possible du filtre dans un système de codage vidéo. Les arbres de décision (i.e. les règles de filtrage) et les valeurs des coefficients de filtrage pour chaque règle sont prédéterminés par une procédure d'entraînement depuis l'encodeur. Cette phase d'entraînement utilise un faible nombre d'images originales représentatives et de leurs versions décodées. Elle est essentielle afin de construire les arbres de décision et pour calculer les coefficients de filtrage qui minimisent la distorsion des images décodées. L'encodeur transmet ensuite les arbres de décision ainsi que les coefficients de filtrage au décodeur.

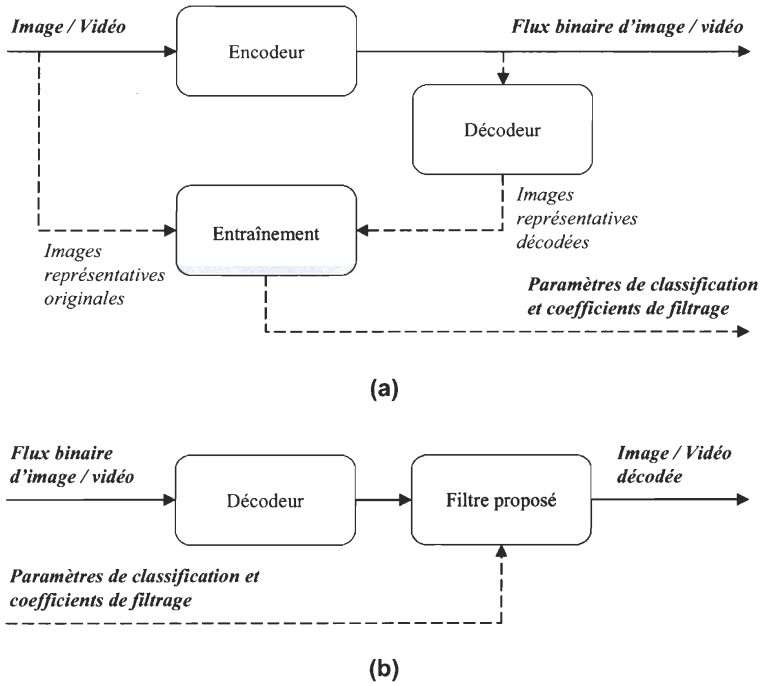


FIGURE 4.3 – Système de codage vidéo incluant le filtre par arbres de décision : (a) encodeur, (b) décodeur.

4.3 Base de données d'entraînement

L'algorithme de classification qui nous intéresse utilise une approche supervisée et requiert en conséquence une base d'exemples nécessaire au processus d'entraînement. La création de celle-ci est une étape cruciale dont dépendront fortement les performances du filtre. La base d'exemples sera fournie par l'intermédiaire d'une base de données. Cette dernière doit être créée avec les images les plus représentatives afin d'en extraire les règles les plus pertinentes. Elle contiendra un ensemble d'exemples d'entraînement nécessaires à la création des modèles de filtrage (i.e. de prédiction) relatifs à chaque type de pixel à filtrer.

Dans notre cas, la phase d'entraînement est effectuée pour chaque scène de la séquence vidéo. Une image originale et sa version décodée sont les seules images utilisées pour créer la base de données. Les images d'entraînement sont divisées en blocs de 4×4 pixels puis, afin de créer les instances de la base de données relatives au filtrage horizontal, chaque groupe de quatre pixels adjacents, dont deux sont situés de chaque bord de la limite du bloc, formeront un vecteur. Ces quatre pixels correspondront aux quatre premiers attributs de la base de données. Chaque instance de celle-ci consiste alors en deux vecteurs : le premier $\{p_1, p_2, p_3, p_4\}$ provient de l'image décodée et le second $\{p_1^O, p_2^O, p_3^O, p_4^O\}$ provient de l'image originale. La figure 4.4 en illustre le principe. Finalement, la base de données est constituée de toutes les instances possibles pouvant être extraites des images d'entraînement. Par exemple pour une image QCIF[3] (i.e. de résolution 144×176), il sera possible de créer 6192 instances horizontales distinctes.

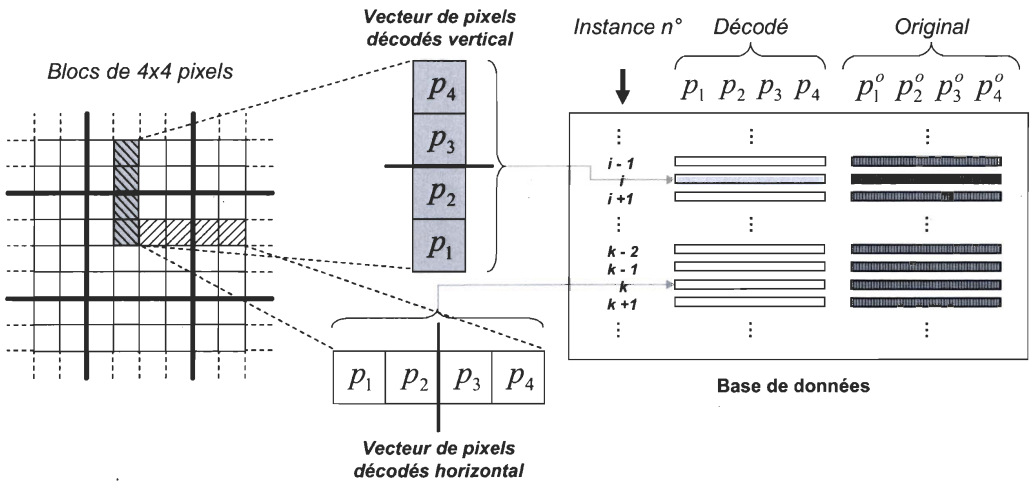


FIGURE 4.4 – Création des instances de la base de données.

La création des instances relatives au filtrage vertical s'effectue de la même manière, excepté que chaque instance consiste cette fois-ci en deux vecteurs de quatre pixels verticaux adjacents et transverses à la limite horizontale d'un bloc. Il est important de

3. QCIF : *Quarter Common Intermediate Format*, 144×176 pixels.

rappeler aussi que la base de données est créée spécifiquement pour une valeur particulière du paramètre de quantification QP ainsi que pour une scène particulière dans la séquence vidéo. Les règles extraites changeraient si une autre valeur de quantification était utilisée ou/et si la scène changeait. De plus, l'efficacité des règles décroît progressivement dans la scène au fur et à mesure que la distance (i.e. le nombre d'images intermédiaires) entre l'image à filtrer et l'image représentative augmente. Un autre élément clé à considérer ici est la taille de la base de données ; généralement, une base de données mieux fournie en exemples d'entraînement produirait des filtres plus efficaces.

Il est également important de mentionner la différence existant entre les exemples d'entraînement et les instances constituant la base de données. En fait, chaque instance correspond à quatre différents exemples d'entraînement, chacun relié à la création d'un arbre différent. Un exemple d'entraînement se représente en effet par le vecteur suivant : $\{p_1, p_2, p_3, p_4, p_i^O\}$ avec $i = 1, 2, 3, 4$. L'indice i représentera ainsi le numéro du pixel original pris en compte pour la création du modèle de prédiction (i.e. filtrage) y étant associé.

Les arbres de décision ainsi que les coefficients de filtrage associés sont créés en utilisant un algorithme de partitionnement itératif des données. Comme la base de données contient la valeur des pixels de l'image originale et de l'image décodée, les règles de filtrage résultantes et les coefficients de filtrage peuvent réduire de façon importante la distorsion de l'image décodée. De plus amples détails sont donnés dans les sections suivantes.

4.4 Sélection de l'algorithme de classification

Dans cette partie, nous décrivons les étapes qui ont mené à la sélection de l'algorithme de classification basé sur la création d'arbres de décision. En utilisant le logiciel Weka, quatre différents algorithmes supervisés ont pu être testés [58] :

(i) l'algorithme de *régression linéaire* permet d'extraire un modèle linéaire multiparamétrique pour l'ensemble des exemples d'entraînement ;

(ii) l'algorithme *RepTree* procure un arbre de régression où chaque feuille représente la valeur moyenne de l'attribut cible de la classe ;

(iii) l'algorithme *M5'* est un algorithme fournissant un arbre de modèles où chaque feuille correspond à une classe de données modélisée par une régression linéaire multiparamétrique [59] ;

(iv) le *perceptron multicouche* est un réseau de neurones organisé en couche. Il est basé sur la rétropropagation de l'erreur et peut contenir de une à plusieurs couches cachées dont chacune se constitue de un où plusieurs neurones. La version par défaut utilisée sous Weka ne contient qu'une seule couche cachée de 2 neurones. Il

peut dans certains cas être très délicat de trouver la structure adéquate du réseau. En règle générale, la nécessité d'ajouter des couches supplémentaires dépend du caractère non-linéaire des données.

La performance des différents algorithmes a été évaluée grâce aux trois mesures suivantes : (i) le coefficient de corrélation ρ entre une image filtrée et sa correspondante originale ; (ii) l'erreur absolue [4] (MAE : *Mean Absolute Error*) existante entre les deux images, et finalement (iii) l'erreur relative [5] (RAE : *Relative Absolute Error*). En plus de ces trois mesures, le nombre de règles générées est considéré pour l'algorithme *M5'* et *Reptree*. En effet, c'est un facteur déterminant pour le calcul de l'augmentation du débit-binaire engendré par la transmission du modèle de filtrage au décodeur.

Supposons qu'un modèle de filtrage *idéal* existe. Celui-ci fournirait alors une image filtrée identique à l'image originale. Ceci équivaudrait à un coefficient de corrélation égal à 1, et une erreur moyenne relative (ou absolue) égale à 0.

Les modèles générés avec chaque algorithme de classification ont été entraînés en utilisant l'image n°1 de la séquence vidéo Foreman (QCIF - 30 fps), et appliqués sur une base de données test. Cette dernière a été créée depuis l'image n°20 de la même séquence. Le tableau 4.1 montre les résultats obtenus pour les quatre algorithmes de classification utilisés. Dans le cas du perceptron multicouche, le nombre entre parenthèse représente le nombre de neurones sur l'unique couche cachée.

Il est possible de déduire de ces résultats expérimentaux que l'algorithme *M5'* semble être le plus performant pour notre application. Il surpasse les trois autres algorithmes en regard des trois mesures. De plus, le *M5'* est plus simple et plus rapide à mettre en oeuvre que le *perceptron multicouche*. Il génère également moins de règles (qui nécessiteront d'être transmises au décodeur) que *Reptree*. Nous avons également comparé le *M5'* avec un *perceptron multicouche* comportant une structure plus complexe que la version par défaut donnée par Weka (résultats non montrés ici). Le *M5'* reste plus performant en terme de complexité calculatoire et de performance.

D'autres algorithmes d'apprentissage existent pouvant être appliqués au codage vidéo. Cependant ils sont souvent plus complexes à mettre en oeuvre et nécessitent des temps de calcul assez longs. Ils ne feront pas l'objet d'étude dans la présente thèse.

4.5 Les arbres de décision

4.5.1 Principe

Les arbres de décision ont l'avantage de pouvoir être générés rapidement et d'être facilement compréhensible. La figure 4.5 illustre un exemple d'une structure d'arbre. Chaque noeud v_i correspond à un test tandis qu'une feuille L_j correspond à une classe

4. $MAE = \sum_i |\hat{y}_i - \bar{y}| / n$ avec $\hat{y}_i$ la valeur prédite de y_i, $\bar{y}$ la moyenne des y_i (sur n valeurs).
5. $RAE = \sum_i |\hat{y}_i - \bar{y}| / \sum_i (y_i - \bar{y})^2$.

TABLE 4.1 – Performances des différents algorithmes de classification avec les données de l'image n°20 de la séquence Foreman (QCIF - 30 fps).

Pixel	p_1	p_2	p_3	p_4
Régression linéaire				
Corr. coeff.	0.9923	0.9924	0.9926	0.9925
MAE	4.6018	4.5291	4.5235	4.5243
RAE (%)	9.7007	9.4794	9.4572	9.4706
RepTree				
Règles	409	577	469	433
Corr. coeff.	0.9918	0.9916	0.9919	0.9919
MAE	4.6929	4.6944	4.6515	4.6903
RAE (%)	9.8928	9.8253	9.7247	9.8181
M5'				
Règles	27	42	31	30
Corr. coeff.	0.9925	0.9927	0.9925	0.9926
MAE	4.526	4.4364	4.4889	4.4806
RAE (%)	9.5409	9.2852	9.3848	9.379
Perceptron Multicouche (2)				
Corr. coeff.	0.9922	0.9924	0.9926	0.9925
MAE	4.7572	8.5824	5.3744	4.8991
RAE (%)	10.0284	17.9627	11.2361	10.2551

de données. En conséquence, la structure de l'arbre procure un ensemble de règles individuelles qui s'appliquent à chaque modèle de données. En effet, une règle s'interprète comme un chemin depuis la racine vers une feuille et une feuille correspond à un modèle de prédiction de la classe de données y étant associée.

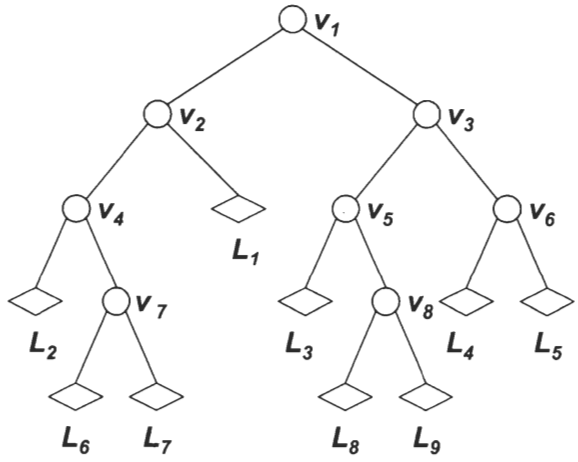


FIGURE 4.5 – Structure typique d'un arbre de décision.

Comme il a été fait mention plus tôt, les arbres de décision sont créés à partir d'une base d'exemples par utilisation d'un algorithme de partitionnement itératif. À chaque itération, l'algorithme détermine la valeur de l'attribut la plus discriminatoire. La mesure pour sélectionner la meilleure séparation dans l'arbre est souvent associée au gain d'information Δ produite par la valeur de l'attribut. Ce gain peut aussi s'interpréter comme la réduction de l'entropie $\Delta(\mathcal{X}, a_i)$ causée par le partitionnement avec l'attribut a_i :

$$\Delta(\mathcal{X}, a_i) = H(\mathcal{X}) - \sum_{j=1}^{k} \frac{\left|\mathcal{X}_{a_i=v_j}\right|}{|\mathcal{X}|} H\left(\mathcal{X}_{a_i=v_j}\right), \tag{4.2}$$

où $\mathcal{X}_{a_i=v_j} \subset \mathcal{X}$, $|\mathcal{X}|$ représente la cardinalité des valeurs de l'attribut a_i qui atteint le noeud *parent*, k est le nombre de valeurs reliées à l'attribut a_i et $\left|\mathcal{X}_{a_i=v_j}\right|$ est la cardinalité des valeurs de a_i associé au noeud *enfant* v_j. $H(\mathcal{X})$ est l'entropie de $\mathcal{X}$:

$$H(\mathcal{X}) \stackrel{def}{=} -\sum_{i} p_i \log_n p_i \ , \tag{4.3}$$

où n est la base logarithmique de la mesure de l'entropie.

4.5.2 L'algorithme *M5'*

Le *M5'* est un algorithme de création d'arbre de décision uniquement spécifique aux attributs numériques. Depuis la base d'exemples fournie par la base de données, l'algorithme génère des règles permettant de classifier les données dans différentes classes, et

de créer, pour chacune d'elles un modèle de régression linéaire multiparamétrique. La structure de l'arbre est une représentation directe des règles de classification. Chaque feuille correspond à un modèle différent de régression. Dans notre implémentation, les modèles de régression sont décrit comme :

$$\tilde{p}_i = a_{ij} \cdot p_1 + b_{ij} \cdot p_2 + c_{ij} \cdot p_3 + d_{ij} \cdot p_4 + K_{ij}, \tag{4.4}$$

où $a_{ij}, b_{ij}, c_{ij}, d_{ij}$ et K_{ij} sont les coefficients du modèle de filtrage et $\{p_1, p_2, p_3, p_4\}$ représente le vecteur de pixels décodés dans la base de données. L'indice $i = 1, 2, 3, 4$ correspond au numéro du pixel à filtrer dans ce vecteur et l'indice j correspond au numéro de la règle de filtrage liée à l'arbre i.

Dans l'arbre de décision, chaque noeud correspond à un test où un attribut (p_1, p_2, p_3 ou p_4) du vecteur de pixels décodés est comparé avec un seuil s. Dans le processus d'apprentissage, les coefficients $a_{ij}, b_{ij}, c_{ij}, d_{ij}$ et K_{ij} sont déterminés en classifiant les vecteurs de pixels décodés de la base d'exemples dans un nombre de classes et en minimisant l'erreur entre la valeur du pixel filtré $\tilde{p}_i$ et sa valeur originale p_i^O.

L'arbre de décision est créé d'après les trois étapes suivantes : (i) création d'un arbre initial ; (ii) élagage de l'arbre initial et création des modèles de filtrage ; (iii) lissage des modèles de filtrage. Ces étapes sont les mêmes que celles expliquées dans [60] et [59]. Le but étant de construire un modèle liant la valeur originale du pixel p_i^O au vecteur de pixels décodés $\{p_1, p_2, p_3, p_4\}$ de la base d'entraînement [59]. Ces étapes sont rapidement décrites dans ce qui va suivre. Les lecteurs peuvent cependant se référer aux articles [60] et [59] pour plus de détails concernant le *M5'* ainsi qu'à l'ouvrage de L. Breiman et *al.* [61] concernant les arbres de régression et de classification en général.

Création de l'arbre initial

Afin de construire l'arbre initial, *M5'* sépare récursivement les données de la base d'exemples en différents sous-ensembles (i.e. classes) [59]. Tout d'abord, un pixel constituant le vecteur de pixels décodés, p_k ($k = 1, 2, 3, 4$) est sélectionné dans le but de former le noeud racine de l'arbre. Cette sélection consiste à trouver le meilleur couple (k^*, s^*) (où s est une valeur seuil) qui va maximiser le critère de séparation. Ce dernier est utilisé pour déterminer quel pixel est le meilleur pour séparer l'ensemble $\mathcal{T}$ constitué de N exemples d'entraînement qui atteignent ce noeud parent (étant donné que c'est le noeud racine, tous les exemples sont pris en compte). Cet ensemble $\mathcal{T}$ est ensuite séparé en deux sous-ensembles : $\mathcal{T}_{sup}$ correspondant à M exemples d'entraînement où $p_k > s$ et $\mathcal{T}_{inf}$ contenant $N - M$ exemples où $p_k \leq s$. Dans le cas de *M5'*, le critère de séparation est défini comme étant la réduction de la déviation standard du pixel original p_i^O dans l'ensemble $\mathcal{T}$:

$$SDR(k, s, \mathcal{T}) = R\left(p_i^O \in \mathcal{T}\right) - \frac{M}{N} \cdot R\left(p_i^O \in \mathcal{T}_{sup}\right) - \frac{N-M}{N} \cdot R\left(p_i^O \in \mathcal{T}_{inf}\right), \tag{4.5}$$

où $i, k = 1, 2, 3, 4$. La fonction R peut se définir sur l'ensemble $\mathcal{E}$ comme [61] :

$$R(p_i^O \in \mathcal{E}) = \frac{1}{|\mathcal{E}|} \sum_{p_i^O \in \mathcal{E}} \left| p_i^0 - \tilde{p}_i \right| , \tag{4.6}$$

avec $\tilde{p}_i$ la valeur prédite du pixel p_i^0 calculée à partir d'un modèle de prédiction temporaire construit sur l'ensemble $\mathcal{T}$ et ce uniquement avec les attributs atteignant le noeud.

L'algorithme sélectionne ensuite le meilleur couple (k^*, s^*) qui maximise (4.5) :

$$(k^*, s^*) = \arg\max_{(k,s)} \left(SDR(k, s, \mathcal{T}) \right) . \tag{4.7}$$

Le même procédé est répété pour la création des autres noeuds. Le processus s'arrête dès que la déviation standard des sous-ensembles créés devient très faible (moins de 5%) où lorsqu'un faible nombre d'exemples constituent les sous-ensembles (quatre ou moins) [59, 58].

Élagage de l'arbre initial

Il est possible que l'arbre initial contienne un nombre trop important de noeuds et de feuilles. Une procédure dite d'élagage permet de le simplifier. C'est une procédure essentielle qui permet d'éviter la *sur-spécialisation* des feuilles et qui permet de réduire la distorsion. De là, des modèles plus généraux sont créés, constituant ainsi une information plus courte à transmettre au décodeur. Un modèle de filtrage peut aussi s'interpréter comme une technique de compression de l'information puisqu'il vise à réduire la distorsion de l'image et donc à lui rajouter de l'information. En effet, le but de la compression vidéo est de trouver le meilleur compromis entre la qualité des images reconstruites et la taille de l'information à transmettre. L'algorithme M5' utilise cette procédure d'élagage par défaut.

Tout d'abord, un modèle de régression linéaire est construit pour chaque noeud de l'arbre initial [59]. Pour chaque noeud, le modèle est calculé avec les attributs (i.e. les pixels) qui atteignent celui-ci, et est simplifié en vue de réduire l'erreur. Finalement, l'arbre est élagué depuis les feuilles vers la racine afin de réduire les erreurs entre le pixel filtré et l'original : $E = \left| \tilde{p}_i - p_i^O \right|$.

Lissage des modèles linéaires

Un procédé de lissage est utilisé pour atténuer les discontinuités entre les modèles linéaires adjacents des feuilles. Ce procédé augmente substantiellement la précision des modèles de filtrage [60].

4.6 Exemple de filtrage

Pour rappel, le filtre admet en entrée un vecteur de pixels décodés transverse à une limite (horizontale ou verticale) de bloc d'une image décodée. Supposons que nous cherchons à filtrer le i-ième pixel de ce vecteur de pixels décodés. La classification des données se fera alors avec l'aide du i-ième arbre de décision. De là, si les entrées sont classifiées en tant que classe k, les coefficients de filtrage correspondant à la classe k seront utilisés afin de filtrer ce pixel.

Une image à filtrer subit séparément un filtrage horizontal et un filtrage vertical. Les deux résultats de filtrage sont ensuite moyennés pour produire le résultat final.

Ce qui va suivre montre un exemple du processus de filtrage. Dans la figure 4.6, le pixel à filtrer correspond au pixel p_1 dans les deux vecteurs de pixels décodés horizontaux et verticaux. Ces vecteurs de pixels décodés sont ensuite classifiés à l'aide du premier arbre de décision. La figure 4.7 montre un exemple d'arbre de décision associé à p_1. Pour indication, cet arbre fut généré à l'aide de l'image n°1 de la séquence *Foreman* au format QCIF échantillonnée à 30 fps. Ici nous utilisons l'encodeur H.264 sans intraprédiction et avec une quantification $QPI = 35$.

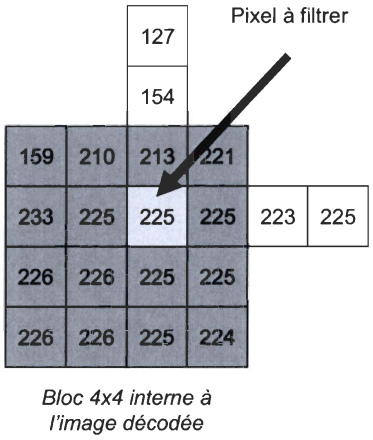


FIGURE 4.6 – Exemple d'un pixel devant être filtré.

Dans la figure 4.6, le vecteur de pixels décodés horizontal transverse à la limite du bloc est constituée par les valeurs $p_1 = 225$, $p_2 = 225$, $p_3 = 223$, $p_4 = 225$ et le vecteur de pixels décodés vertical est constitué par $p_1 = 225$, $p_2 = 213$, $p_3 = 154$, $p_4 = 127$. En conséquence et en accord avec l'arbre de décision fournit en figure 4.7, le filtrage horizontal de p_1, $\tilde{p}_1^H$, s'effectue par le modèle linéaire n°27 :

$$\tilde{p}_1^H = 0.6283 \cdot p_1 + 0.1114 \cdot p_2 - 0.0039 \cdot p_3 + 0.0027 \cdot p_4 + 58.7864. \tag{4.8}$$

De la même façon, le filtrage vertical est associé au modèle linéaire n°24 :

$$\tilde{p}_1^V = 0.1338 \cdot p_1 - 0.1166 \cdot p_2 - 0.0263 \cdot p_3 + 0.1131 \cdot p_4 + 206.1226. \tag{4.9}$$

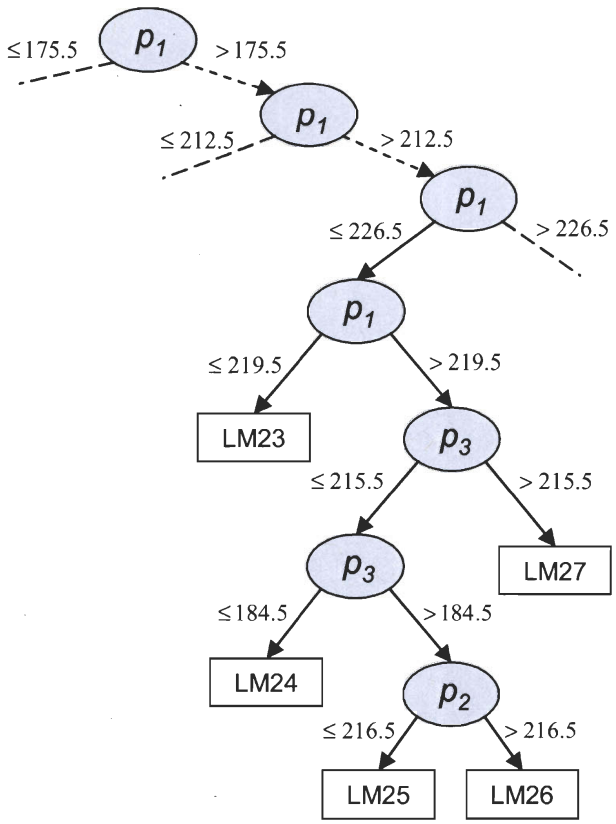


FIGURE 4.7 – Extrait de l'arbre de décision associé à p_1 (*Foreman* - QCIF - 30 fps - $QPI = 35$). LM signifie "Linear Model". Les lignes en pointillés ici permettent uniquement de s'affranchir de certaines étapes dans l'arbre afin de simplifier la structure.

Le résultat final du pixel est donné par la valeur arrondie (sur 8 bits) de la moyenne de ces deux résultats de filtrage (cf. figure 4.8) :

$$\tilde{p}_1 = \text{round}\left(\frac{\tilde{p}_1^H + \tilde{p}_1^V}{2}\right). \tag{4.10}$$

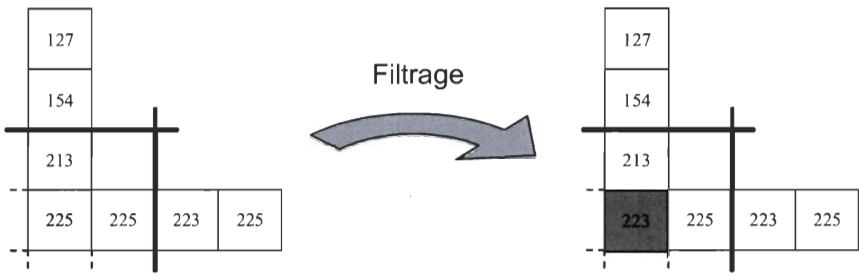


FIGURE 4.8 – Résultat du filtrage de p_1.

Les pixels restants du vecteur de pixels décodés d'entrée p_2 p_3 et p_4 sont filtrés de la même manière, excepté que chaque pixel recourt à un arbre de décision et un ensemble de règles différents.

4.7 Résultats expérimentaux

Sans pertes de généralisation, nous avons utilisé un encodeur vidéo simplifié pour évaluer les performances du filtre proposé. Cet encodeur consiste en l'utilisation de la transformée DCT entière 4×4, de la quantification et du filtrage réducteur d'artefact utilisés dans la norme H.264/AVC pour le codage des images I. Chaque image est tout d'abord divisée en blocs de 4×4 pixels. Chaque bloc est ensuite transformé et quantifié. Nous ne considérons pas ici la compensation de mouvement, l'intraprédiction, les macroblocs ainsi que les images P et B. Les images décodées sont obtenues directement depuis les coefficients de DCT quantifiés, dé-quantifiés, transformés par la DCT inverse et filtrés par le filtrage réducteur d'artefacts.

Les performances du filtrage par arbres de décision (générés avec l'algorithme *M5'*) ont été évaluées d'après trois séquences vidéo différentes au format QCIF échantillonnées à 30 fps. Chacune des séquences comporte 60 images. La base de données fut créée en utilisant la première image de chaque séquence. Les bords horizontaux des blocs sont filtrés avant les bords verticaux. La moyenne des deux résultats de filtrage a été arrondie pour une précision entière sur 8-bits.

La figure 4.9 illustre l'évolution du PSNR de la séquence *Foreman* avec un paramètre de quantification $QP = 35$. La figure 4.10 illustre quant à elle l'évolution du gain sur les 60 premières images de la séquence. Comme nous pouvons le constater, le filtre

réducteur d'artefacts de H.264 procure un gain d'environ 0.45 dB sur les 60 images. Le filtre proposé améliore quant à lui ce gain d'environ 0.69 dB (application directe sur les images filtrées par le filtre de H.264). Aussi pour illustrer l'amélioration de la qualité subjective de l'image, la figure 4.11 et la figure 4.12 montrent l'image n°20 avant et après l'application du filtre par arbres de décision. Il peut être clairement observé que le filtre proposé améliore significativement la qualité des images décodées.

Le tableau 4.2 fournit les gains moyens du PSNR obtenus pour les trois séquences en fonction de différentes valeurs de QP. Un gain significatif, jusqu'à 0.69 dB, peut être obtenu lorsque QP est supérieur à 35, tandis que le gain devient minime lorsque QP devient faible. Ceci s'explique simplement du fait que pour de grandes valeurs de quantification, plus de distorsion existe.

TABLE 4.2 – Gains moyens du PSNR obtenus sur les 60 premières images des séquences vidéo utilisées.

QP	**20**	**25**	**28**	**30**	**35**	**40**
Foreman	0	0.17	0.29	0.43	0.69	0.59
Susie	0.02	0.14	0.17	0.20	0.24	0.24
Miss America	0.05	0.24	0.26	0.33	0.41	0.33

L'algorithme de classification *M5'* a été également comparé avec l'algorithme de simple régression linéaire. La figure 4.13 illustre une comparaison de gain moyen du PSNR obtenu avec ces deux algorithmes et ce pour la séquence *Foreman*. *M5'* expose de meilleures performances excepté pour de faibles valeurs du paramètre de quantification où la régression linéaire surpasse légèrement le *M5'*.

Le filtre par arbres de décision proposé requiert la transmission d'une information décrivant les règles de filtrages (arbres de décision) ainsi qu'aux ensembles de coefficients de filtrage associés pour chaque scène de la séquence vidéo. Cette information peut être considérée comme un des points faibles du filtre. Cependant, les résultats expérimentaux ont montré que celle-ci peut devenir négligeable. Par exemple, l'information générée pour la séquence *Foreman* avec $QP = 35$ est approximativement de 44.2 kbits si écrite dans un fichier avec texte. En utilisant une méthode de compression sans pertes telle que *RAR*[6] pour compresser ce fichier texte, la taille se réduit à environ 13 kbits. Dans nos expériences, cette information est utilisée pour les 60 images de la séquence vidéo (où la scène ne change pas). Ainsi, l'élévation du débit binaire relative à sa transmission n'est seulement que de 6.5 kbps à une cadence d'image de 30 fps. La figure 4.14 expose la courbe de débit-distorsion obtenue avec le filtre proposé sur les 60 premières images de la séquence *Foreman*.

6. RAR : *Roshal ARchive* est un logiciel servant à la compression et à l'archivage de fichiers développé par E. Roshal.

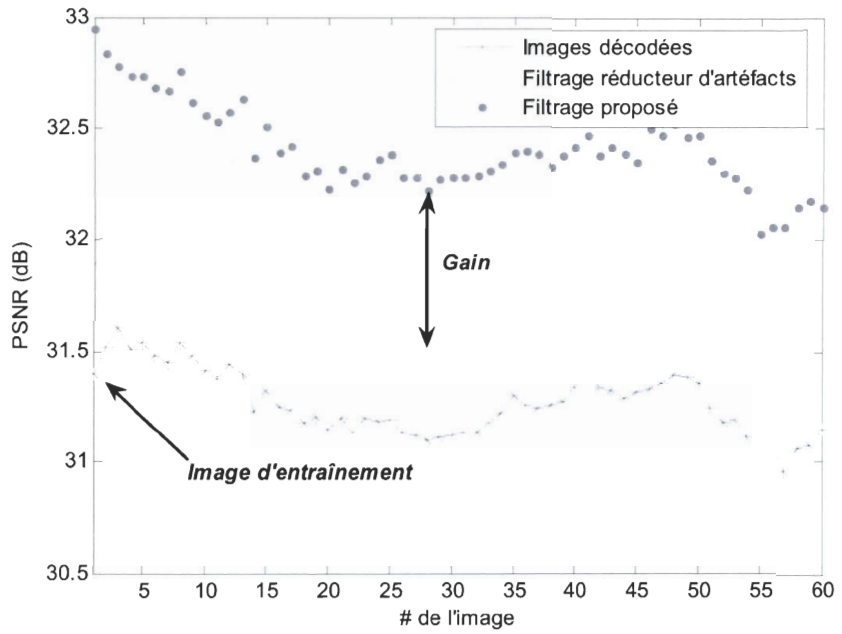


FIGURE 4.9 – Évolution du PSNR sur les 60 premières images de *Foreman* avec $QP = 35$.

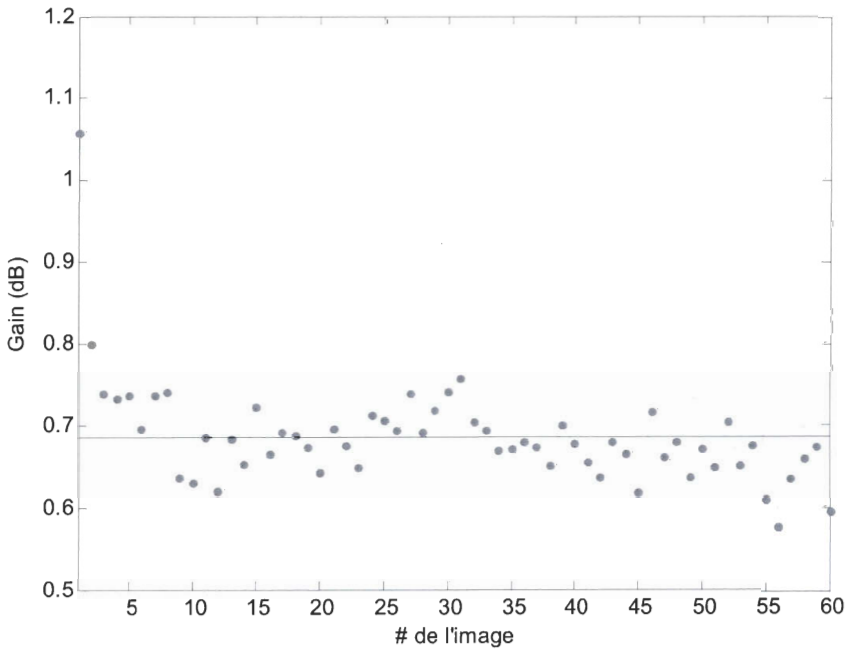


FIGURE 4.10 – Évolution du gain sur les 60 premières images de *Foreman* avec $QP = 35$. Le gain le plus élevé est donné pour l'image d'entraînement.

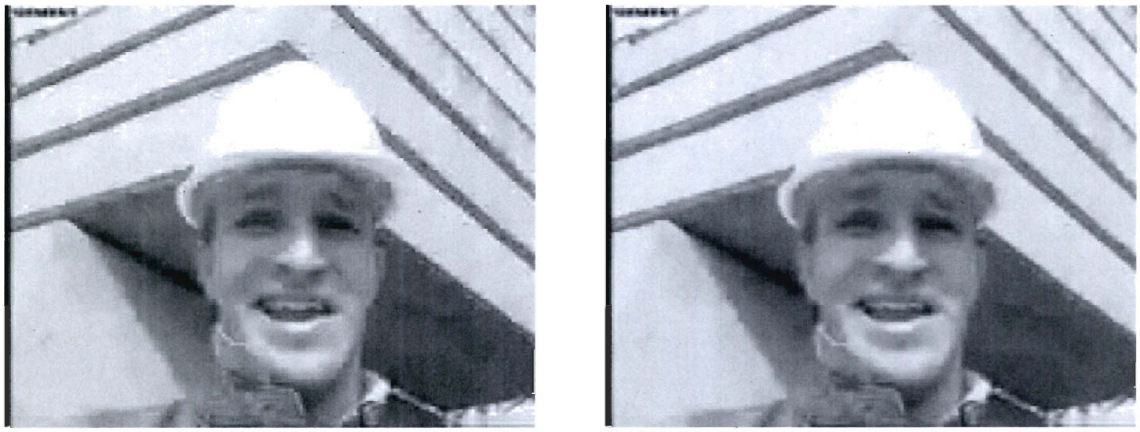


FIGURE 4.11 – Image n°20 de la séquence vidéo *Foreman* avec $QP = 35$. *Gauche* : avant filtrage par arbres de décision (31.588 dB). *Droite* : après filtrage (32.231 dB).

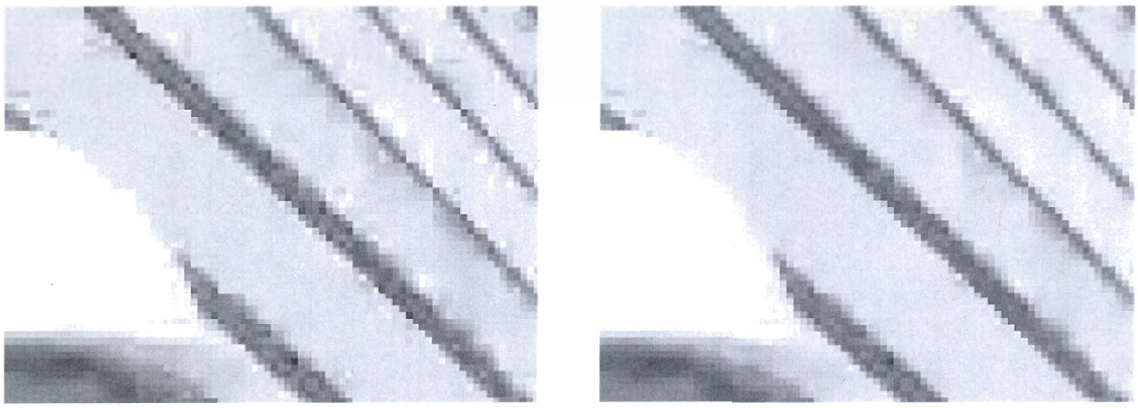


FIGURE 4.12 – Portion de l'image n°20 de la séquence vidéo *Foreman* avec $QP = 35$. *Gauche* : avant filtrage par arbres de décision (31.588 dB). *Droite* : après filtrage (32.231 dB).

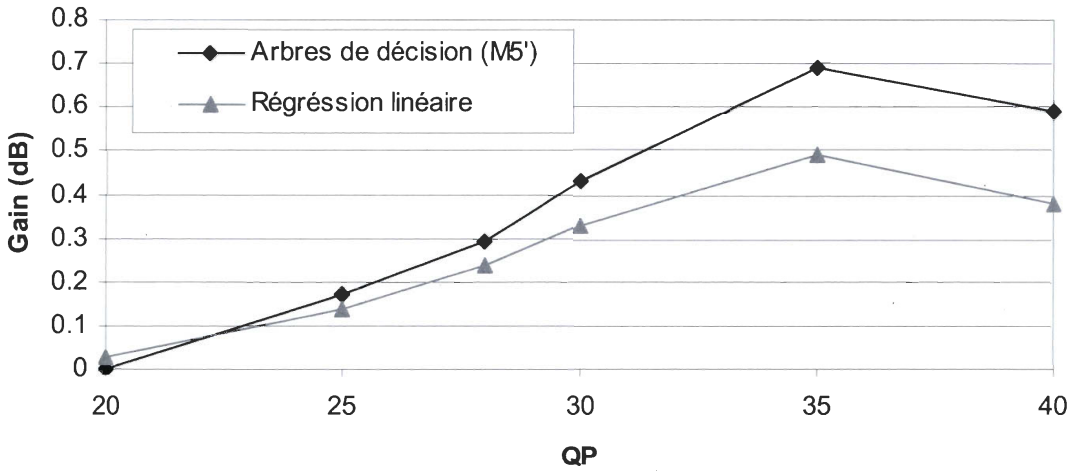


FIGURE 4.13 – Comparaison entre *M5'* et l'algorithme de régression linéaire pour la séquence *Foreman*.

Typiquement, une scène vidéo dure au moins 5 secondes, ce qui signifie qu'un changement peut intervenir après 150 images (à 30 fps) ; l'élévation du débit binaire ne devrait donc pas excéder 3 kbps.

4.8 Conclusion

Une nouvelle méthode de filtrage vidéo vient d'être présentée. Ce filtre repose sur l'utilisation d'arbres de décision et permet une réduction significative de la distorsion des images décodées. Les arbres de décision fournissent selon les entrées du filtre, les coefficients de filtrage les plus aptes à réduire la distorsion. Ces arbres de décision sont au préalable générés par l'intermédiaire d'une méthode d'apprentissage supervisée à l'aide d'une base d'exemples. Celle-ci créée à partir d'images représentatives offre alors des scenarii de filtrage adaptés à une scène spécifique de la séquence vidéo. Par ailleurs, la transmission de la structure des arbres ainsi que des ensembles des coefficients de filtrage pour chaque scène de la séquence n'apporte pas d'augmentation significative du débit binaire.

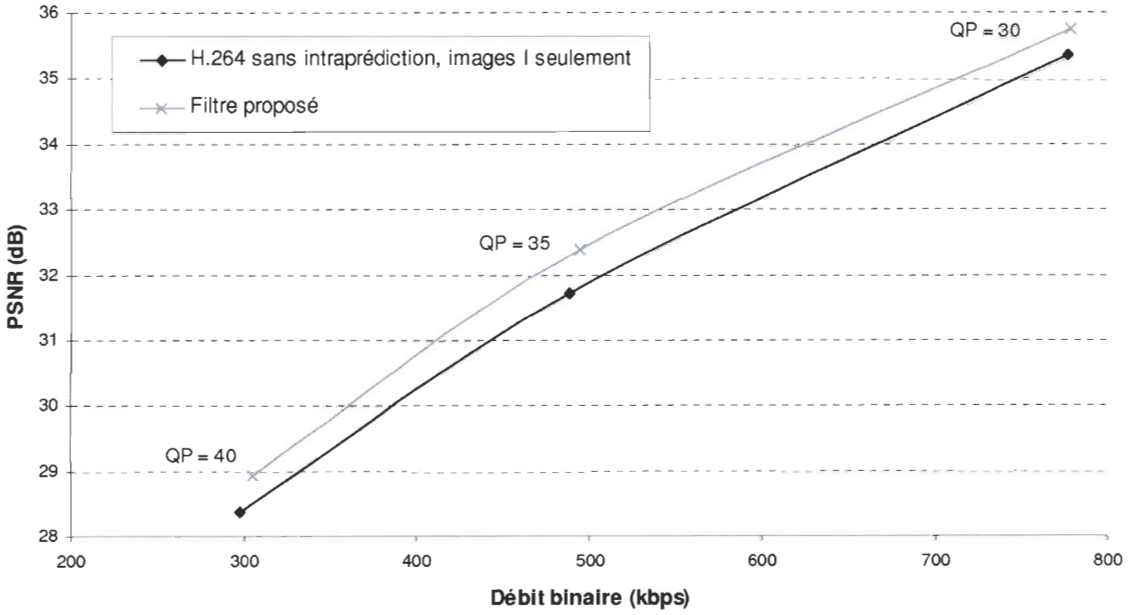


FIGURE 4.14 – Courbe de débit-distorsion obtenue avec le filtre par arbres de décision sur les 60 premières images de la séquence *Foreman* (QCIF - 30 fps).

Chapitre 5

Schéma de compression vidéo distribuée guidée par le coefficient DC

Dans ce chapitre, un nouveau schéma de codage permettant d'améliorer significativement les performances d'un système de codage vidéo distribué est présenté, celui-ci repose sur l'analyse des coefficients DC résultant de la DCT. Dans ce schéma de codage, les coefficients DC et AC de la DCT sont encodés et transmis séparément. Les coefficients DC sont transmis au décodeur et comparés à ceux de l'information latérale. Le résultat de cette comparaison permet ensuite de décider si les coefficients AC restants doivent être transmis. Les résultats expérimentaux ont montré que ce schéma de codage procurait une amélioration significative des performances tout en conservant un encodeur relativement peu complexe.

5.1 Étude statistique des différences entre les coefficients de la DCT

Nous allons examiner ici les relations statistiques entre les différences des coefficients AC et DC de la source et de l'information latérale. Tout d'abord les différences entre les coefficients quantifiés de la DCT provenant de la source et de l'information latérale seront calculées. Nous calculerons ensuite le coefficient de corrélation existant entre ces différences. Dans ce qui va suivre nous allons, pour simplifier, prendre le point de vue du codeur de Slepian-Wolf : la source référera alors aux coefficients quantifiés X^Q de la DCT et l'information latérale référera aux coefficients quantifiés Y^Q. La transformation DCT utilisée sur des blocs 4×4 ainsi que la procédure de quantification seront les mêmes que celles utilisés dans la norme H.264/AVC [20].

La figure 5.1 expose l'ordre dans lequel les coefficients de la DCT sont lus pour le codage par plage (RLE) dans la H.264/AVC. Cet ordre est en relation avec une

augmentation des fréquences spatiales (horizontales et verticales), des plus basses aux plus hautes. L'objectif de l'étude suivante est de démontrer qu'il est possible d'utiliser

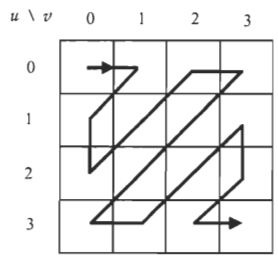


FIGURE 5.1 – Balayage en zigzag des coefficients issus de la DCT.

le coefficient DC comme *signature* du bloc entier.

Pour chaque image Wyner-Ziv (WZ) de la séquence, la différence absolue entre la source et l'information latérale est calculée. La différence absolue entre un bloc de source et son correspondant dans l'information latérale est donnée par :

$$\mathrm{AD}(u,v) = \left|X^Q(u,v) - Y^Q(u,v)\right|, \tag{5.1}$$

où le couple (u,v) détermine la position à l'intérieur du bloc. Les relations entre (u,v) et la position à l'intérieur de l'image (i,j) sont définies par :

$$u = i \bmod 4, \tag{5.2}$$

$$v = j \bmod 4. \tag{5.3}$$

Dans la suite, $\mathrm{AD}(0,0)$ référera à la différence du coefficient DC et $\mathrm{AD}(u,v)$ avec $u \neq 0$ et $v \neq 0$ à la différence des coefficients AC. Ainsi pour chaque valeur de $\mathrm{AD}(0,0)$ nous calculerons : (i) la moyenne des différences des coefficients AC $\mathrm{AD}(u,v)$ et (ii) l'écart-type de $\mathrm{AD}(u,v)$.

La figure 5.2 montre un exemple des résultats obtenus pour la différence du premier coefficient AC soit $\mathrm{AD}(0,1)$. Dans cet exemple, seules sont représentées les valeurs des différences $\mathrm{AD}(0,0)$ comprises entre 0 et 50 (i.e. $0 \leq \mathrm{AD}(0,0) \leq 50$). Le nombre de blocs de chaque valeur de $\mathrm{AD}(0,0)$ est représenté sur une échelle logarithmique. De plus, ces résultats sont calculés pour les 50 premières images WZ de la séquence Foreman (QCIF - 30fps) et pour un paramètre de quantification (propre aux images WZ) $QPW = 0$. L'information latérale est créée par l'intermédiaire d'un algorithme d'interpolation à compensation de mouvement. Celui-ci permet d'interpoler une image entre deux images K décodées consécutives. Le paramètre de quantification utilisé pour les images K (QPI) est mis à 0.

Afin de mieux illustrer la relation existante entre $\mathrm{AD}(0,1)$ et $\mathrm{AD}(0,0)$ nous avons également représenté en figure 5.3 les différentes valeurs possibles de $\mathrm{AD}(0,1)$ en fonction des valeurs de $\mathrm{AD}(0,0)$ et ce uniquement pour la première image WZ de la séquence Foreman.

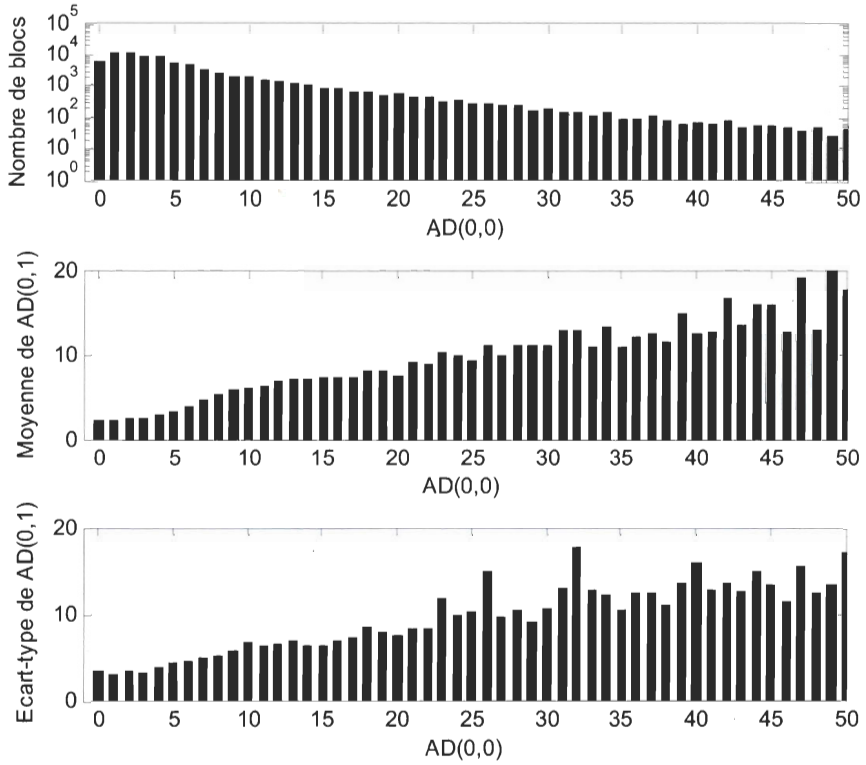


FIGURE 5.2 – Résultats obtenus pour la différence du coefficient AC AD$(0,1)$ avec $QPW = 0$. *Haut* : Nombre de blocs correspondant à chaque valeur de AD$(0,0)$; *Centre* : Moyenne de AD$(0,1)$ en fonction de AD$(0,0)$; *Bas* : Ecart-type de AD$(0,1)$ en fonction de AD$(0,0)$.

Nous pouvons déduire de la figure 5.2 que la différence des coefficients AC AD$(0,1)$ augmente de façon générale en fonction de l'augmentation de la différence du coefficient DC AD$(0,0)$. Plus spécifiquement, les différences du coefficient AC sont petites lorsque les différences du coefficient DC sont petites. De plus, la plupart des blocs possède une différence des coefficients DC petites, comme le montre la figure 5.2.

Des résultats assez similaires ont été obtenu pour des coefficients de plus haute fréquence et pour différentes valeurs du paramètre de quantification QPW. Ils ne seront pas présentés ici.

Pour mieux mettre en évidence la relation entre la source et l'information latérale, nous avons calculé le coefficient de corrélation entre les différences du coefficient DC, AD$(0,0)$ et celles des coefficients AC, AD(u,v) avec $(u,v) \neq (0,0)$. Avec ce coefficient, il devient alors aisé d'illustrer la relation entre AD$(0,0)$ et AD(u,v). Le tableau 5.1 montre les valeurs des coefficients de corrélation obtenus pour $QPW = 0$.

Ce tableau montre d'une manière générale que les différences relatives aux coefficients AC sont suffisamment corrélés avec les différences du coefficient DC. Cette corrélation est assez élevée si l'on considère le fait que la corrélation entre les coeffi-

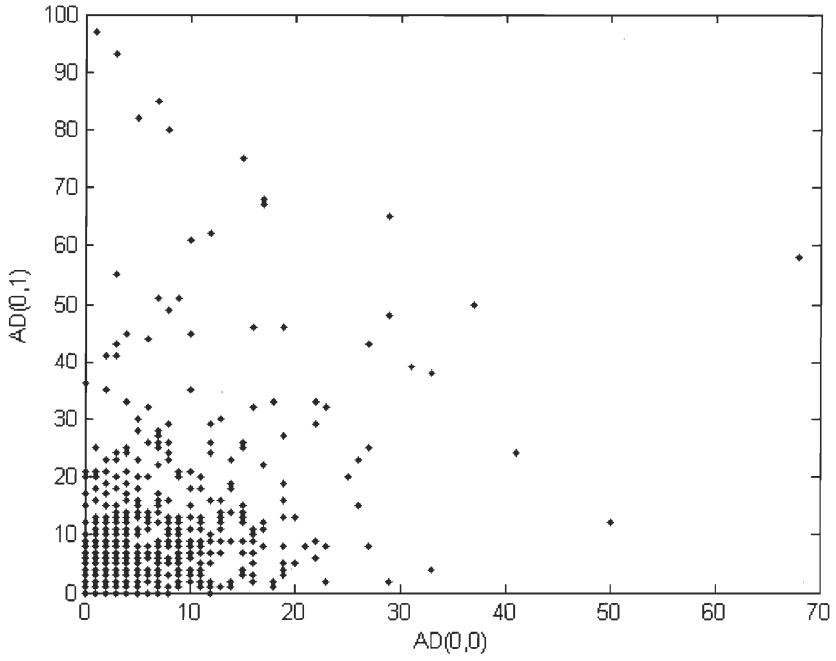


FIGURE 5.3 – Répartition des différences des coefficients AC AD(0,1) en fonction des différences des coefficients DC AD(0,0) pour la première image WZ de la séquence *Foreman*.

TABLE 5.1 – Coefficient de corrélation entre la différence absolue AD$(0,0)$ (DC) et les autres différences absolues AD(u,v) (AC).

$u \setminus v$	0	1	2	3
0	1.0000	0.4684	0.3335	0.2216
1	0.5499	0.3863	0.2871	0.2179
2	0.4298	0.2980	0.2293	0.1879
3	0.2940	0.2256	0.1987	0.1661

cients de la transformée devrait être nulle. Elle s'explique simplement par l'utilisation de la valeur absolue dans la relation (5.1). Les résultats impliquent alors que pour de larges différences du coefficient DC de larges différences des coefficients AC peuvent être observées. Cela signifie également que le coefficient DC peut être exploiter pour déterminer quels blocs de l'information latérale ont le plus besoin d'être corrigés. Quoi qu'il en soit, nous verrons par la suite que, seule une différence $\mathrm{AD}(0,0)$ égale à 0 ou non est d'intérêt dans le système proposé.

5.2 Architecture du codec proposé

L'architecture du codec est illustrée à la figure 5.4. La différence majeure entre celui-ci et le codec présenté dans [22] et [5] est la présence d'un encodeur/décodeur de Slepian-Wolf à double couche. La figure 5.5 donne de plus amples détails concernant ces derniers. La contribution réside ici dans la transmission de deux différentes couches de coefficients à différents instants. Chaque image WZ peut effectivement se décomposer en deux couches bien distinctes : la couche DC et la couche AC. La couche DC consiste en un regroupement de l'ensemble des coefficients DC quantifiés des blocs de l'image (WZ ou l'information latérale). De la même manière, la couche AC regroupe l'ensemble des coefficients AC quantifiés de l'image. Actuellement notre codec utilise un groupe d'image de longueur 2 : GOP= IW. L'image WZ (précédemment notée W) est encodée par l'encodeur de Wyner-Ziv tandis que les images K (notée I) sont intra-codées par l'intermédiaire de la norme H.264. Le même système peut cependant être facilement adapté pour des groupes d'images plus longs où plus d'images WZ sont impliquées.

La figure 5.4 est une représentation des différentes étapes de compression d'une image WZ :

(i) La DCT est d'abord appliquée sur chaque bloc de l'image ; les coefficients en résultant sont alors séparés suivant la couche DC et la couche AC tel que montré à la figure 5.5.

(ii) La couche DC est ensuite encodée par l'intermédiaire d'un code de canal (turbo-codes, LDPC, ...) puis transmise au décodeur.

(iii) Une fois le décodage de la couche DC effectué, le décodeur compare les coefficients DC de chaque bloc à ceux correspondant de l'information latérale, puis effectue une classification sur chacun des blocs. Si deux coefficients DC sont semblables, le bloc de la source ne sera pas considéré comme porteur de suffisamment d'information par rapport à l'information latérale et sera classifié en tant que bloc *non-significatif*. Dans le cas contraire si les deux coefficients DC sont différents, le bloc sera classifié comme *significatif* à l'égard de l'information latérale. Les résultats de ce processus de classification seront alors transmis vers l'encodeur.

(iv) L'encodeur utilise les résultats de la classification pour *guider* l'encodage de la

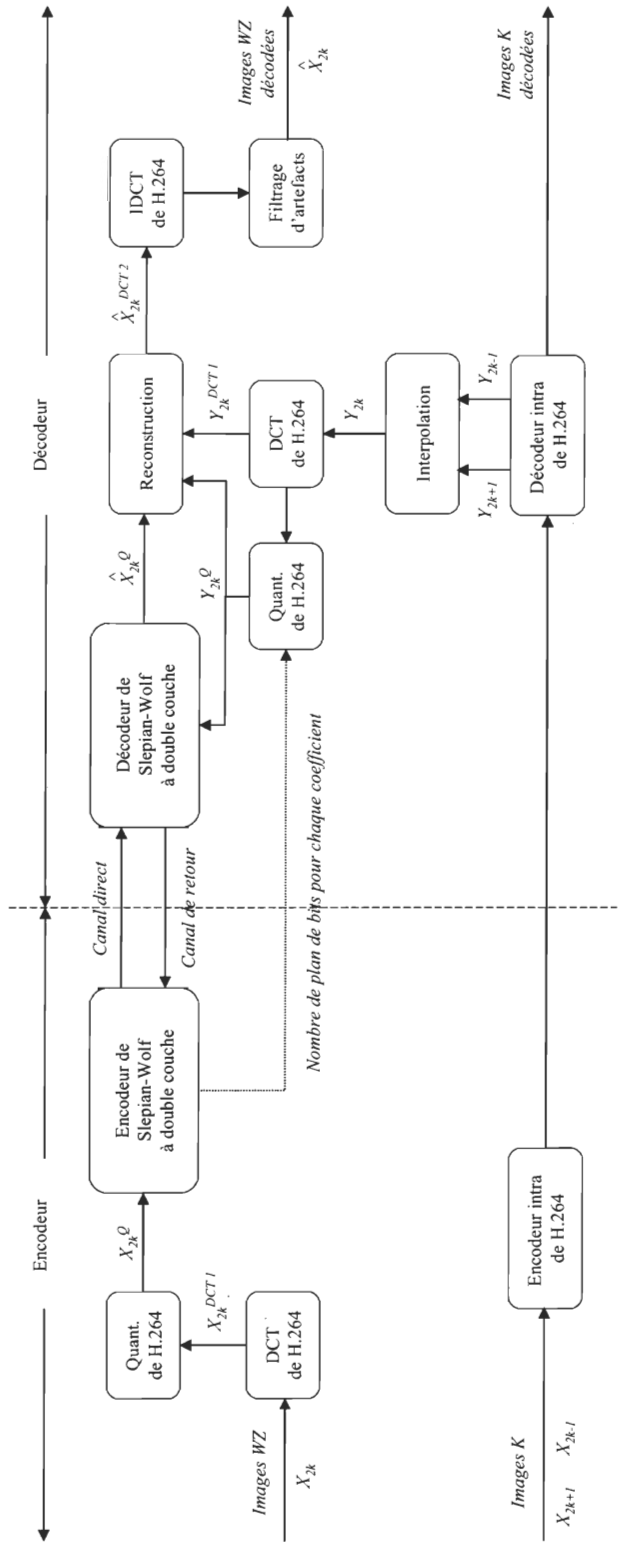


FIGURE 5.4 – Diagramme en bloc du codec proposé.

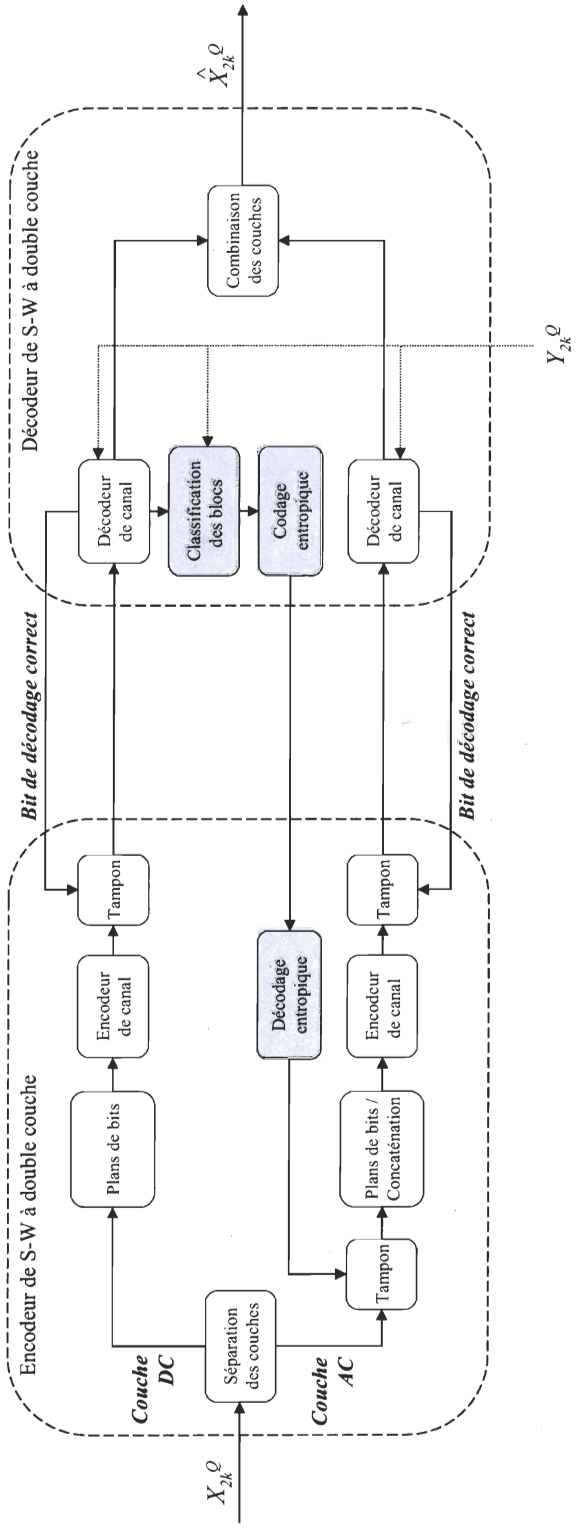


FIGURE 5.5 – Diagramme en bloc de l'encodage et du décodage de Slepian-Wolf à double couche.

couche AC. Les coefficients des blocs signifiants seront les seuls encodés puis transmis au décodeur.

(v) Finalement, après le décodage des coefficients AC des blocs les plus significatifs, le décodeur combine les deux couches DC et AC.

Le détail des procédures d'encodage et de décodage est fourni dans les sections suivantes.

5.2.1 Encodeur

Soient $X_1, X_2, ..., X_N$ les images provenant d'une séquence vidéo. Les images *paires*, X_{2k} correspondent aux images WZ tandis que les images *impaires* correspondent au images K. Les images K sont encodées par l'intermédiaire de l'intra-codeur relatif à la norme H.264/AVC. Chaque image WZ est en revanche encodée indépendamment des autres images. Pour chacune d'elle, la transformation DCT à arithmétique entière propre à H.264 [56] est appliquée sur chaque bloc 4×4 de l'image, produisant ainsi l'image transformée X_{2k}^{DCT1} :

$$X_{2k}^{DCT1} = \mathbf{C} X_{2k} \mathbf{C}^T, \tag{5.4}$$

où $\mathbf{C}$ est la matrice de transformation définie comme suit :

$$\mathbf{C} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix}. \tag{5.5}$$

De plus, comme la matrice $\mathbf{C}$ ne contient que quatre valeurs distinctes, i.e., -2, -1, 1, et 2, cette transformation peut très simplement être implémentée par l'utilisation unique d'additions et d'opérations de décalage.

La prochaine étape consiste en l'opération de quantification des coefficients de la DCT par l'utilisation du même procédé décrit dans [56]. Les symboles X_{2k}^{Q} sont ainsi produits :

$$\begin{aligned} X_{2k}^{Q}(i,j) = \operatorname{sign}\Big(X_{2k}^{DCT1}(i,j)\Big) \cdot \\ \Big[\Big(\big|X_{2k}^{DCT1}(i,j)\big| \, M(Q_M, r) + f \cdot 2^{(15+Q_E)}\Big) >> (15 + Q_E)\Big]. \end{aligned} \tag{5.6}$$

Q_M et Q_E sont reliés au paramètre de quantification QP tel que :

$$Q_M = QP \bmod 6, \tag{5.7}$$

$$Q_E = QP/6. \tag{5.8}$$

La matrice M est définie dans [56] comme :

$$M = \begin{bmatrix} 13107 & 5243 & 8066 \\ 11916 & 4660 & 7470 \\ 10082 & 4194 & 6554 \\ 9362 & 3647 & 5825 \\ 8192 & 3355 & 5243 \\ 7282 & 2893 & 4559 \end{bmatrix}. \tag{5.9}$$

Le paramètre f dans l'équation (5.6) détermine la zone morte du quantifieur. Sa valeur typique se situe entre 0 et 1/2. Cependant pour notre application, sa valeur sera fixée à 1/2. La valeur du paramètre r dans (5.6) est reliée à la position (u, v) du coefficient considéré dans le bloc transformé (voir la figure 5.6) :

- $r = 0$ pour $(u, v) = \{(0,0), (0,2), (2,0), (2,2)\}$,
- $r = 1$ pour $(u, v) = \{(1,1), (1,3), (3,1), (3,3)\}$,
- $r = 2$ ailleurs.

0	2	0	2
2	1	2	1
0	2	0	2
2	1	2	1

FIGURE 5.6 – Valeurs de r selon la position du coefficient dans le bloc.

Les symboles quantifiés subissent ensuite différents processus selon qu'ils appartiennent ou non à la couche DC. Pour la couche DC, les coefficients quantifiés sont groupés et une extraction des plans de bits est effectuée. Chaque plan de bits est ensuite encodé à l'aide d'un code de canal. Finalement, les bits ainsi formés sont stockés dans une mémoire tampon avant d'être transmis au décodeur.

Pendant ce temps, les coefficients AC quantifiés sont mis en mémoire. Ils sont stockés jusqu'à ce que l'encodeur soit informé des coefficients des blocs spécifiques devant être transmis. Les coefficients sélectionnés sont ensuite groupés en accord avec leur position (ou leur fréquence) dans le bloc puis séparés sous forme de plan de bits. Un tel groupe de coefficients sera également appelé à l'avenir une *bande* [1].

Pour savoir quels blocs sont significatifs en terme d'apport informationnel pour l'information latérale, l'encodeur doit décompresser une séquence reçue du décodeur et

1. Le terme bande est relatif aux bandes de fréquence utilisées en communication numérique. Pour notre cas, il existera ainsi 16 bandes différentes relevant du découpage en bloc 4×4. Ce terme sera tout de même à distinguer du terme *couche*. En effet, la couche DC sera constituée d'une seule bande de fréquence tandis que la couche AC sera constituée au maximum des 15 autres bandes de fréquence.

comportant les résultats de la classification des blocs. Ces résultats ont subi au préalable une compaction au décodeur. En conséquence, l'encodeur voit sa complexité légèrement augmentée tout en restant globalement très simple.

De plus, durant le processus de création de la couche AC, il est possible que la longueur des plan de bits ne soit pas un multiple de la longueur d'entrée L des codes de canal. Pour cette raison, il est nécessaire d'appliquer un procédé spécial comme illustré à la figure 5.7. Pour chaque groupe de coefficients AC quantifiés (ou chaque bande relative aux coefficients AC), les plan de bits associés sont concaténés et ensuite fragmentés en messages plus courts de longueur L. Ceux-ci sont ensuite transmis à l'encodeur de canal. Pareillement, selon la longueur D des bits restants , une condition permet de déterminer si ces bits peuvent être transmis avec ou sans compression. D'une part, si D permet d'obtenir une compression efficace, c'est-à-dire si sa valeur est supérieure à une valeur seuil D_C ($D > D_C$), des bits additionnels dit de *bourrage*[2] seront ajoutés afin d'obtenir un nouveau message de longueur L. Ces bits de bourrage seront de valeur nulle pour notre application. La version compressée (de manière distribuée) de ce message sera alors attendue d'avoir une longueur inférieure à D. En revanche si D est inférieure (ou égale) à la valeur seuil D_C ($D \leq D_C$), les bits restant seront transmis directement au décodeur sans compression.

Afin de réduire le débit binaire global, l'encodeur a également la possibilité de transmettre uniquement P des 16 coefficients présents dans chaque bloc, induisant ainsi P bandes distinctes. La transmission de ces P coefficients est faite en accord avec l'ordre de lecture en zigzag des coefficients comme défini dans [20] ou à la figure 5.1. Ainsi, si l'utilisateur décide de transmettre un seul coefficient dans chaque bloc ($P = 1$), la transmission des résultats de la classification des blocs à l'encodeur pourra être évitée, réduisant par la même occasion le débit binaire.

Par ailleurs, il est nécessaire de transmettre le nombre de plans de bits $R_{u,v}$ requis pour chaque bande de fréquence. De là, un message constitué des P valeurs sera créé et transmis au décodeur. Cette information aussi courte qu'elle soit (globalement inférieure ou égale à 80 bits comme nous la définissons) est essentielle pour le décodage. En effet, grâce à elle, le décodeur sait exactement comment distinguer chaque message relatif à chaque plan de bit et chaque coefficient constituant la bande.

5.2.2 Décodeur

Dans le décodeur, les images K sont décodées via le décodeur conventionnel d'intra-images de H.264/AVC. Les images WZ sont en revanche décodées conjointement avec l'information latérale. Cette dernière est une image générée par interpolation entre deux images K décodées consécutives. Au cours de la première étape de compression,

2. Le terme « bit de bourrage » (*bit stuffing* en anglais) est emprunté de certains protocoles réseaux, tel que le protocole IP.

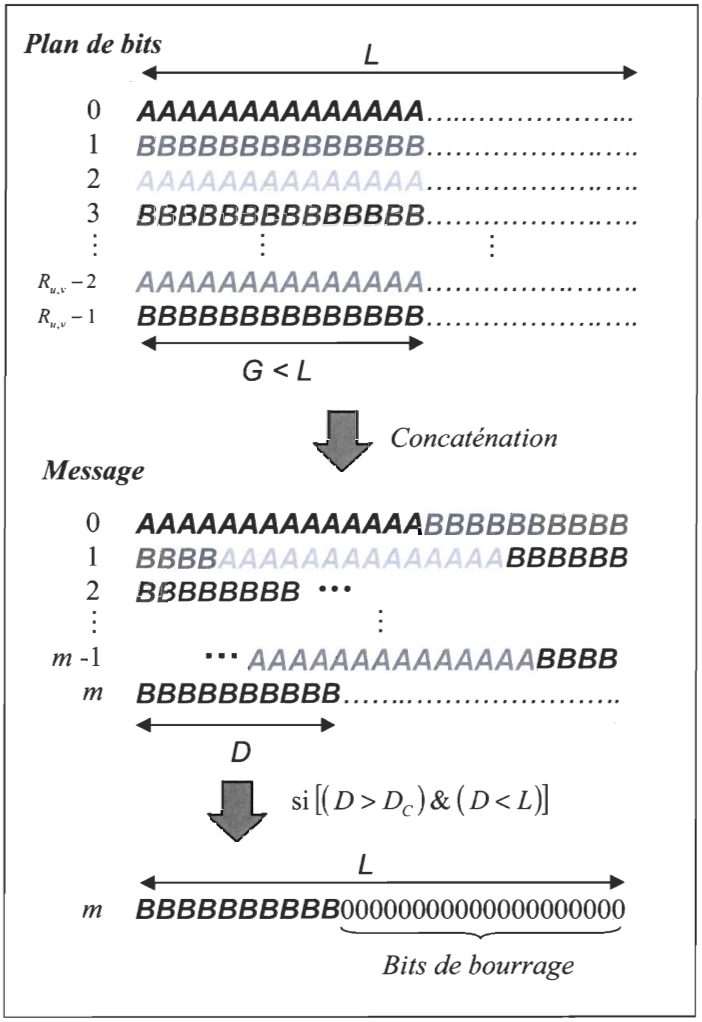


FIGURE 5.7 – Exemple de concaténation des plans de bits : chaque plan de bits possède une longueur initiale G inférieure à L. Après la concaténation, si les bits restants ont une longueur D plus grande que D_C, ils seront concaténés avec des bits de valeur nulle afin de produire un nouveau message de longueur L. Dans le cas contraire, ils seront transmis directement au décodeur.

le décodeur reçoit tous les coefficients DC quantifiés de l'image WZ. Ceux-ci sont comparés aux coefficients correspondant de l'information latérale. Pour chaque bloc, si la valeur des deux coefficients DC sont semblables un message est transmis à l'encodeur indiquant que les coefficients AC quantifiés du bloc n'ont pas besoin d'être envoyés. La nouveauté de cette méthode est que le coefficient DC agit comme une signature pour le bloc. Son avantage est que l'on peut éviter d'avoir à transmettre les coefficients AC, conduisant ainsi à une réduction significative du débit binaire des images WZ. Durant le processus de classification des blocs, une signature binaire de chaque bloc est générée. Pour l'instant, une valeur de 1 indique que les coefficients DC quantifiés sont semblables. De même un 0 indique qu'ils sont différents. De là, une séquence de longueur correspondante au nombre total de blocs dans l'image sera produite. Pour réduire l'impact de cette séquence sur le débit binaire global, cette séquence sera compactée via un encodeur sans perte puis transmise à l'encodeur. Si aucune compaction n'est appliquée, 1584 bits pour une image QCIF (correspondant aux 1584 blocs de l'image) devront être transmis. Ceci équivaudrait à une augmentation de 23.76 kbps à 15 fps pour les images WZ [3]. Fort heureusement, avec l'utilisation d'une information latérale de haute qualité et pour de fortes quantifications (i.e. faibles débits binaires), cette séquence sera grandement compactée. En effet, dans ce cas, seul quelques blocs de l'information latérale auront besoin d'être corrigés.

La plupart des systèmes de codage vidéo distribué actuels incluent un canal de retour utilisé par le décodeur. L'objectif principal de ce canal est d'informer l'encodeur pour que celui-ci transmette plus de bits de parité (ou de syndrome) lorsque le message n'a pas pu être décodé convenablement. Ce processus permet un contrôle direct du débit binaire mais peut induire une augmentation significative indirecte de celui-ci à cause du fort nombre de requêtes potentielles [62, 63]. Un moyen simple fut trouvé pour palier à ce problème. Il s'agit simplement de forcer l'encodeur de Slepian-Wolf à envoyer continuellement des bits de parité (pour chaque période de temps T) jusqu'à ce qu'il reçoive un message (d'un ou plusieurs bits) du décodeur indiquant que le décodage fut correctement effectué avec les derniers bits de parité reçus. Ceci permet également à l'encodeur de débuter la transmission d'un autre message. Dans notre cas, seulement un bit est utilisé pour confirmer un décodage correct. Ainsi l'impact de ce procédé sur le débit global devient négligeable.

La compression sans perte située dans l'encodeur de Slepian-Wolf repose paradoxalement sur des codes de canal. Dans notre application, les codes doivent être courts et efficaces. Pour cela nous avons préféré les codes LDPC (cf. Annexe B) plutôt que les turbo-codes. De plus, les codes LDPC procurent un moyen plus *pratique* pour vérifier si le message décodé est correct tout en possédant une complexité calculatoire au décodage moins importante que les turbo-codes. Les codes LDPC ne sont cependant pas efficaces en terme d'adaptabilité au débit à moins qu'ils soient couplés avec un accumulateur de

3. Ceci correspond également à 30 fps pour un GOP $\equiv$ IW.

syndrome [6]. Pour cette raison, nous avons utilisé les mêmes codes LDPCA [4] que dans [6]. La matrice de parité utilisée est irrégulière, carrée et de longueur L de 396 bits. Ces codes courts réduisent la consommation de mémoire du décodeur sans pour autant affecter de manière significative les performances de décodage comparativement à des codes plus longs [6].

L'utilisation de codes courts fait que des erreurs au décodage apparaissent plus fréquemment. Certains motifs d'erreurs engendrent le même syndrome que celui reçu et il devient alors impossible au décodeur de les détecter. Afin d'aider ce dernier, le codeur de canal a été couplé avec des bits de *contrôle de redondance cyclique* (CRC : *Cyclic Redundancy Check*). Ainsi pour chaque syndrome émis vers le décodeur, le CRC du message correspondant est transmis en même temps. Le décodeur, en calculant le CRC du message décodé, peut ainsi vérifier son exactitude. Si le CRC calculé est différent du CRC reçu, le décodeur continuera alors de recevoir des bits de syndrome jusqu'à ce que les deux valeurs de CRC correspondent. De cette manière, une réduction significative d'erreurs de décodage s'opère par un apport réduit de bits supplémentaires. Dans notre cas, le polynôme générateur utilisé pour le calcul du CRC est de degré 8 et vaut $G(x) = x^8 + x^2 + x + 1$. Chaque CRC ainsi calculé se représente sur 8 bits.

Pour prendre en compte l'accumulateur, nous utilisons pour le décodage une version modifiée de l'algorithme *Somme-Produit* dans sa version vraisemblance logarithmique (cf. Annexe B). Nous supposons dans la suite que tous les bits de message viennent du même plan de bits b. Aussi, pour assurer un décodage correct (ou une correction) de chaque message $\left[y_0^b\, y_1^b\, \ldots\, y_{L-1}^b\right]$ avec $y_n^b \in \{0,1\}$, il faut initialiser l'information intrinsèque I_{int}^n correspondante à chaque bit y_n^b. L'information intrinsèque est à situer dans un contexte de codage de canal. Elle n'est en rien relative à la définition de l'information en théorie de l'information (cf. Chapitre 2). L'information intrinsèque s'étend dans une plage allant de $-\infty$ à $+\infty$ et renseigne sur la certitude que l'on a de la valeur d'un bit (l'infini correspondant à la certitude absolue). Elle dépend des caractéristiques du canal existant entre les plans de bits de chaque coefficient de transformée quantifié de la source X_{2k}^Q et de l'information latérale Y_{2k}^Q. Ce canal est habituellement modélisé par un canal binaire symétrique (cf. figure 3.9) avec une probabilité d'entrecroisement ϵ_b. Dans ce cas particulier, l'information intrinsèque s'écrit :

$$I_{int}^n \triangleq \log\left(\frac{\Pr(y_n^b|x_n^b=0)}{\Pr(y_n^b|x_n^b=1)}\right) \tag{5.10}$$

$$= \begin{cases} \log\left(\frac{1-\epsilon_b}{\epsilon_b}\right), & \text{si } y_n^b = 0 \\ \log\left(\frac{\epsilon_b}{1-\epsilon_b}\right), & \text{si } y_n^b = 1 \end{cases} \tag{5.11}$$

$$= (-1)^{y_n^b} \log\left(\frac{1-\epsilon_b}{\epsilon_b}\right). \tag{5.12}$$

4. *Low-Density Parity-Check Accumulate.*

Durant le processus décodage, la valeur de ϵ_b devra être au préalable estimée.

La reconstruction des valeurs quantifiées de la source n'utilise pas le procédé standard de reconstruction présenté dans [5]. Cette dernière n'est en effet pas optimale [53]. Le procédé optimal de reconstruction, plus complexe à mettre en oeuvre, est présenté à l'annexe E. Pour notre application, nous avons préféré utiliser la méthode de reconstruction décrite ci-dessous. Celle-ci offre l'avantage d'avoir de meilleurs résultats pour les forts débits binaires (en comparaison de la méthode standard) tout en étant moins complexe que l'optimale :

$$\hat{X}_{2k}^{DCT2}(i,j) = \begin{cases} Y_{2k}^{DCT2}(i,j) & \text{, si } \hat{X}_{2k}^{Q}(i,j) = Y_{2k}^{Q}(i,j), \\ \text{round}\,(I_{mid}) & \text{, autrement,} \end{cases} \tag{5.13}$$

avec

$$Y_{2k}^{DCT2}(i,j) = \text{sign}\Big(Y_{2k}^{DCT1}(i,j)\Big).\Big[\big|Y_{2k}^{DCT1}(i,j)\big|\, M(Q_M,r)S(Q_M,r) >> 2^{15}\Big]. \tag{5.14}$$

Pareillement I_{mid} peut prendre plusieurs valeurs selon le type de coefficients considéré et selon sa valeur par rapport à zéro. Ainsi pour les coefficients DC, I_{mid} se calcule comme suit :

$$I_{mid} = \begin{cases} (1-f)\cdot S(Q_M,r).2^{Q_E-1} & \text{, si } \hat{X}_{2k}^{Q}(i,j) = 0, \\ \Big(\hat{X}_{2k}^{Q}(i,j) - f + 0.5\Big)\cdot S(Q_M,r).2^{Q_E} & \text{, autrement.} \end{cases} \tag{5.15}$$

De même pour les coefficients AC, nous avons :

$$I_{mid} = \begin{cases} 0 & \text{, si } \hat{X}_{2k}^{Q}(i,j) = 0, \\ \Big(\hat{X}_{2k}^{Q}(i,j) - f + 0.5\Big)\cdot S(Q_M,r).2^{Q_E} & \text{, si } \hat{X}_{2k}^{Q}(i,j) > 0, \\ \Big(\hat{X}_{2k}^{Q}(i,j) + f - 0.5\Big)\cdot S(Q_M,r).2^{Q_E} & \text{, si } \hat{X}_{2k}^{Q}(i,j) < 0. \end{cases} \tag{5.16}$$

La matrice S est définie dans [56] comme :

$$S = \begin{bmatrix} 10 & 16 & 13 \\ 11 & 18 & 14 \\ 13 & 20 & 16 \\ 14 & 23 & 18 \\ 16 & 25 & 20 \\ 18 & 29 & 23 \end{bmatrix}. \tag{5.17}$$

De même, pour les blocs où les coefficients AC quantifiés n'ont pas été transmis (et donc pour les blocs significatifs et non-significatifs), le procédé de reconstruction utilise les coefficients de l'information latérale.

Une fois que tous les coefficients $\hat{X}_{2k}^{DCT2}(i,j)$ ont été reconstruits, la transformation entière en cosinus discret inverse de H.264 est appliquée pour calculer l'image WZ décodée avant l'opération de filtrage. Ici encore, cette transformation est la même que celle décrite dans [56], et ainsi pour chaque bloc :

$$\hat{X}_{2k}^{BF} = \mathbf{C}_{inv}\hat{X}_{2k}^{DCT2}\mathbf{C}_{inv}^{T} , \tag{5.18}$$

avec la matrice de transformation inverse :

$$\mathbf{C}_{inv} = \begin{bmatrix} 1 & 1 & 1 & 1/2 \\ 1 & 1/2 & -1 & -1 \\ 1 & -1/2 & -1 & 1 \\ 1 & -1 & 1 & -1/2 \end{bmatrix} . \tag{5.19}$$

Comme le montre la figure 5.4, le schéma de codage vidéo distribué présenté ici utilise le filtrage réducteur d'artefacts propre à la norme H.264/AVC. Des résultats expérimentaux ont montré qu'il est important que ce filtre soit actif avec la méthode de reconstruction proposée et le guidage par coefficients DC. Son utilisation permet surtout d'améliorer la qualité des images aux bas débits. Ce filtre est quasiment identique à celui du standard H.264/AVC [20, 3] si ce n'est qu'il opère seulement sur des blocs 4×4. De plus, certaines modifications lui permettent de filtrer uniquement les blocs reconstruits jugés les plus significatifs. Pareillement, le paramètre Bs (*Boundary-strength*), qualifiant la frontière des blocs, est toujours mis à la valeur 1. En effet, de meilleurs résultats sont observés pour cette valeur.

5.3 Conditions expérimentales et résultats

Dans cette partie, les résultats expérimentaux seront présentés afin de valider notre schéma de compression. Dans toutes les expérimentations, seule la luminance des images est prise en compte pour l'évaluation des performances. De même, nous ne considérons que les images WZ qui possèdent une cadence de 15 fps. Les images K sont ici considérées parfaites au décodeur (afin de faciliter la comparaison avec les codecs de la littérature). Pour cela, le paramètre de quantification des images K, QPI, est toujours mis à 0. Enfin nous fixerons le paramètre de contrôle de la zone morte du quantificateur f à 1/2, correspondant ainsi aux plus faibles performances[5] atteintes en modifiant celui-ci.

5.3.1 Paramétrage de la valeur D_C

Conséquence de la nature de notre système, les plans de bits en relation avec les coefficients AC quantifiés peuvent avoir une longueur qui n'est pas un multiple de

5. Bien qu'ils ne soient pas présentés dans cette thèse, les résultats d'expérimentations portant sur la modification du paramètre f ont montré que les performances générales du codec croissent légèrement à mesure que f décroit ; la valeur nulle conduisant aux meilleurs résultats.

L (la longueur d'entrée du code de canal). Afin de circonvenir à ce problème, nous avons adopté la solution proposée à la section 5.2.1 pour concaténer tous les plans de bits. Les bits restants après la division entière par L sont manipulés différemment. Selon leur longueur D, deux scénarii peuvent s'appliquer. Si D est inférieure où égale à $D_C = 20$ (20 est considérée comme une valeur de bon compromis), ces bits restants seront transmis au décodeur sans compression. Autrement, ils seront concaténés avec des 0 afin d'obtenir une longueur L puis seront compressés.

5.3.2 Génération de l'information latérale

Dans nos expériences, nous avons utilisé deux méthodes pour générer l'information latérale : (i) un algorithme d'interpolation d'image basée sur un principe de compensation de mouvement travaillant directement avec les objets internes à l'image (CRC-FRC : CRC-*Frame Rate Converter*). Celui-ci procure une conversion de débit d'image de haute qualité [64]. (ii) Un simple moyennage entre deux images K consécutives Y_{2k-1} et Y_{2k+1}.

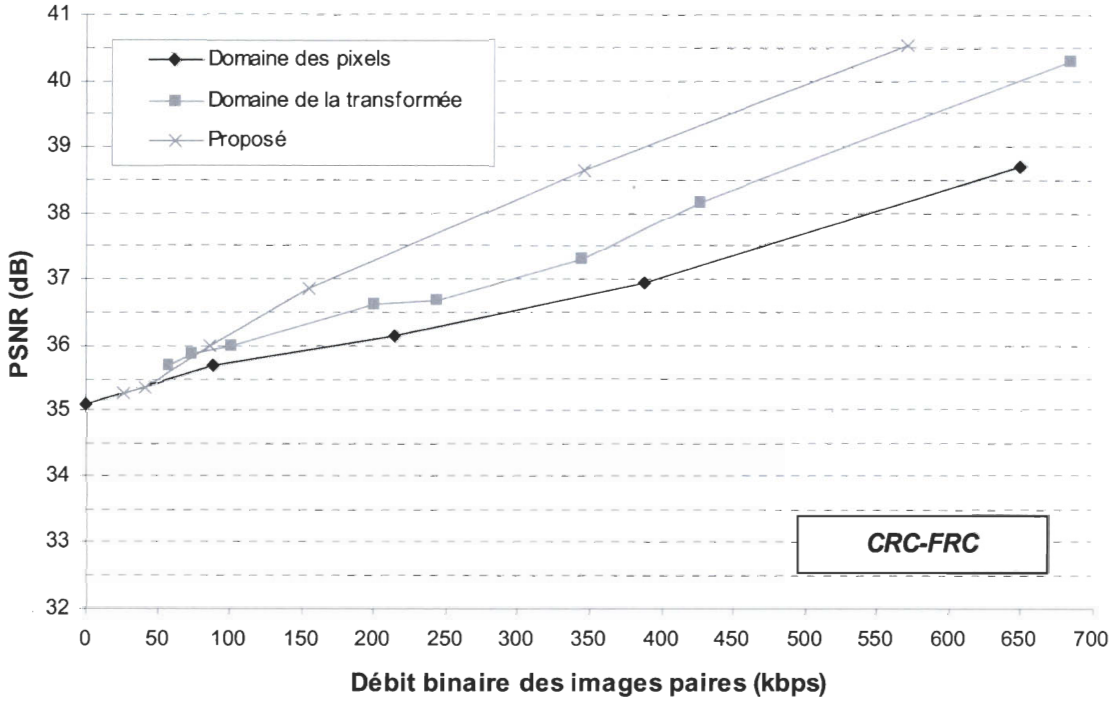
5.3.3 Résultats expérimentaux

Les résultats expérimentaux sont présentés, comme nous venons de le voir, avec deux types d'information latérale : l'une est générée par le CRC-FRC et l'autre par un simple moyennage. Les séquences vidéo tests sont la séquence *Foreman* et la séquence *Carphone*. Les deux possèdent une résolution QCIF. Elles sont échantillonnées à 30 fps et contiennent 101 images (soit 50 images WZ). Le tableau 5.2 expose le nombre P de coefficients transmis par bloc et ce pour différents paramètres de quantification QPW des images WZ.

TABLE 5.2 – Nombre de coefficients transmis par bloc et par valeur du paramètre QPW.

QPW	20	25	30	35	40	40
P	15	15	10	10	3	1

À des fins de comparaison, nous avons représenté les courbes de performance du domaine DCT et du domaine des pixels relatives aux articles [5] et [24] respectivement. La figure 5.8 concerne les courbes de débit-distorsion (PSNR vs. débit binaire) obtenues avec la séquence *Foreman* et la figure 5.9 présente les résultats obtenus avec la séquence *Carphone*. Des exemples d'images décodées sont également illustrés par les figures 5.10 et 5.11 pour les deux types d'information latérale. De plus, dans nos expérimentation, nous supposons que la valeur de la probabilité d'entrecroisement ϵ_b dans (5.12) est parfaitement connue au décodeur.



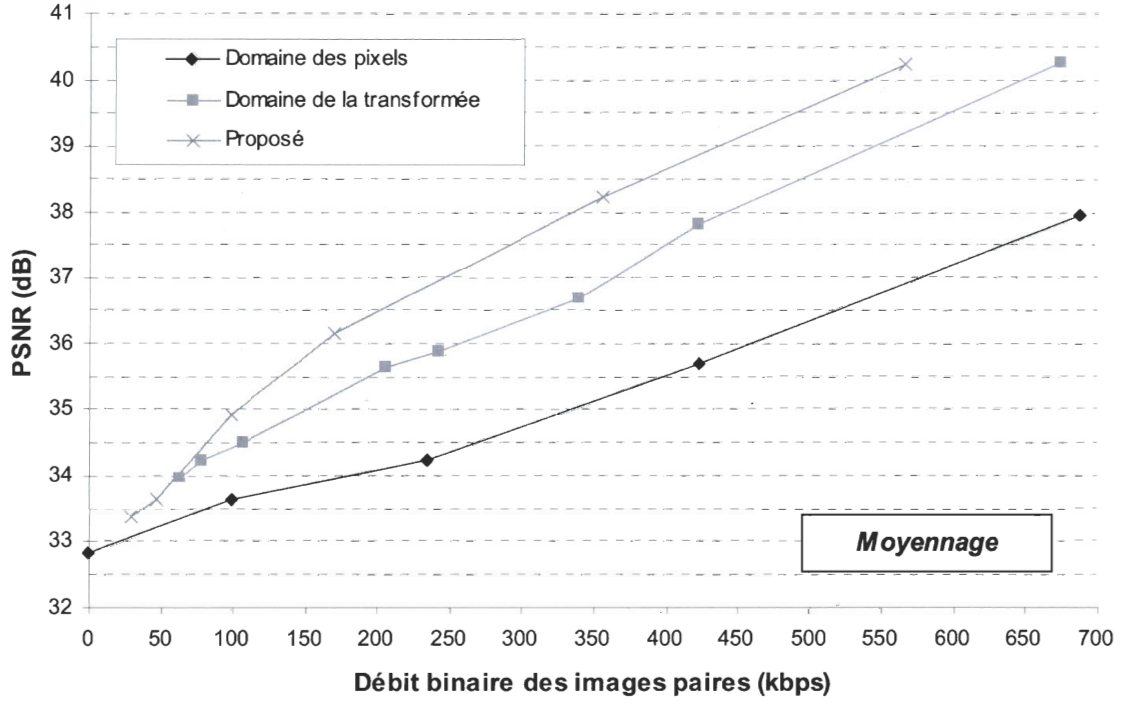


FIGURE 5.8 – Courbes de débit-distorsion pour la séquence vidéo *Foreman* (QCIF, $QPI = 0$, GOP : IW, images impaires WZ à 15 fps). *Haut :* information latérale générée avec le CRC-FRC. *Bas :* information latérale générée par moyennage de deux images K consécutives.

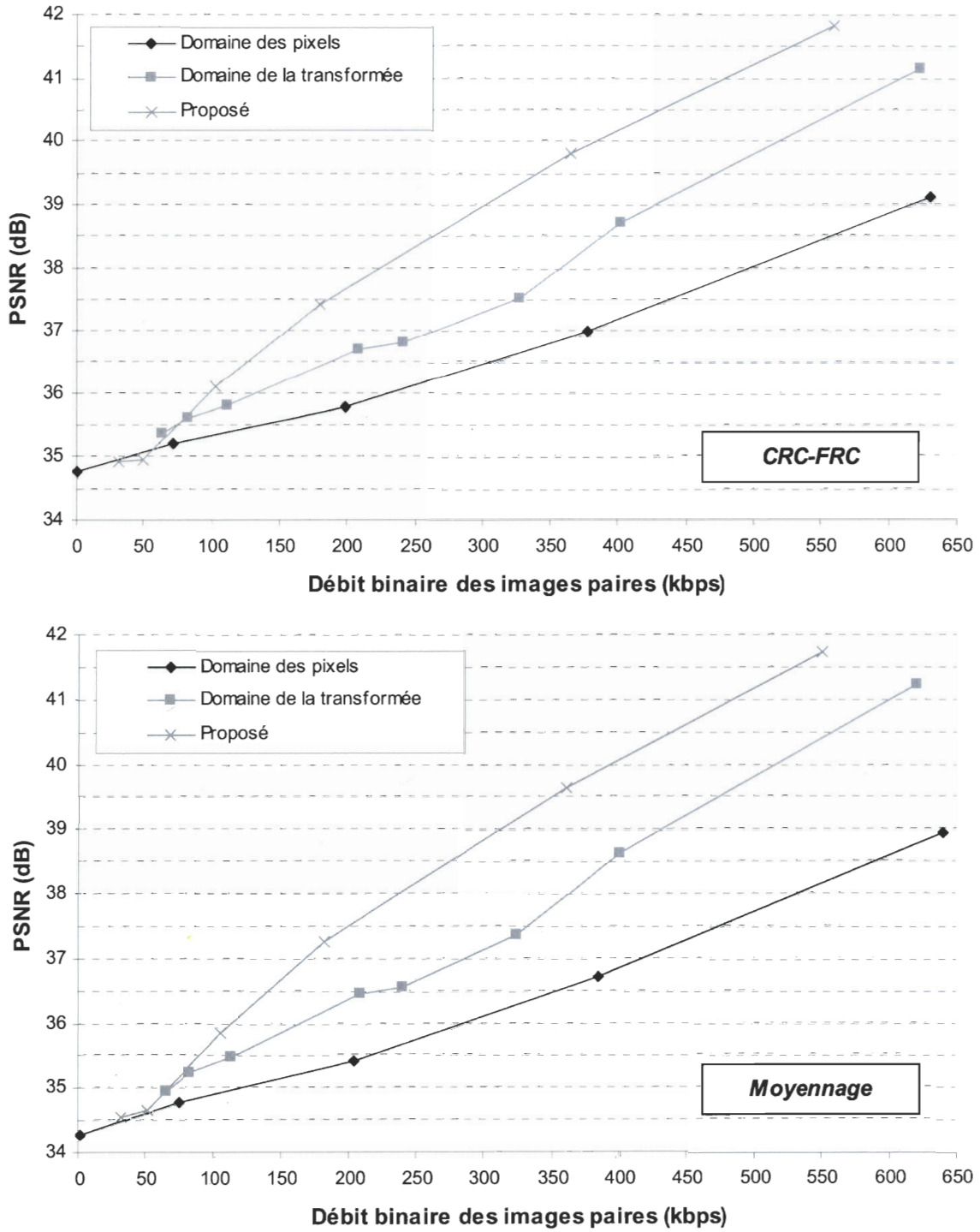


FIGURE 5.9 – Courbes de débit-distorsion pour la séquence vidéo *Carphone* (QCIF, $QPI = 0$, GOP : IW, images impaires WZ à 15 fps). *Haut :* information latérale générée avec le CRC-FRC. *Bas :* information latérale générée par moyennage de deux images K consécutives.

FIGURE 5.10 – Reconstruction de l'image 4 de la séquence *Foreman* (QCIF - 30fps) : $QPI = 0$, $QPW = 30$, $P = 10$. *Gauche* : information latérale générée avec le CRC-FRC (36.727 dB). *Droite* : image décodée (38.296 dB) : 0.37 bpp.

FIGURE 5.11 – Reconstruction de l'image 4 de la séquence *Foreman* (QCIF - 30fps) : $QPI = 0$, $QPW = 30$, $P = 10$. *Gauche* : information latérale générée par simple moyennage (32.007 dB). *Droite* : image décodée (38.206 dB) : 0.43 bpp.

Ces deux figures montrent que notre schéma de compression peut amener à une amélioration jusqu'à plus de 1.5 dB à hauts débits en comparaison des systèmes de codage vidéo distribué conventionnels.

5.4 Conclusion

Nous avons présenté dans ce chapitre un nouveau schéma de codage vidéo distribué. Il offre une réduction significative du débit binaire en ne transmettant que les coefficients AC quantifiés des blocs les plus significatifs. Pour cela, l'encodeur transmet en premier lieu l'ensemble des coefficients DC quantifiés au décodeur. Le décodeur compare alors ces derniers à ceux de l'information latérale pour déterminer si ils sont semblables. Le résultat de cette comparaison, transmise à l'encodeur par une voie de retour, permet à l'encodeur de savoir au final si les blocs sont suffisamment significatifs en terme d'information apportée.

Chapitre 6

Représentation binaire adaptative

Dans les dernières années, la communauté du codage vidéo a montré un intérêt croissant envers le codage vidéo distribué. Il est aujourd'hui devenu le nouveau paradigme de compression vidéo où le processus d'encodage doit être simple. Ceci permet l'avènement d'un foisonnement d'applications où la complexité de calcul et où la capacité de mémoire dans l'encodeur est limitée. Dans ce chapitre, nous allons traiter d'une nouvelle méthode de représentation binaire de l'information visuelle dans le contexte du codage vidéo distribué. Cette méthode est une représentation adaptative qui permet une réduction de la taille des messages issus de l'image source sans aucune perte d'information. Elle permet une transmission binaire efficace à travers un canal sans perte et produit une réduction significative du débit binaire global tout en maintenant une faible complexité d'encodage. Les résultats expérimentaux ont montré une augmentation significative des performances comparativement aux représentations binaires plus classiques telles que le codage binaire naturel et le codage Gray.

6.1 Rappels sur le codage vidéo distribué

Le codage vidéo distribué diffère du codage vidéo conventionnel dans le sens que chaque image d'une séquence vidéo est encodée séparément et indépendamment des autres. La corrélation temporelle de deux images successives est exploitée uniquement au décodeur par l'intermédiaire de l'information latérale. De là, un décodage conjoint est nécessaire pour décompresser l'image source reçue. Les systèmes de codage vidéo distribué reposent paradoxalement sur des codes correcteurs d'erreurs dans le but de compresser l'information sans pertes, et ceci dans une manière distribuée [8]. L'information latérale peut alors être perçue comme une version bruitée de l'image source. En conséquence, et de par d'utilisation des codes correcteurs d'erreurs, le processus de décodage peut s'interpréter comme une correction de l'information latérale.

Dans les systèmes de codage vidéo distribué, la corrélation existante entre l'image source et l'information latérale est un paramètre très important. En effet, celui-ci est

étroitement lié au débit binaire global et donc des performances du système. Une faible corrélation signifie généralement que plus de bits (provenant de l'encodeur) seront nécessaires afin de corriger l'information latérale. Différents moyens existent pour augmenter cette corrélation. L'utilisation d'une transformation (DCT, ondelettes,...) pour exploiter la corrélation spatiale et l'amélioration de la qualité de l'information latérale au décodeur sont les voies les plus évidentes. Une autre technique existe cependant, elle consiste en la modification de la représentation binaire naturelle des pixels de l'image par une représentation binaire Gray [44] (cf. Annexe C).

Nous avons également vu dans le chapitre précédent qu'une réduction de la taille des messages de source par sélection de l'information la plus significative est une voie prometteuse pour réduire le débit binaire. Pour cela, l'utilisation d'une voie de retour (du décodeur vers l'encodeur) est indispensable afin d'aider le processus d'encodage à compresser l'information la plus significative dans l'image source.

Dans ce chapitre, nous proposons une méthode qui permet de combiner à la fois une réduction de la taille des messages en sélectionnant l'information la plus significative et une augmentation de la corrélation entre la source et l'information latérale. Cette méthode requiert également l'utilisation d'une voie de retour ainsi que d'un procédé d'estimation de l'erreur résiduelle (existante entre l'image source et l'information latérale) dans le décodeur.

6.2 Principe de la méthode

Réduire la taille de l'information à transmettre en sélectionnant l'information la plus significative peut avoir deux grands désavantages : (i) l'information transmise deviendra *critique* dans le sens qu'elle contiendra beaucoup plus de différences avec l'information latérale. Une reconstruction erronée peut alors avoir des conséquences plus importantes sur la qualité de la séquence vidéo décodée. (ii) La probabilité d'entrecroisement ϵ du canal binaire symétrique (cf. figure 3.9) entre l'image source (point de vue du codec de Slepian-Wolf) et l'information latérale en sera augmentée, réduisant par la même occasion les performances de compression des codes de canal. En effet, une grande probabilité d'entrecroisement entraînera une faible performance de compression puisque plus d'erreurs seront à corriger. Cependant, même si les performances de compression seront réduites, nous allons voir que le gain obtenu sur le débit binaire reste avantageux.

6.2.1 Énoncé du problème

Aujourd'hui, la majorité des systèmes de codage vidéo distribué sont basés sur une extraction des plans de bits afin d'atteindre de meilleures performances. La réponse des codes de canal (i.e. codes correcteurs d'erreurs) dépend de la probabilité d'erreur (i.e. probabilité d'entrecroisement). De là, les messages provenant des bits les plus signifi-

catifs (correspondant aux faibles valeurs de ϵ) conduiront à une meilleure compression. Prenons le point de vue du codec de Slepian-Wolf et appelons X l'image source et Y l'information latérale. Avec une représentation binaire naturelle, un problème persiste. Par exemple, quelquefois, même si les valeurs de deux pixels co-situés dans X et Y ne diffèrent que de 1, leur représentation binaire sera différente sur la plupart des bits :

$$X : (63)_{10} = (00111111)_2,$$
$$Y : (64)_{10} = (01000000)_2.$$

Ici la distance de Hamming vaut $d_H(X, Y) = 7$. En conséquence, dans ce genre de cas, la plupart des bits auront besoin d'être corrigés au décodeur. Ainsi, la corrélation entre chaque plan de bits de la source et de l'information latérale sera réduite et induira de la même manière une augmentation du débit binaire.

Néanmoins, une solution existe déjà : c'est la représentation Gray exposée dans [44] et en annexe C. Cette représentation binaire permet qu'un seul bit soit différent entre deux valeurs consécutives. De cette façon, la corrélation entre X et Y sera augmentée :

$$X : (63)_{10} = (00100000)_2,$$
$$Y : (64)_{10} = (01100000)_2.$$

Dans cet exemple, la distance de Hamming vaut $d_H(X, Y) = 1$.

Cette représentation permet d'obtenir une diminution du débit binaire mais ne permet pas d'éviter la transmission des bits les plus significatifs (et donc ceux qui procurent les plus importantes erreurs de reconstruction lors d'un mauvais décodage). En d'autres termes, l'encodeur doit tout de même transmettre au décodeur les plans de bits les plus significatifs.

6.2.2 Solution proposée

La solution suivante permet une augmentation de la corrélation entre X et Y et permet d'éviter la transmission des bits les plus significatifs. Son principe repose sur la concentration des différences existantes entre X et Y sur les bits les moins significatifs. Le fondement du codage vidéo distribué interdit à l'encodeur de connaitre la valeur des éléments de l'information latérale, la méthode proposée permet alors de transmettre au décodeur un *résumé* de la valeur absolue des éléments plutôt que la valeur relative (i.e. différence avec l'information latérale). Cette méthode est utilisée directement dans le codec de Slepian-Wolf des systèmes de codage vidéo distribué. La figure 6.1 montre le schéma bloc du codeur/décodeur de Slepian-Wolf utilisant la méthode proposée.

La méthode proposée trouve son origine d'un constat très simple : fournir une représentation binaire séparée pour chaque chiffre constituant la valeur de chaque composante d'un pixel (habituellement codée sur 8 bits). De cette manière, et en reprenant

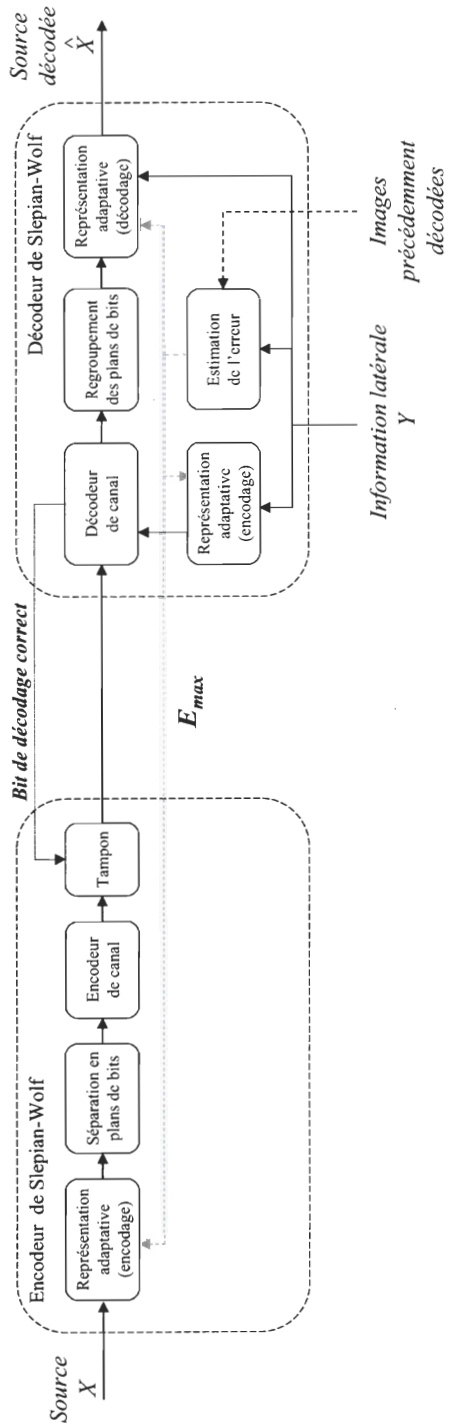


FIGURE 6.1 – Schéma bloc proposé de l'encodeur et décodeur de Slepian-Wolf utilisant la représentation adaptative.

le même exemple que ci-dessus, les valeurs peuvent s'écrire comme :

$$X : (6)_{10} \mid (3)_{10} \equiv (0110)_2 \mid (0011)_2,$$
$$Y : (6)_{10} \mid (4)_{10} \equiv (0110)_2 \mid (0100)_2.$$

Puisque le décodeur connaît le chiffre des dizaines avec l'information latérale, il devient inutile de le transmettre. En conséquence, une représentation sur 4 bits peut être une solution. En pratique cependant, cela devient plus complexe. Par exemple, si $X = 63$ et $Y = 54$ alors la reconstruction de X ne serait pas correcte et donnerait 53. Dans le premier exemple donné, nous avons en fait supposé indirectement que la valeur maximale de la différence entre X et Y n'excédait pas le chiffre des dizaines et donc une erreur maximale absolue de 3 (afin d'éviter les ambiguïtés de reconstruction).

La clé de voûte de la méthode est ici la valeur maximale de l'erreur : $E_{max} = \max |X - Y|$. Dans ce cas, le décodeur doit estimer ce paramètre et l'envoyer à l'encodeur [1]. La distribution des erreurs entre X et Y est supposée suivre une loi de Laplace [24] :

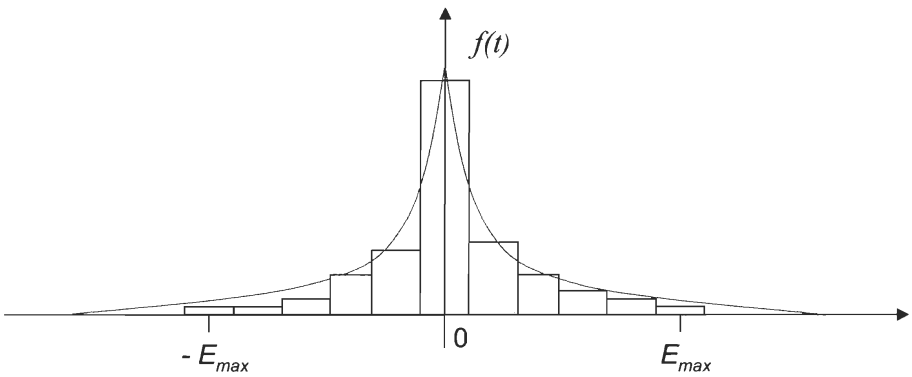


FIGURE 6.2 – Distribution Laplacienne des différences entre X et Y. E_{max} représente la valeur maximale absolue de l'erreur.

Encodage

Dans le cas général, le chiffre le moins significatif de la source doit être en relation directe avec l'erreur maximale absolue E_{max}. De ce fait, cette relation permet une augmentation ou une diminution directe du nombre de plans de bits à transmettre.

L'idée générale est ici de représenter chaque valeur de la source X dans une base B dépendante de E_{max}. La relation entre ces deux paramètres est la suivante :

$$B = 2\,(E_{max} + 1)\,. \tag{6.1}$$

1. E_{max} pourra également, dans des applications futures, directement être estimée dans l'encodeur.

Ainsi B représente le nombre maximal de différences entre X et Y et $\lceil \log_2(B) \rceil$ correspond au nombre maximal de bits à transmettre. E_{max} doit cependant être calculé au décodeur et transmis à l'encodeur. Chaque valeur de X est ensuite convertie dans la base B puis seul le chiffre le moins significatif (chiffre des unités) est pris en compte. Finalement, cela est équivalent à calculer le reste après la division entière de X par B :

$$X_{tr} = X \bmod B. \tag{6.2}$$

Cette méthode permet de transformer un espace d'entiers relatifs (fini ou non) en un espace à B valeurs entières positives ou nulles. En d'autres termes, la plage de valeurs initiales est *découpée* en un nombre (entier ou non) de plages comportant des valeurs comprises entre 0 et $B - 1$. Il est par ailleurs nécessaire de connaître l'information latérale Y afin de pouvoir associer chaque élément de l'espace d'arrivée contenant les valeurs de X_{tr} à chaque élément de l'espace initial associé à X.

Avant d'être transmis au décodeur, X_{tr} est converti en représentation binaire Gray pour maintenir un maximum de corrélation en deux valeurs successives. Le tableau 6.1 montre un exemple de conversion des valeurs autour de $X = 63$ avec $E_{max} = 3$.

TABLE 6.1 – Exemple de conversion de la valeur $X = 63$ avec $E_{max} = 3$. D'une manière équivalente, différentes valeurs autour de X sont exposées. Elles correspondent à différentes valeurs possibles de Y où aucune ambiguïté n'existe pour la reconstruction.

	$X - E_{max}$			X			$X + E_{max}$
$(X)_{10}$	60	61	62	63	64	65	66
$(X)_8$	74	75	76	77	100	101	102
X_{tr}	4	5	6	7	0	1	2
Bin. repr.	100	101	110	111	000	001	010
Gray repr.	110	111	101	100	000	001	011

Décodage

Le décodage de X_{tr} au décodeur se fait simplement. Premièrement, comme le décodeur connaît Y, E_{max} et donc Y_{tr}, les bits de parités reçus depuis l'encodeur corrigent la version binaire Gray de Y_{tr}. Le résultat de cette correction correspond à la version binaire Gray de X_{tr}. Une fois la conversion dans le système décimal effectuée, la quantité q suivante est calculée :

$$q = (Y_{tr} - X_{tr} + E_{max}) \bmod B. \tag{6.3}$$

Le décodage de X_{tr} se fait alors ainsi :

$$X = Y + E_{max} - q. \tag{6.4}$$

Remarques : (i) Cette méthode de représentation binaire fonctionne également pour les valeurs négatives de X. De cette façon, on évite toute transmission du bit de signe. (ii) Connaissant Y, la valeur $(Y \pm (E_{max} + 1)) \mod B$ correspondrait à la valeur de X_{tr} délimitant la zone d'absence d'ambiguïté de reconstruction. On peut voir en effet avec l'exemple du tableau 6.1 qu'avec $X_{tr} = 3$ et $Y = 63$ le décodeur devrait choisir entre $X = 67$ ou $X = 59$.

Démonstration

$$
\begin{aligned}
q &= (Y_{tr} - X_{tr} + E_{max}) \mod B && (6.5)\\
&= (Y - k' \cdot B - (X - k \cdot B) + E_{max}) \mod B && (6.6)\\
&= (Y - X + B \cdot (k - k') + E_{max}) \mod B && (6.7)\\
&= (Y - X + E_{max}) \mod B. && (6.8)
\end{aligned}
$$

où k et k' sont deux entiers. Nous savons aussi que $|Y - X| \leq E_{max}$, et donc que $0 \leq Y - X + E_{max} < B$, ainsi :

$$q = Y - X + E_{max}. \tag{6.9}$$

De cette façon, nous pouvons écrire :

$$
\begin{aligned}
Y + E_{max} - q &= Y + E_{max} - (Y - X + E_{max}) && (6.10)\\
&= X. && (6.11)
\end{aligned}
$$

6.2.3 Représentation *sub-bit*

Nous savons que $0 \leq X_{tr} \leq B$ et de cela, il découle qu'une représentation immédiate serait de coder chaque valeur de X_{tr} avec un nombre de bits équivalent à $N = \lceil \log_2 (B) \rceil$. En outre, la théorie nous indique qu'il est possible de trouver une représentation sur $\log_2 (B)$ bits, ce qui assez difficile à concevoir en pratique. L'idéal serait de trouver une transformation permettant de diminuer le nombre de bits tout en conservant une corrélation entre X_{tr} et Y_{tr} suffisamment élevée afin de ne pas diminuer la performance des codes de canal.

Une solution simple fut trouvée. Il est en effet possible de représenter certaines valeurs de X_{tr} avec un nombre moins important de bits, c'est-à-dire avec $N - 1$ bits. Puisque X_{tr} est codé selon la représentation Gray, nous pouvons facilement constater que les derniers $N - 1$ bits d'un mot de code peuvent correspondre aux $N - 1$ bits d'un autre mot de code seulement si X_{tr} est supérieur à une valeur seuil. Ceci provient du fait que le codage de Gray est un codage binaire réfléchi. Ce seuil S peut s'exprimer de la façon suivante :

$$S = 2^N - B - 1. \tag{6.12}$$

TABLE 6.2 – Représentation des valeurs de la source dans la base $B = 12$. Il est aisé de voir que les 4 premiers mots de codes Gray ont un radical de 3 bits différent des autres ($S = 3$).

X	X_{tr}	Rep. bin.	Rep. Gray
0	0	0 0 0 0	0 0 0 0
1	1	0 0 0 1	0 0 0 1
2	2	0 0 1 0	0 0 1 1
3	3	0 0 1 1	0 0 1 0
4	4	0 1 0 0	0 1 1 0
5	5	0 1 0 1	0 1 1 1
6	6	0 1 1 0	0 1 0 1
7	7	0 1 1 1	0 1 0 0
8	8	1 0 0 0	1 1 0 0
9	9	1 0 0 1	1 1 0 1
10	10	1 0 1 0	1 1 1 1
11	11	1 0 1 1	1 1 1 0
12	0	0 0 0 0	0 0 0 0
...	...	...	...

En conséquence, toutes les valeurs de X_{tr} inférieures ou égales à S peuvent se représenter sur $N-1$ bits tandis que toutes les valeurs supérieures à S nécessitent N bits. Le tableau 6.2 fournit un exemple où le seuil S est mis en évidence.

La longueur moyenne des mot-codes $\bar{\ell}$ dépend directement de la distribution des valeurs de la source X. Pour une image naturelle, cette distribution dépend du contenu de l'image. Nous allons voir des exemples de calcul de $\bar{\ell}$ dans la partie suivante.

Remarque : Au décodeur, le traitement devra s'effectuer des plans de bits les moins significatifs vers les plans de bits les plus significatifs. Ceci permet de retrouver les valeurs où le bit le plus significatif a été transmis.

6.3 Étude d'exemples

On reprend ici le système de codage vidéo distribué guidé par le coefficient DC et celui présenté par Brites et *al.* [5] (cf. § 3.3.2). Comme ces systèmes fonctionnent dans le domaine transformé (par utilisation de la DCT), il a été nécessaire d'estimer le paramètre E_{max} pour chaque bande de fréquence, et par conséquence B puis $\bar{\ell}$. Dans ce qui va suivre, nous appellerons par la variable $X^Q_{u,v}$ l'ensemble des coefficients quantifiés de la transformée associés à la bande de fréquence (u,v) comme définie au chapitre 5. $X^Q_{u,v}$ correspond en fait à X dans le schéma de la figure 6.1 puisque ce dernier se place du point de vue du codec de Slepian-Wolf. Par extension, nous écrirons les termes $Y^Q_{u,v}$ et $X^{u,v}_{tr}$ afin de garder l'association à la bande de fréquence (u,v).

Les figures 6.3 et 6.4 représentent : (i) la distribution des valeurs de $X^Q_{u,v}$, (ii) la distribution de la différence $X^Q_{u,v} - Y^Q_{u,v}$ ainsi que (iii) la distribution de $X^{u,v}_{tr}$ pour la bande DC et la troisième bande AC de l'image 2 de la séquence *Foreman.* Ces résultats concernent uniquement le schéma de compression vidéo distribuée guidée par le coefficient DC. L'information latérale est générée par l'algorithme CRC-FRC. Pour rappel, et comme il l'est montré à travers la figure 6.4, les bandes de fréquences AC $(u \neq 0, v \neq 0)$ contiennent des valeurs négatives.

Le tableau 6.3 fournit quant à lui des informations statistiques sur l'ensemble des bandes de fréquence pour l'image 2. Il permet entre autre de comparer la longueur moyenne des mot-codes $\bar{\ell}$ au nombre de plan de bits $R_{u,v}$ requis pour chaque bande de fréquence sans l'utilisation de la représentation adaptative. On peut ainsi aisément voir que $\bar{\ell}$ est toujours en deçà du nombre de plans de bits $R_{u,v}$. En ce sens, la représentation binaire adaptative propose une représentation plus efficace que celle originale.

Il peut arriver que la longueur moyenne des mots de codes $\bar{\ell}$ soit inférieure à la limite théorique $\log_2(B)$. Il y a deux raisons qui influencent ce phénomène :

1. Une image n'est pas une source d'information uniforme. La répartition des valeurs à coder n'est donc pas homogène et il est donc très possible qu'une majorité des valeurs soit en dessous du seuil S, nécessitant ainsi moins de bits. Ce phénomène est bien illustré dans les figures 6.3 et 6.4.

2. L'ensemble d'arrivée n'est pas distribué uniformément : le résultat de la division du nombre d'éléments de l'ensemble des valeurs à coder par B n'est pas entier.

Le tableau 6.4 montre un autre exemple. Cette fois en revanche le schéma de codage vidéo distribué utilisé est celui de Brites et *al.* (cf. § 3.3.2) avec la séquence *Carphone* (QCIF - 30 fps). Les données sont fournies pour l'image 2 et l'information latérale est générée par l'algorithme CRC-FRC. Grâce à ce tableau, il est aussi possible de voir que pour certains cas (notés en caractères gras) la longueur moyenne $\bar{\ell}$ peut être supérieure au nombre de plan de bits $R_{u,v}$ nécessaire sans la représentation binaire adaptative. Cela signifie juste que la méthode n'est pas optimale et qu'elle peut être améliorée. Une méthode simple serait de comparer la valeur de $\log_2(B)$ à $R_{u,v}$. Si $\log_2(B)$ est supérieur à $R_{u,v}$, la représentation binaire directe des valeurs de $X^Q_{u,v}$ semblerait un meilleur choix.

TABLE 6.3 – Tableau comparatif pour le système de compression guidé par le coefficient DC : $QPI = 0$, $QPW = 20$ et $P = 15$. La longueur moyenne des mot-codes $\bar{\ell}$ (en bits) après représentation adaptative est indiquée pour chaque bande de fréquence transmise. Les données sont fournies d'après l'image 2 de la séquence *Foreman* (QCIF - 30 fps) et l'information latérale est générée via l'algorithme CRC-FRC (36.86 dB).

Bande (u,v)	E_{max}	B	Nb. val.	Nb. val. $> S$	$\log_2(B)$	N	$\bar{\ell}$	$R_{u,v}$
(0,0)	9	20	1584	599	4.322	5	4.378	8
(0,1)	7	16	653	653	4.000	4	4.000	7
(1,0)	6	14	653	447	3.807	4	3.685	6
(2,0)	6	14	653	314	3.807	4	3.481	6
(1,1)	5	12	653	260	3.585	4	3.398	6
(0,2)	5	12	653	184	3.585	4	3.282	5
(0,3)	3	8	653	653	3.000	3	3.000	4
(1,2)	4	10	653	135	3.322	4	3.207	5
(2,1)	5	12	653	191	3.585	4	3.292	5
(3,0)	5	12	653	144	3.585	4	3.221	5
(3,1)	4	10	653	124	3.322	4	3.190	4
(2,2)	5	12	653	141	3.585	4	3.216	5
(1,3)	3	8	653	653	3.000	3	3.000	4
(2,3)	3	8	653	653	3.000	3	3.000	4
(3,2)	3	8	653	653	3.000	3	3.000	4

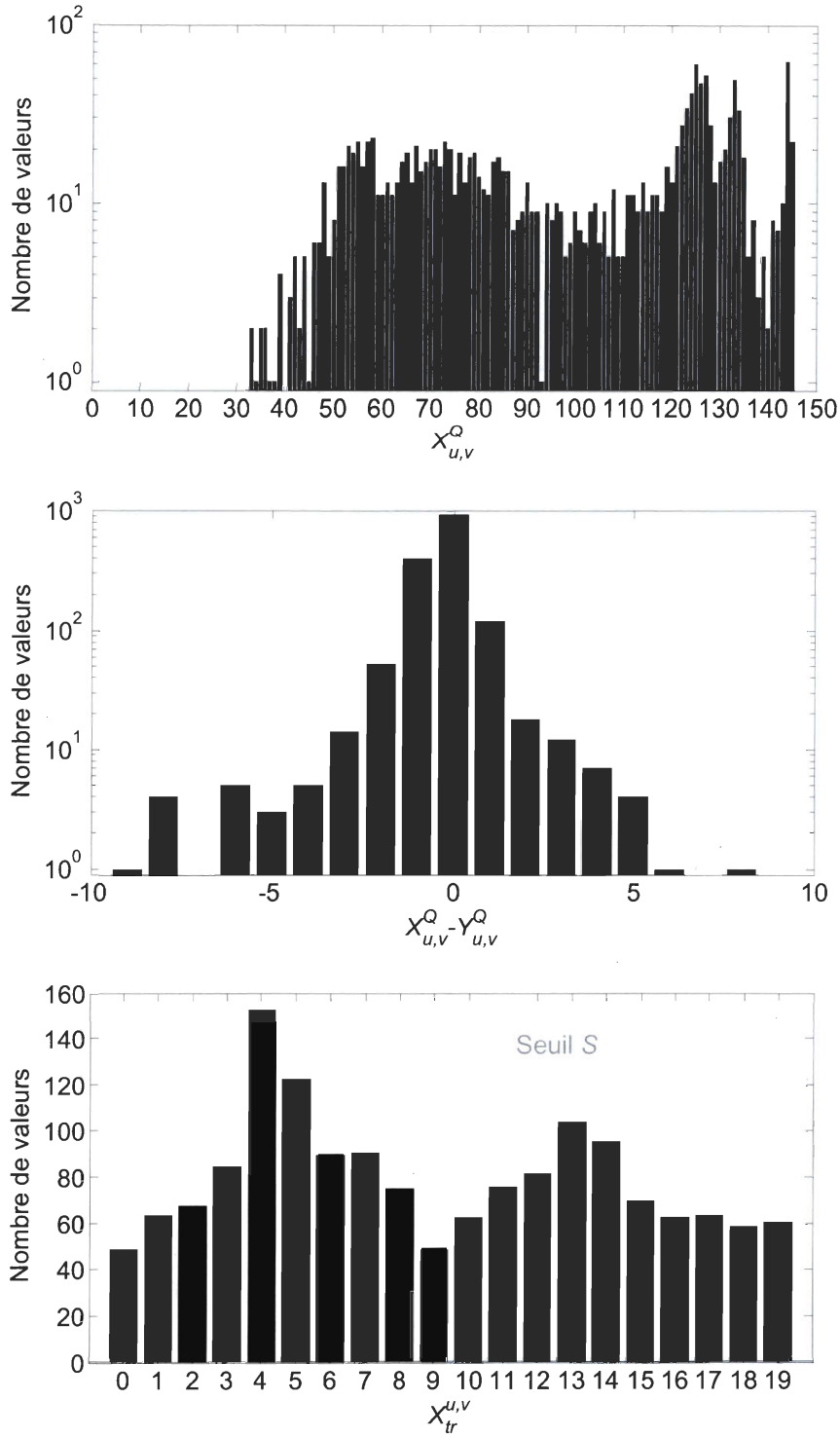


FIGURE 6.3 – Distribution des valeurs $X^Q_{u,v}$, des différences $X^Q_{u,v} - Y^Q_{u,v}$ et des valeurs $X^{u,v}_{tr}$ pour la bande DC $(0,0)$: *Foreman* (QCIF - 30 fps), image 2, $QPI = 0$, $QPW = 20$. L'information latérale est générée via l'algorithme CRC-FRC (36.86 dB).

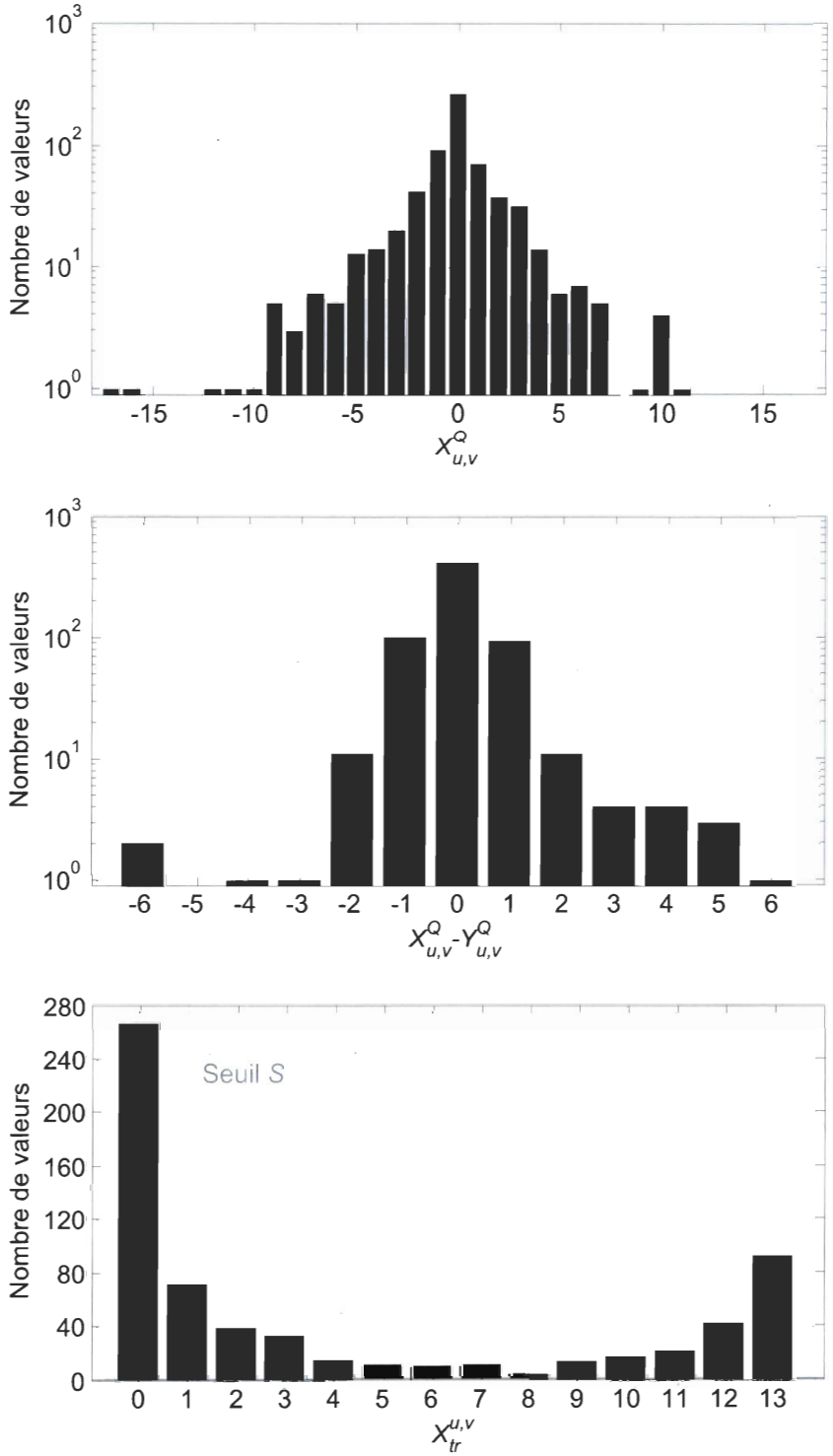


FIGURE 6.4 – Distribution des valeurs $X^Q_{u,v}$, des différences $X^Q_{u,v} - Y^Q_{u,v}$ et des valeurs $X^{u,v}_{tr}$ pour la bande AC $(2,0)$: *Foreman* (QCIF - 30 fps), image 2, $QPI = 0$, $QPW = 20$. L'information latérale est générée via l'algorithme CRC-FRC (36.86 dB).

TABLE 6.4 – Tableau comparatif pour le système du système de Brites et al. [5] avec la matrice de quantification M5 (cf. figure 3.11). La longueur moyenne des mot-codes $\bar{\ell}$ (en bits) après représentation adaptative est indiquée pour chaque bande de fréquence transmise. Les données sont fournies d'après l'image 2 de la séquence *Carphone* (QCIF - 30 fps) et l'information latérale est générée par l'algorithme CRC-FRC (31.99 dB).

Bande (u, v)	E_{max}	B	Nb. val.	Nb. val. $> S$	$\log_2(B)$	N	$\bar{\ell}$	$R_{u,v}$
(0,0)	5	12	1584	1071	3.585	4	3.676	5
(0,1)	3	8	1584	1584	3.000	3	3.000	4
(1,0)	2	6	1584	329	2.585	3	2.208	4
(2,0)	1	4	1584	1584	2.000	2	2.000	3
(1,1)	1	4	1584	1584	2.000	2	2.000	3
(0,2)	3	8	1584	1584	3.000	3	3.000	3
(0,3)	1	4	1584	1584	2.000	2	2.000	2
(1,2)	1	4	1584	1584	2.000	2	2.000	2
(2,1)	1	4	1584	1584	2.000	2	2.000	2
(3,0)	1	4	1584	1584	2.000	2	2.000	2
(3,1)	1	4	1584	1584	2.000	2	2.000	2
(2,2)	1	4	1584	1584	2.000	2	2.000	2
(1,3)	2	6	1584	12	2.585	3	2.008	2

6.4 Conditions expérimentales et résultats

Tout comme dans la partie précédente, la représentation proposée a été appliquée sur le système de codage vidéo distribué de référence décrit à la partie 3.3.2 ainsi que sur le schéma de compression guidé par le coefficient DC du chapitre 5. Les courbes de performances sont données dans le cas idéal, c'est-à-dire lorsque l'estimation de E_{max} est parfaite.

Les codes correcteurs d'erreurs utilisés ici sont les mêmes que ceux du système du chapitre 5, soient des codes LDPCA irréguliers et de longueur 396 bits. Leur décodage est assisté à l'aide d'un code détecteur d'erreurs CRC dont le polynôme générateur vaut $G(x) = x^8 + x^2 + x + 1$ (9 bits). Chaque message est donc associé à un CRC de 8 bits.

Dans les expérimentations, seule la luminance est prise en considération. Nous n'avons aussi tenu compte que du débit binaire des images WZ (15 fps). Enfin, les images K sont supposées parfaites au décodeur. Pour cela leur paramètre de quantification (QPI) a été mis à 0.

Les figures 6.5 et 6.7 donnent les courbes de débit-distorsion (PSNR vs. débit binaire) en rapport avec la séquence *Foreman* (QCIF - 30 fps - 101 images). Les figures 6.6 et 6.8 montrent quant à elle les courbes de débit distorsion obtenues avec la séquence *Carphone* comportant les mêmes caractéristiques. L'information latérale est générée ici à l'aide de l'algorithme CRC-FRC (cf. annexe D). Pour des raisons de comparaison, nous avons également illustré les courbes de performances des deux systèmes : l'un avec la représentation binaire naturelle et l'autre avec les codes Gray.

Ces figures montrent que la représentation binaire proposée peut procurer une amélioration de plus de 1 dB par rapport à l'utilisation des codes Gray.

Remarque : La taille de l'information nécessaire à l'encodeur pour déterminer l'erreur maximale absolue pour chaque bande de fréquence n'est pas prise en compte ici. L'augmentation du débit binaire est en effet très faible. Par exemple, si E_{max} est représenté sur 5 bits et sachant qu'il existe jusqu'à 16 bandes de fréquence transmises dans le système étudié, ceci équivaudrait à transmettre une information de longueur $16 \times 5 = 80$ bits par image WZ. L'augmentation maximale qui en résulterait serait alors de $80 \times 15/1000 = 1.2$ kbps. Bien évidemment, la taille de cette information s'adapte en fonction du nombre de coefficients à transmettre par bloc.

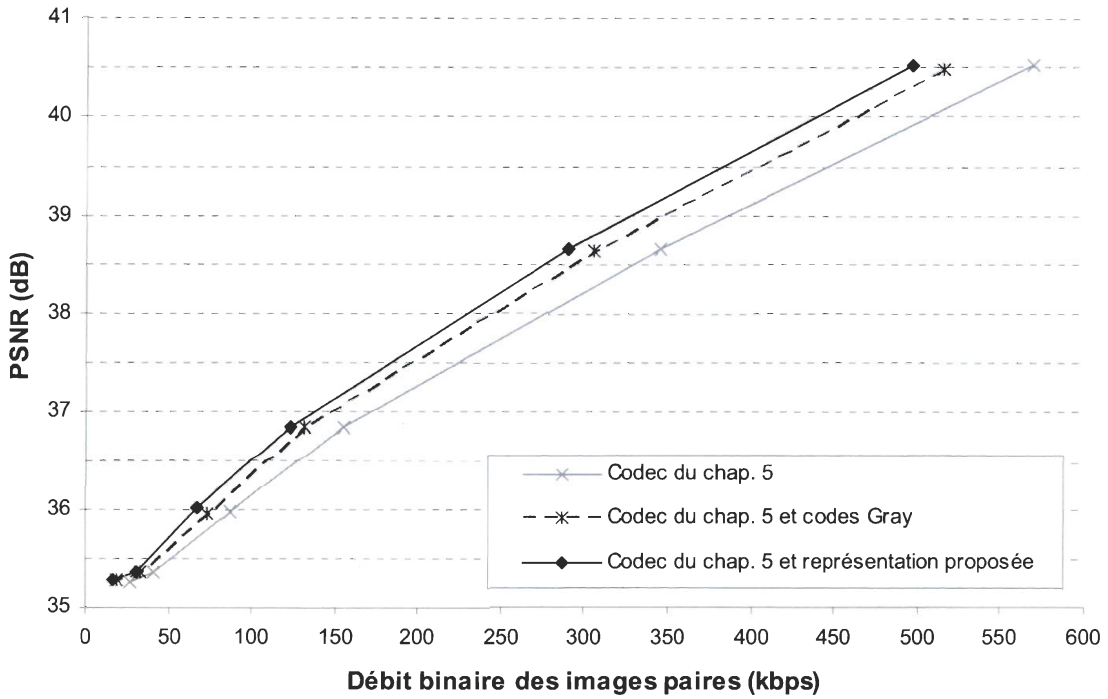


FIGURE 6.5 – Courbes de débit-distorsion pour le système de compression guidé par le coefficient DC avec la séquence *Foreman*.

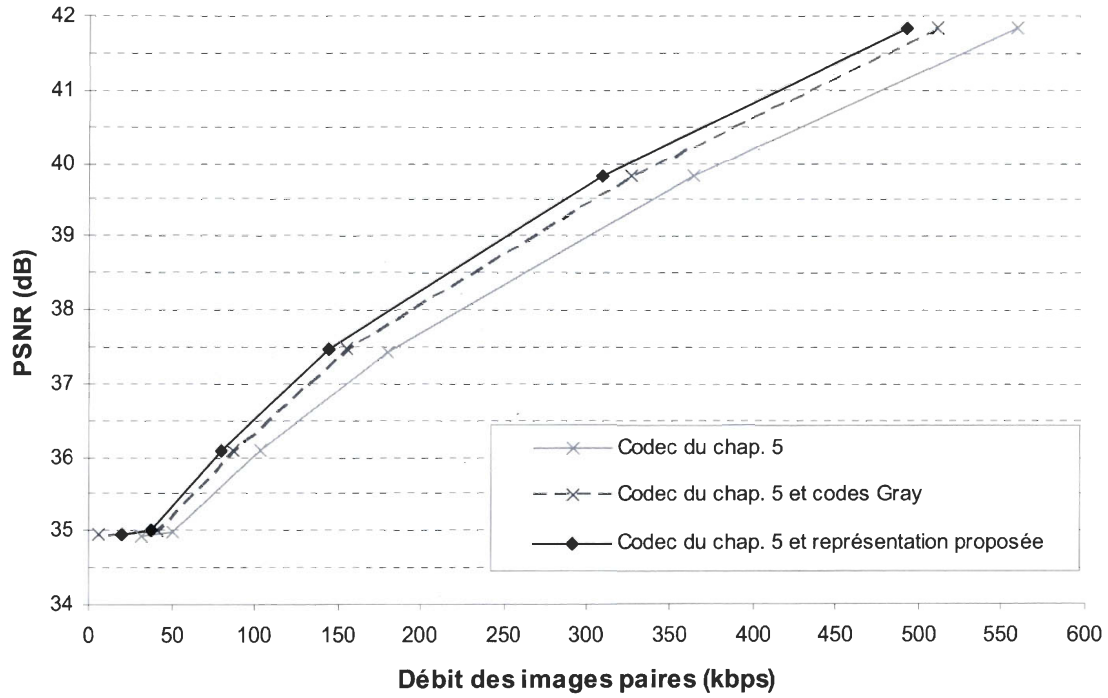


FIGURE 6.6 – Courbes de débit-distorsion pour le système de compression guidé par le coefficient DC avec la séquence *Carphone*.

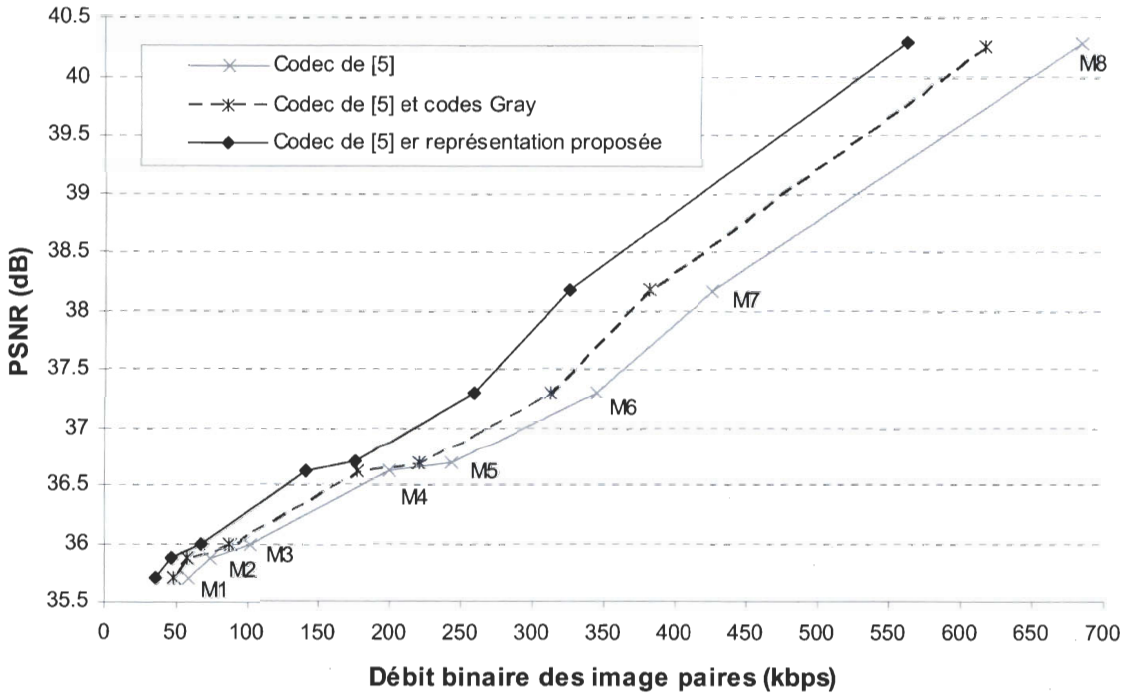


FIGURE 6.7 – Courbes de débit-distorsion du système de Brites et al. [5] avec la séquence *Foreman*.

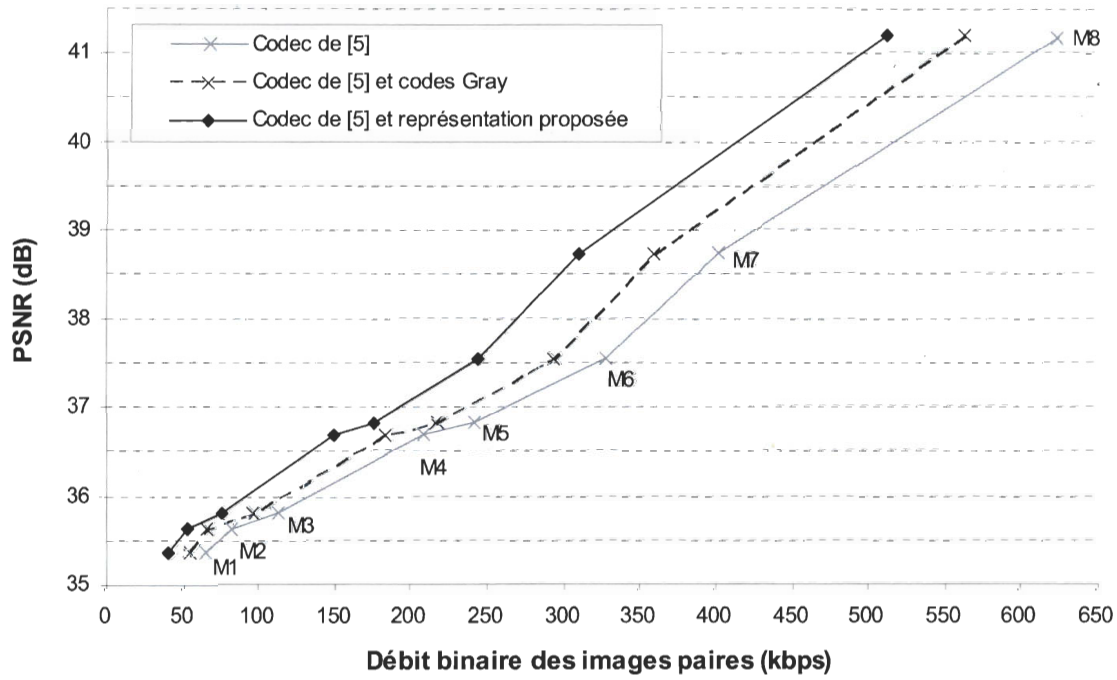


FIGURE 6.8 – Courbes de débit-distorsion du système de Brites et al. [5] avec la séquence *Carphone*.

6.5 Conclusion

Dans ce chapitre, une nouvelle méthode de représentation binaire (pour le codage vidéo distribué) a été présentée. Le principe de la méthode se résume à transmettre l'information (représentée par un codage Gray) relative aux différences entre la source et l'information latérale par l'intermédiaire d'une version *modulo* des valeurs de la source. De cette manière, l'information essentielle se concentre sur les bits de poids faibles tout en conservant une corrélation forte entre deux mots de code successifs. Une diminution efficace du débit binaire est ainsi obtenue.

L'estimation des différences, faite au décodeur, est transmise directement à l'encodeur par l'intermédiaire d'un canal de retour. Selon le type d'application, l'estimation peut envisager être faite à l'encodeur, évitant par la même occasion l'utilisation de ce canal de retour.

Chapitre 7

Conclusion générale

7.1 Introduction

Le codage vidéo distribué est le nouveau paradigme de la compression vidéo. Il permet un encodage simple et indépendant des images d'une séquence vidéo. Son principe de base repose sur les travaux de Slepian-Wolf et de Wyner-Ziv effectués dans les années 1970. Contrairement à un codage conventionnel, l'essentiel des calculs se situe au décodage, engendrant ainsi un renversement de l'asymétrie de complexité.

Les codes de canal utilisés dans les systèmes de codage vidéo distribué permettent de produire une compression de données indirecte. En effet, le décodage des images sources requiert une information latérale correspondant à une version estimée de l'image encodée. Cette information latérale contient des erreurs que les codes de canal vont corriger afin de retrouver l'image source. La version codée (ou compressée) de cette dernière est en fait un sous-ensemble des bits de parité (ou de syndrome) issus des codes de canal et qui furent transmis par l'encodeur.

7.2 Contributions

Dans cette thèse de doctorat, trois contributions originales ont été présentées. Elles concernent le domaine du codage vidéo. Deux d'entre elles sont néanmoins spécifiques au nouveau procédé de codage vidéo distribué où chaque image vidéo subit un encodage indépendant mais un décodage conjoint. Pareillement, ces trois contributions se complètent. Leurs effets peuvent s'additionner si elles sont intégrées au même codec.

Dans le chapitre 4, nous avons proposé un filtre basé sur une méthode d'apprentissage qui permet de réduire la distorsion et d'améliorer la qualité des images d'une séquence vidéo. Le filtre est entraîné avec un faible nombre d'images d'une scène vidéo et est appliqué sur toutes les images constituant la scène. Il peut facilement s'étendre à toutes normes de codage vidéo reposant sur le traitement par bloc de pixels telles que MPEG-2 et H.264/AVC. Les résultats préliminaires avec un codec H.264/AVC simplifié

sont prometteurs. Des gains significatifs peuvent être obtenus pour de fortes quantifications. Par ailleurs, l'augmentation du débit binaire due à la transmission des paramètres de filtrage au décodeur est faible. Ce principe de filtrage fut présenté à la conférence *Visual Communications and Image Processing* (VCIP) à San José, Californie, en janvier 2008 [65] et fut également l'objet du dépôt d'un brevet : U.S. Application #12,184,671, Canadian Application #2,738,465.

Dans le chapitre 5, un nouveau schéma de compression vidéo distribuée a été présenté. Celui-ci permet de réduire significativement le débit binaire des images WZ par utilisation d'une voie de retour informant l'encodeur à propos des blocs de l'information latérale qui sont les plus susceptibles d'être améliorés (ou corrigés). De par ce schéma, chaque image se sépare en deux couches. La première consiste en un regroupement de l'ensemble des coefficients DC quantifiés tandis que la seconde ne regroupe qu'un sous-ensemble des coefficients AC quantifiés. Les résultats expérimentaux ont prouvé que des gains significatifs peuvent être obtenus pour des débits binaires élevés. En outre, ce schéma de compression amène à une réduction du temps de décodage et réduit l'utilisation du canal de retour intrinsèque au codec de Slepian-Wolf. Finalement, l'utilisation de la transformée entière et du procédé de quantification propre au standard H.264/AVC [20] conduit à une réduction de complexité à l'encodeur tout en améliorant la qualité des images reconstruites au décodeur. Ce procédé de codage sera prochainement soumis pour présentation à une conférence. Un second article plus détaillé sera également soumis en vue de sa publication dans un journal scientifique.

Enfin, dans le chapitre 6, une nouvelle représentation binaire pour les systèmes de codage vidéo distribué a été présentée. C'est une représentation adaptative qui requiert l'utilisation d'un canal de retour afin d'informer l'encodeur de l'erreur maximale absolue existant entre l'image source et l'information latérale. Cette méthode offre une réduction significative du débit binaire puisqu'elle permet de n'envoyer que les bits les plus pertinents de chaque élément constituant l'image source (la valeur d'un pixel ou d'un coefficient de transformée). Les résultats expérimentaux ont montré que des gains significatifs peuvent être obtenus pour des débits élevés. Cette méthode n'a toutefois pas encore été proposée à une conférence où un journal mais une demande de brevet provisoire est en cours.

7.3 Suggestion de travaux futurs

Les trois contributions présentées dans cette thèse ne sont pas complètes. Plusieurs aspects sont sujets à amélioration et n'ont pas été étudiés. Des travaux futurs sont envisageables et concernent par exemple l'application du filtre par arbres de décision sur un codec H.264/AVC complet, prenant en compte l'intraprédiction, les macroblocs, la compensation de mouvement ainsi que les images P et B. Un design différent de la base de données peut également être possible. Un exemple serait de prendre en considération

plus de pixels autour de la limite de bloc. De même, des critères d'arrêts pour raffiner les arbres de décision peuvent être aussi étudiés. Un algorithme de compression sans perte plus adapté pour la transmission de la structure des arbres de décision et des coefficients de filtrage serait pareillement à considérer.

Concernant le schéma de codage guidé par le coefficient DC, des améliorations sont possibles pour ce qui est de la méthode classification des blocs. La méthode proposée dans cette thèse fonctionne sur un principe binaire (à savoir si la différence des coefficients DC de l'image source et l'information latérale est nulle ou non). Il est parfaitement concevable d'élaborer un peu plus cette classification.

Un autre projet pourrait porter sur la méthode de représentation adaptative. Il s'agirait ici d'utiliser des bases d'encodage calculées à partir de valeurs d'erreurs maximales absolues réduites $E_{red} < E_{max}$. L'erreur ainsi produite à la reconstruction serait contrôlée et permettrait une réduction du débit binaire. La correction des erreurs au décodage pourrait pour lors se faire à l'aide du contenu de l'image puisque les blocs corrompus seraient facilement détectables.

Annexe A

Codage de source et de canal

A.1 Codage de source

L'usage d'un canal de transmission coûte d'autant plus cher que le message à transmettre est long ; *coûter* devant s'entendre ici en un sens très général, celui d'exiger l'emploi de ressources limitées telles que le temps, la puissance ou la bande passante. Le codage de source peut donc, pour diminuer ce coût, viser à substituer un message aussi court que possible au message émis par la source mais fidèle également à l'information à transmettre.

On considère que les possibilités du canal sont utilisées de façon optimale lorsque les événements élémentaires délivrés par la source (les possibilités d'un symbole dans l'alphabet) sont de même probabilité d'apparition. Le principe du codage de source est donc de transformer le message engendré par une source discrète où les choix ne sont pas équiprobables et/ou pas indépendants, n'utilisant donc pas au mieux l'alphabet disponible, en un message où les symboles successifs sont équiprobables et indépendants (ceci dans le cas idéal). La transformation opérée devant bien entendu être réversible en ce sens que le message initial doit pouvoir être totalement reconstitué à partir du résultat du codage.

A.1.1 Classes de codes

La transmission de données depuis une source ne se fait pas sans certaines précautions quant au choix du code à employer. En effet, il est primordial que la séquence reçue doit être uniquement décodable pour pouvoir déterminer correctement l'information originelle. On dira alors qu'un code est uniquement déchiffrable si toute suite de mot-codes ne peut être interprétée (décodée) que d'une seule manière.

L'ensemble de tous les codes possibles $\{\mathcal{C}\}$ peut être subdivisé en plusieurs sous-ensembles [9] :

- Les ***codes préfixes*** pour lesquels aucun mot n'est le début d'un autre. Il est

clair qu'un code préfixe est uniquement déchiffrable puisqu'il peut être décodé sans aucune référence au futur. En effet, la fin d'un mot-code est reconnaissable immédiatement. Ex. : $\{0, 10, 110, 111\}$.

- Les ***codes uniquement déchiffrables*** pour lesquels un mot peut être le préfixe d'un autre. Son décodage requiert alors une référence au futur. Ex. : $\{0, 01, 011\}$.
- Les ***codes non-singuliers*** pour lesquels tous les mots de codes sont différents des autres. C'est-à-dire $\mathbf{c}_k \neq \mathbf{c}'_k$ si $k \neq k'$. Ex. : $\{0, 1, 00, 11\}$.
- ***Tous les codes possibles*** où les mot-codes sont formés sans aucune conditions. Il peut donc arriver que plusieurs mots de codes soient identiques. Ici les séquences de mot-codes ne sont cependant pas uniquement décodables.

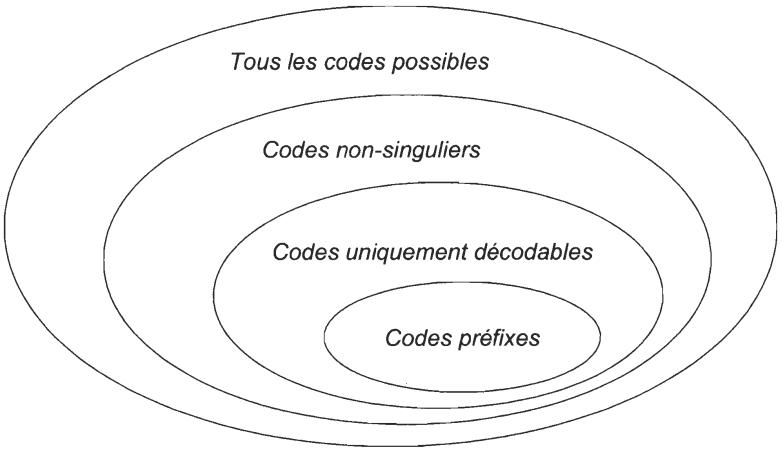


FIGURE A.1 – Les différentes classes de codes.

A.1.2 Séquences typiques et atypiques

Soit une source sans mémoire X où chaque symbole d'entrée possède une probabilité $p(x_i)$. Soit $\mathbf{x}$ un vecteur de longueur N dans l'espace d'épreuve $\mathcal{X}$ tel que $\mathbf{x} = (x_1, \ldots, x_N)$ avec $p(\mathbf{x}) = \prod_{i=1}^{N} p(x_i)$. Alors pour un nombre arbitrairement petit $\delta > 0$, l'ensemble des séquences typiques $\mathcal{T}_X(\delta)$ de longueur N se définit comme suit [9] :

$$\mathcal{T}_X(\delta) \equiv \left\{ \mathbf{x} : \left| -\frac{1}{N} \log_b p(\mathbf{x}) - H(X) \right| < \delta \right\}. \tag{A.1}$$

Les séquences atypiques est l'ensemble complémentaire tel que :

$$\mathcal{T}_X^c(\delta) \equiv \left\{ \mathbf{x} : \left| -\frac{1}{N} \log_b p(\mathbf{x}) - H(X) \right| \geq \delta \right\}. \tag{A.2}$$

A.1.3 Théorème de Shannon-McMillan

Prenons ici une source X d'alphabet de taille J, un nombre δ arbitrairement petit ainsi que des vecteurs de longueur suffisamment importante ($N \geq N_0$) tel que l'ensemble des J^N vecteurs possibles $\{\mathbf{x}\}$ peut être partitionné en ensemble de séquences typiques $\mathcal{T}_X$ ou atypiques $\mathcal{T}_X^c$. D'après ce que nous avons vu précédemment, les séquences typiques sont les plus probables, la probabilité qu'un vecteur $\mathbf{x}$ appartienne donc à l'ensemble des séquences atypiques est bornée de telle sorte que [10] :

$$P\left[\mathbf{x} \in \mathcal{T}_X^c\right] < \epsilon. \tag{A.3}$$

Par ailleurs, si $\mathbf{x}$ est un élément de l'ensemble des séquences typiques alors sa probabilité d'apparition est proche de $b^{-NH(X)}$:

$$b^{-N[H(X)+\delta]} < p(\mathbf{x}) < b^{-N[H(X)-\delta]}. \tag{A.4}$$

De plus, le nombre total de séquences typiques soit $\|\mathcal{T}_X\|$ est borné :

$$(1-\epsilon)b^{-N[H(X)-\delta]} < \|\mathcal{T}_X\| < b^{N[H(X)+\delta]}. \tag{A.5}$$

A.1.4 Théorème du codage de source

La redondance de l'information délivrée par la source s'élimine comme nous l'avons vu en codant de manière unique les différentes configurations émises par la source. Deux types de codes peuvent alors être utilisés : les codes à longueur variable et ceux à longueur fixe. Les premiers ont une longueur de mot qui varie en fonction de leur fréquence d'apparition. Un mot sera d'autant plus long que sa probabilité d'apparition sera petite. Ces codes sont souvent employés pour la compression de données.

Codes à longueur variable

Dans cette partie nous considérons un code $\mathcal{C}$ uniquement décodable dont les mots de code possèdent une longueur moyenne $L(\mathcal{C})$. Soit X une source d'information sans mémoire et d'alphabet K représentée par un code à longueur variable $\mathcal{C} = \{\mathbf{c}_1, \ldots, \mathbf{c}_K\}$ où chaque composant $c_{k,l}$ est pris dans un alphabet de taille J. Il est alors possible de démontrer que la longueur moyenne des mot-codes est bornée de la manière suivante [10, 9] :

$$\frac{H(X)}{\log_b J} \leq L(\mathcal{C}) < \frac{H(X)}{\log_b J} + 1, \tag{A.6}$$

avec $L(\mathcal{C}) = \sum_{k=1}^{K} p(x_k) l_k$.

1. Les mots de codes sont produits par une séquence unique de la source. Ils vérifient par ailleurs l'inégalité de Kraft : $\sum_{k=1}^{K} J^{l_k} \leq 1$ avec K le nombre total de mot-codes définis par un alphabet de taille J et l_k la longueur de ceux-ci.

Prenons maintenant une séquence de N variables aléatoires i.i.d telle que $\mathbf{X} = (X_1, \ldots, X_n, \ldots, X_N)$. L'entropie correspondante vaut alors :

$$H(\mathbf{X}) = NH(X_n) = NH(X). \tag{A.7}$$

De même la longueur moyenne du mot-code requis pour représenter une séquence de longueur N vaut :

$$L_N(\mathcal{C}) = NL(\mathcal{C}). \tag{A.8}$$

Cette dernière peut donc être bornée ainsi :

$$\frac{H(\mathbf{X})}{\log_b J} \leq L_N(\mathcal{C}) < \frac{H(\mathbf{X})}{\log_b J} + 1 \ . \tag{A.9}$$

Ce qui ramenée à la longueur d'un mot-code par symbole de source nous donne :

$$\frac{H(X)}{\log_b J} \leq L(\mathcal{C}) < \frac{H(X)}{\log_b J} + \frac{1}{N} \ . \tag{A.10}$$

Nous constatons alors que la longueur moyenne peut être arbitrairement proche de l'entropie par symbole $H(X)$ si de grandes séquences sont encodées. En effet, ceci permet de réduire le rapport 1/N.

Codes à longueur fixe

Le théorème du codage de source est le premier des trois théorèmes fondamentaux sur la théorie de l'information énoncés par Shannon en 1948. Ce théorème ne donne que les conditions nécessaires et suffisantes sur l'existence d'un codage de source efficace, il ne fournit aucun algorithme permettant d'atteindre les performances décrites. Considérons pour cela une source X d'entropie $H(X)$. Ce théorème stipule alors qu'il est possible de générer un code à longueur fixe dont les séquences de la source, de longueur N, sont encodées suivant des mot-codes[2] de longueur L et ce, avec une faible probabilité d'erreur $P_e < \epsilon$ au décodage [10]. Les conditions pour obtenir ce code sont énoncées ci-dessous :

$$L > N\frac{H(X)}{\log_b J} \ , \tag{A.11}$$

avec N suffisamment grand : $N > N_0$.

Ainsi il suffit d'attribuer à chaque suite typique un mot-code différent et de faire correspondre à toutes les suites atypiques un même mot-code. En recevant ce mot-code par défaut, on ne sera pas en mesure de retrouver la séquence de source sans risque d'erreur[3] mais cette éventualité ne se produira qu'avec une probabilité inférieure à ϵ, nombre fixé arbitrairement.

2. Les constituants des mot-codes proviennent d'un alphabet de taille J.

3. La probabilité d'erreur au décodage est aussi la probabilité d'obtenir un mot-code appartenant à l'ensemble des séquences atypiques.

En pratique cela se traduit par le fait que les séquences binaires fournies par la source peuvent être encodées avec une probabilité d'erreur aussi faible que souhaitée avec un rendement R bit/symbole tel que $R > H(X)$. Si $R < H(X)$, la probabilité d'erreur va s'accroître rapidement et ce, quelque soit la complexité des systèmes de codage et de décodage.

Le théorème fondamental du codage de source apporte une précision supplémentaire sur la distinction entre l'unité d'information binaire, appelée « *shannon* », et les symboles de l'alphabet binaire, que nous désignons par bit : ce n'est qu'après un codage de source idéal, c'est-à-dire atteignant la borne inférieure dans l'expression du théorème fondamental, que chacun des symboles binaires est porteur d'une quantité d'information égale à un shannon. Dans tous les autres cas, il porte une quantité d'information inférieure (s'il en porte une).

Remarque : Les codes à longueur fixe correspondent plutôt à une compression de données avec perte d'information, négligeable si ϵ est très petit.

Réciproque du théorème du codage de source

Soient $\epsilon > 0$ et une source discrète sans mémoire d'entropie $H(X)$. Si R est un nombre positif vérifiant $R < H(X)/\log_b(J)$, alors pour tous les codes de rendement R encodant des séquences de source de longueur $N \geq N_0$, la probabilité P_e d'associer avec risque d'erreur, une séquence de source à un mot-code vérifie [10] :

$$P_e > 1 - \epsilon. \tag{A.12}$$

A.2 Codage de canal

Dans le cadre du codage de source, nous avons étudié comment utiliser les redondances d'une source pour diminuer son débit binaire tout en conservant sa quantité d'information. Dans cette partie en revanche, il s'agit de transmettre l'information d'une source X dans un canal de transmission au moyen d'un code qui ajoute de la redondance d'information afin de combattre les erreurs de transmission. Un canal est un dispositif physique (ou temporel) permettant d'acheminer un message entre deux points distants. La variable Y reçue comportera des différences avec la variable X initialement émise dues aux perturbations (bruit) agissant sur le support de transmission.

Néanmoins du point de vue de la théorie de l'information, les imperfections du canal peuvent être traduites en terme d'information qu'apporte la variable de sortie Y sur la variable d'entrée X.

4. Rendement donné par le rapport $R = \frac{L}{N}$. Il s'exprime en bit/symbole.

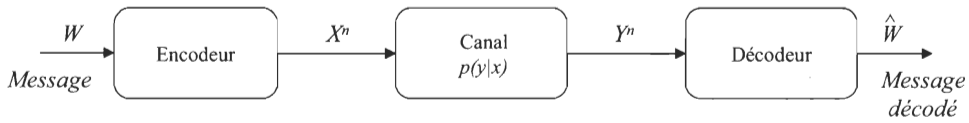


FIGURE A.2 – Système de communication.

A.2.1 Capacité d'un canal discret sans mémoire

Un canal discret est un support de transmission comportant un alphabet fini d'entrée $\mathcal{X}$, un alphabet fini de sortie $\mathcal{Y}$ ainsi qu'une matrice des probabilités de transition $p(y|x)$. Le canal est par ailleurs dit sans mémoire si le symbole de sortie ne dépend statistiquement que du symbole d'entrée le plus récent [9]. On définit la capacité d'un canal discret sans mémoire comme :

$$C = \max_{p(x)} I(X;Y), \tag{A.13}$$

où le maximum est considéré suivant toutes les probabilités d'entrée $p(x)$.

La capacité d'un canal correspond au débit maximal (en bit/canal) pour lequel l'information peut être envoyée avec une faible probabilité d'erreur. C'est aussi le maximum d'information qu'un canal peut transmettre sans erreur.

A.2.2 Capacité d'un canal discret avec mémoire

La capacité d'un canal discret avec mémoire est la limite du maximum de l'information mutuelle sur l'ensemble de tous les vecteurs d'entrée $\mathcal{S}_{\mathbf{p}(X_1,\ldots,X_N)}$ lorsque N tend vers l'infini [9] :

$$C = \lim_{N\to\infty} \max_{\mathcal{S}_{\mathbf{p}(X_1,\ldots,X_N)}} \frac{1}{N} I(X_1,\ldots,X_N;Y_1,\ldots,Y_N). \tag{A.14}$$

Après simplification la capacité devient :

$$C = \lim_{N\to\infty} \max_{\mathcal{S}_{\mathbf{p}(\mathbf{x})}} \left[H(\mathbf{X}) - H(\mathbf{E})\right], \tag{A.15}$$

où $\mathbf{E}$ est le vecteur d'erreur $(E_1,\ldots,E_N)$ fournit par le canal.

A.2.3 Séquences conjointement typiques

La démonstration du second théorème de Shannon (cf. § A.2.4) requiert la connaissance des séquences conjointement typiques. Pour un couple de sources sans mémoire (X,Y) suivant la distribution de probabilités $\{p(x,y)\}$ et d'entropie conjointe $H(X,Y)$,

l'ensemble des séquences typiques $\mathcal{T}_{XY}$ est l'ensemble des couples $(\mathbf{x}, \mathbf{y})$ de vecteurs de taille N vérifiant [10, 9] :

$$\mathcal{T}_{XY}(\delta) \equiv \left\{ (\mathbf{x}, \mathbf{y}) : \left| -\frac{1}{N} \log_b p(\mathbf{x}, \mathbf{y}) - H(X, Y) \right| < \delta \right\}. \tag{A.16}$$

De la même façon, le théorème de Shannon-Mac-Millan pour les couples de variables aléatoires s'énonce comme suit [10] :

$$P\left[(\mathbf{x}, \mathbf{y}) \in \mathcal{T}_{XY}^c\right] < \epsilon, \tag{A.17}$$

$$b^{-N[H(X,Y)+\delta]} < p(\mathbf{x}, \mathbf{y}) < b^{-N[H(X,Y)-\delta]}, \tag{A.18}$$

et :

$$(1-\epsilon) b^{-N[H(X,Y)-\delta]} < \|\mathcal{T}_{XY}\| < b^{N[H(X,Y)+\delta]}. \tag{A.19}$$

Pareillement si une séquence (x^n, y^n) est dans $\mathcal{T}_{XY}(\delta)$ avec x^n fixé on a :

$$b^{-N[H(Y|X)+2\delta]} < p(\mathbf{y}|\mathbf{x}) < b^{-N[H(Y|X)-2\delta]}. \tag{A.20}$$

A.2.4 Théorème du codage de canal

Le théorème du codage de canal est le second théorème fondamental énoncé par Claude Shannon. Ce théorème fournit la condition d'adéquation entre la source et le canal pour obtenir un taux d'erreur aussi faible que souhaité [10]. Ainsi pour $R < C$, il existe un certain code $\mathcal{C}$ de taille M et de longueur N (suffisamment grande i.e. $N \geq N_0$) tel que la probabilité d'erreur au décodage soit bornée par une valeur ϵ très faible :

$$P_e \leq \epsilon. \tag{A.21}$$

Ce théorème donne tout son sens à la notion de capacité que l'on peut interpréter comme la quantité maximum d'information qui peut être transmise sans erreur.

Réciproque du théorème du codage de canal

Le théorème réciproque fournit un minorant de la probabilité d'erreur lorsque la condition d'adéquation source canal n'est pas satisfaite. Soit un canal discret sans mémoire de capacité C sur lequel on transmet des mot-codes de longueur N avec un débit R. La probabilité d'erreur P_e satisfait alors la condition suivante [10] :

$$P_e \geq 1 - \frac{C}{R} - \frac{1}{NR}. \tag{A.22}$$

Remarque : Ce théorème s'applique lorsque le débit R de la source est plus grand que la capacité du canal C auquel on la connecte. On ne peut alors transmettre le contenu de la source avec une probabilité d'erreur inférieure à la borne fournie. Si on se place maintenant sous l'hypothèse suivant laquelle l'entropie de la source est inférieure à la capacité du canal, alors le deuxième théorème de Shannon nous permet d'affirmer que la transmission du contenu de la source peut s'effectuer dans les meilleures conditions.

Annexe B

Codes à faible densité de parité (LDPC)

B.1 Présentation

Les codes LDPC (*Low-Density Parity-Check codes*) ont été inventés par Gallager dans les années soixante. Ce n'est qu'à partir de 1995, soit après l'invention des turbo-codes, que leur *redécouverte* fût établie. Ces codes correcteurs d'erreurs utilisés pour le codage de canal sont tout aussi performants que les turbo-codes. Sous certaines conditions, ils dépassent même les performances de ces derniers [66]. Ils possèdent également un plancher d'erreur plus bas que ceux des turbo-codes. Toutefois, les meilleurs requièrent des séquences d'informations très longues souvent irréalisables en pratique. Le code utilisé dans [66] est le meilleur connu à ce jour ; c'est un code de rendement 1/2 possédant une longueur de 10^7 bits. Il permet d'approcher la limite de Shannon à 0.0045 dB pour un canal additif à bruit blanc gaussien. Par ailleurs le décodage des codes LDPC peut se réaliser de manière itérative tout comme pour les turbo-codes. Cependant l'algorithme est pour une itération donnée beaucoup moins complexe que celui exigé pour les turbo-codes. Le décodage peut par contre nécessiter un nombre d'itérations beaucoup plus important.

De plusieurs façons ils se distinguent par leurs avantages. Des codes tout rendement peuvent être créés simplement en spécifiant la forme de la matrice de parité $\mathbf{H}$ [67]. La difficulté majeure réside alors dans la création de cette matrice. De même, il est presque toujours possible de vérifier l'exactitude du mot de code. De par l'utilisation des contrôles de parité, il est très simple de détecter la présence d'erreurs et ce, en particulier pour les séquences très longues. Par ailleurs, l'utilisation des codes LDPC se fait librement depuis l'expiration de son brevet. L'aspect négatif veut que l'encodage soit de plus grande complexité en comparaison des codes turbo. Elle évolue de façon quadratique avec la dimension du code [67].

B.2 Généralités

Un code LDPC est un code en bloc linéaire (N, K) qui se caractérise par une matrice de contrôle de parité $\mathbf{H}$ à faible densité de 1 et de dimension $M \times N$ (avec $M = N - K$). Elle est souvent schématisée par l'intermédiaire d'un graphe bipartite (ou graphe de Tanner) représentant les m noeuds de contrôle de parité et les n noeuds de message. La figure B.1 expose un exemple d'un tel graphe. Un noeud de message j est connecté

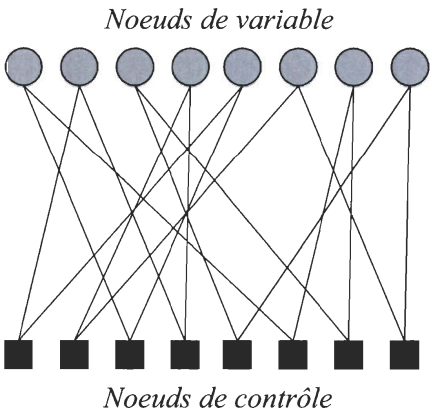


FIGURE B.1 – Graphe de Tanner d'un code LDPC.

au noeud de contrôle i seulement si l'élément (i, j) de la matrice $\mathbf{H}$ vaut 1. La matrice de parité $\mathbf{H}$ contient usuellement des 1 distribués aléatoirement et de façon clairsemée. Leur distribution (*density evolution*) le long des lignes et des colonnes est habituellement caractérisée par deux polynômes[1] (un pour chaque dimension de $\mathbf{H}$). De même la matrice de parité doit vérifier certaines conditions lors de sa création. Elle doit en effet ne contenir aucun cycle de faible périmètre (e.g. 4 ou 6) qui nuisent en fait à l'efficacité des codes. C'est ce qui rend d'ailleurs la création de la matrice si délicate. En règle générale plus le périmètre minimum des cycles est grand et plus les codes sont performants. Il faut aussi noter que des cycles de périmètres trop élevés indiquent une distance de Hamming très faible entre les mots de code. La probabilité de décoder alors un mot de code différent lors du décodage en est augmentée. Un compromis devra donc être pris en compte lors de la génération de $\mathbf{H}$.

Les codes LDPC présentant une répartition de 1 différente selon les lignes et selon les colonnes sont dits *irréguliers*. Ce sont les plus performants [11, 66]. Les lecteurs désireux d'avoir de plus amples informations concernant les codes LDPC et leurs créations peuvent se référer au livre de David J.C. MacKay [68].

1. Les puissances associées aux monômes constituant un polynôme correspondent au nombre de 1 présent dans la ligne (ou colonne) . Les poids des monômes par contre représentent les probabilités d'avoir ce nombre de 1 dans ladite ligne (ou colonne).

B.3 Encodage

L'encodage des messages d'information se réalise de la même manière que pour les codes en bloc. Pour ce faire, une fois la matrice $\mathbf{H}$ générée, il est nécessaire de calculer la matrice génératrice $\mathbf{G}$ (de dimension $N \times K$) telle que $\mathbf{HG} = \mathbf{0}$. Dans [67], on donne une méthode de résolution de ce système d'équations dans le cas où la matrice $\mathbf{H}$ possède une forme systématique. Cette résolution se fait généralement avec l'algorithme du pivot de Gauss sur le corps de Galois à deux éléments. Bien qu'il soit fréquent de rencontrer des matrices $\mathbf{H}$ non-systématiques, nous allons supposer dans la suite de cette partie que la matrice $\mathbf{H}$ est de forme systématique.

Une fois $\mathbf{G}$ déterminée, le mot de code $\mathbf{c}$ $(N \times 1)$ associé au message d'information $\mathbf{x}$ $(K \times 1)$ s'obtient aisément :

$$\mathbf{c} = \mathbf{Gx}. \tag{B.1}$$

B.4 Décodage

Dans cette section nous présentons l'algorithme *Somme-Produit*[2] dans sa version vraisemblance logarithmique. Son avantage par rapport à sa version usuelle est qu'elle évite la normalisation des calculs à chaque itération. Sa description, tirée de l'ouvrage de Todd K. Moon [67], nous permettra d'introduire les notions d'information intrinsèque et extrinsèque relatives au décodage des turbo-codes. L'algorithme est globalement très efficace et peu complexe. Il est en quelque sorte l'analogue de l'algorithme BCJR MAP des turbo-codes en agissant également par transmission d'information extrinsèque d'une itération à l'autre.

Tout débute en fait de l'expression du rapport de vraisemblance logarithmique où petit à petit des éléments relatifs à la structure du code sont introduits. L'expression de ce rapport est définie comme suit :

$$\lambda\left(c_n|\mathbf{r}\right) \triangleq \log \frac{\Pr\left(c_n = 1|\mathbf{r}\right)}{\Pr\left(c_n = 0|\mathbf{r}\right)} = \log \frac{\Pr\left(c_n = 1|r_n, \{r_i, i \neq n\}\right)}{\Pr\left(c_n = 0|r_n, \{r_i, i \neq n\}\right)}, \tag{B.2}$$

où $\mathbf{r}$ est le message reçu au décodeur par l'intermédiaire du canal de transmission et $\mathbf{c}$ est le mot de code. En appliquant le théorème de Bayes, le numérateur peut s'exprimer

2. SPA : *Sum-Product Algorithm.*

comme

$$\Pr\left(c_n = 1 | r_n, \{r_i, i \neq n\}\right) = \frac{\Pr\left(c_n = 1, r_n, \{r_i, i \neq n\}\right)}{\Pr\left(r_n, \{r_i, i \neq n\}\right)} \tag{B.3}$$

$$= \frac{\Pr\left(r_n | c_n = 1, \{r_i, i \neq n\}\right) \Pr\left(c_n = 1, \{r_i, i \neq n\}\right)}{\Pr\left(r_n | \{r_i, i \neq n\}\right) \Pr\left(\{r_i, i \neq n\}\right)} \tag{B.4}$$

$$= \frac{\Pr\left(r_n | c_n = 1\right) \Pr\left(c_n = 1, \{r_i, i \neq n\}\right)}{\Pr\left(r_n | \{r_i, i \neq n\}\right) \Pr\left(\{r_i, i \neq n\}\right)} \tag{B.5}$$

$$= \frac{\Pr\left(r_n | c_n = 1\right) \Pr\left(c_n = 1 | \{r_i, i \neq n\}\right)}{\Pr\left(r_n | \{r_i, i \neq n\}\right)}. \tag{B.6}$$

Pour obtenir cette expression, nous avons supposé l'indépendance entre r_n et $\{r_i, i \neq n\}$ connaissant c_n. La même démarche peut être fait pour le dénominateur. L'expression du rapport de log-vraisemblance peut alors se réécrire en une somme de deux informations :

$$\lambda\left(c_n | \mathbf{r}\right) = \underbrace{\log \frac{\Pr\left(r_n | c_n = 1\right)}{\Pr\left(r_n | c_n = 0\right)}}_{\text{Information intrinsèque } I_n^{int}} + \underbrace{\log \frac{\Pr\left(c_n = 1 | \{r_i, i \neq n\}\right)}{\Pr\left(c_n = 0 | \{r_i, i \neq n\}\right)}}_{\text{Information extrinsèque } I_n^{ext}}. \tag{B.7}$$

L'information intrinsèque est déterminée par la mesure explicite de r_n affectant c_n. Elle dépend intrinsèquement du canal de transmission :

$$I_n^{int} = \begin{cases} 2r_n/\sigma^2, & \text{pour un canal gaussien } (\sigma^2 = N_0/2), \\ (-1)^{r_n} \log\left(\frac{1-\epsilon}{\epsilon}\right), & \text{pour un canal binaire symétrique.} \end{cases} \tag{B.8}$$

L'information extrinsèque est déterminée quant à elle par les *autres* observations et la structure du code. En effet, il est possible de l'exprimer en termes de contrôle de parité. Soit $z_{m,n}$ les noeuds de contrôle de parité calculés utilisant le $m^{\text{ième}}$ noeud associé avec c_n et ce excepté pour c_n. Et donc :

$$z_{m,n} = \sum_{i \in \mathcal{N}_{m,n}} c_i, \tag{B.9}$$

où $\mathcal{N}_m = \{n : \mathbf{H}(m,n) = 1\}$ et $\mathcal{N}_{m,n} = \mathcal{N}_m \backslash n.\mathcal{N}_m$ correspond plus explicitement au vecteur constitué des numéros des colonnes présentant un 1 à la ligne m de la matrice $\mathbf{H}$. Pareillement l'expression $\mathcal{N}_{m,n}$ correspond au vecteur constitué des numéros des colonnes, cette fois autre que la colonne n, présentant un 1 à la ligne m de $\mathbf{H}$. On définira de la même façon les vecteurs $\mathcal{M}_n$ et $\mathcal{M}_{n,m}$ pour les noeuds de variable.

L'expression (B.9) signifie en fait que si $c_n = 1$ alors $z_{m,n} + c_n = 0$; en d'autres termes $z_{m,n} = 1$ pour tous les noeuds de variable $m \in \mathcal{M}_n$ où c_n est impliqué. De même si $c_n = 0$, $z_{m,n} = 0$ pour tous $m \in \mathcal{M}_n$. L'information extrinsèque peut donc se réécrire comme :

$$I_n^{ext} = \log \frac{\Pr\left(z_{m,n} = 1 \text{ pour tous } m \in \mathcal{M}_n | \{r_i, i \neq n\}\right)}{\Pr\left(z_{m,n} = 0 \text{ pour tous } m \in \mathcal{M}_n | \{r_i, i \neq n\}\right)}. \tag{B.10}$$

Nous faisons maintenant l'hypothèse que le graphe associé à la matrice $\mathbf{H}$ ne présente aucun cycle. Il est alors possible de dire que l'ensemble des bits associés à $z_{m,n}$ sont indépendants des bits associés avec $z_{m',n}$, pour $m' = m$. Ainsi nous avons :

$$I_n^{ext} = \log \frac{\prod_{m \in \mathcal{M}_n} \Pr\left(z_{m,n} = 1 | \{r_i, i \neq n\}\right)}{\prod_{m \in \mathcal{M}_n} \Pr\left(z_{m,n} = 0 | \{r_i, i \neq n\}\right)} \tag{B.11}$$

$$= \sum_{m \in \mathcal{M}_n} \log \frac{\Pr\left(z_{m,n} = 1 | \{r_i, i \neq n\}\right)}{\Pr\left(z_{m,n} = 0 | \{r_i, i \neq n\}\right)}. \tag{B.12}$$

Nous définissons alors :

$$\lambda\left(z_{m,n} | \{r_i, i \neq n\}\right) = \log \frac{\Pr\left(z_{m,n} = 1 | \{r_i, i \neq n\}\right)}{\Pr\left(z_{m,n} = 0 | \{r_i, i \neq n\}\right)}. \tag{B.13}$$

L'expression du rapport de vraisemblance devient donc :

$$\lambda\left(c_n | \mathbf{r}\right) = I_n^{int} + \sum_{m \in \mathcal{M}_n} \lambda\left(z_{m,n} | \{r_i, i \neq n\}\right). \tag{B.14}$$

Dans l'hypothèse que les noeuds de contrôle dans $z_{m,b}$ sont conditionnellement indépendants (cas où il n'existe pas de cycles), il est possible d'utiliser la règle de la tangente hyperbolique définie dans [67] et ainsi d'écrire :

$$\lambda\left(c_n | \mathbf{r}\right) = I_n^{int} - 2 \sum_{m \in \mathcal{M}_n} \tanh^{-1}\left(\prod_{j \in \mathcal{N}_{m,n}} \tanh\left(-\frac{\lambda\left(c_j | \{r_i, i \neq n\}\right)}{2}\right)\right). \tag{B.15}$$

Le calcul de $\lambda\left(c_n | \mathbf{r}\right)$ requiert donc la connaissance de $\lambda\left(c_j | \{r_i, i \neq n\}\right)$ qui représente le rapport de log-vraisemblance conditionnel aux bits permettant de connecter les noeuds de contrôles aux noeuds de variable n.

Définissons maintenant :

$$\eta_{m,n} = -2 \tanh^{-1}\left(\prod_{j \in \mathcal{N}_{m,n}} \tanh\left(-\frac{\lambda\left(c_j | \{r_i, i \neq n\}\right)}{2}\right)\right). \tag{B.16}$$

Ce terme correspond au *message* que le noeud de contrôle m envoie vers le noeud de variable n. L'équation (B.15) peut être alors écrite comme :

$$\lambda\left(c_n | \mathbf{r}\right) = I_n^{int} + \sum_{m \in \mathcal{M}_n} \eta_{m,n}. \tag{B.17}$$

Cette fois ce terme correspond au message que le noeud de variable n envoie vers le noeud de contrôle m.

Afin d'employer une méthode itérative de calcul entre les équations (B.16) et (B.17), il est nécessaire de retirer au message $\lambda\left(c_n | \mathbf{r}\right)$ la contribution qu'a reçu le noeud de

variable m de lui-même lors d'une précédente étape de calcul. Ceci est fait afin d'éviter qu'un biais existe dans $\eta_{m,n}$ puisque chaque message $\lambda(c_n|\mathbf{r})$ contient déjà l'information intrinsèque I_n^{int}. On peut écrire alors que :

$$\lambda(c_n|\{r_i, i \neq n\}) = \lambda(c_n|\mathbf{r}) - \eta_{m,n}. \tag{B.18}$$

L'algorithme peut être ainsi clairement décrit et nous allons dans ce qui suit en fournir un résumé.

Algorithme de décodage des codes LDPC

1. **Entrées :** $\mathbf{H}$, le message reçu $\mathbf{r}$, le nombre total d'itérations L ainsi que le(s) paramètre(s) du canal de transmission.
2. **Initialisation :** $\eta_{m,n}^{[0]} = 0$, $\lambda_n^{[0]} = I_n^{int}$ et $\ell = 1$.
3. **Mise à jour des noeuds de contrôle :** Pour chaque couple (m, n) où $\mathbf{H}(m, n) = 1$ on calcule :

$$\eta_{m,n}^{[\ell]} = -2\tanh^{-1}\left(\prod_{j\in\mathcal{N}_{m,n}} \tanh\left(-\frac{\lambda_j^{[\ell-1]} - \eta_{m,j}^{[\ell-1]}}{2}\right)\right). \tag{B.19}$$

4. **Mise à jour des noeuds de variable :** Pour chaque n on calcule :

$$\lambda_n^{[\ell]} = I_n^{int} + \sum_{m\in\mathcal{M}_n} \eta_{m,n}^{[\ell]}, \tag{B.20}$$

puis l'on effectue une décision telle que :

$$\hat{c}_n = \begin{cases} 1, & \text{si } \lambda_n^{[\ell]} > 0, \\ 0, & \text{si } \lambda_n^{[\ell]} \leq 0. \end{cases} \tag{B.21}$$

$\mathbf{H}\hat{\mathbf{c}} = \mathbf{0}$ signifie alors que le décodage s'est effectué correctement. Le code étant systématique, le message d'information décodé $\hat{\mathbf{x}}$ se retrouve aisément dans $\hat{\mathbf{c}}$. Autrement, si $\mathbf{H}\hat{\mathbf{c}} \neq \mathbf{0}$ et si le numéro de l'itération ℓ est inférieur au nombre total prédéterminé d'itérations L, on incrémente ℓ et on retourne au point 3. Dans le cas contraire, on arrête l'algorithme et on déclare un échec de décodage.

B.5 Application au codage distribué

Pour ce qui est de l'utilisation des codes LDPC avec le codage distribué, la littérature expose des résultats meilleurs qu'avec les codes turbo. Ils permettent quasiment d'atteindre la limite de Slepian-Wolf [11].

B.5.1 Système

Soient deux sources binaires corrélées X et Y, X représente alors la source d'information et Y l'information latérale disponible au décodeur. Le canal binaire symétrique existant entre les x_i et y_i des vecteurs de sources $\mathbf{x} = [x_1, x_2, \ldots, x_N]^T$ et $\mathbf{y} = [y_1, y_2, \ldots, y_N]^T$ possède une probabilité d'entrecroisement $\Pr[x_i \neq y_i] = \epsilon$ telle que $\epsilon < 0.5$.

Encodage

La compression dite *distribuée* s'opère par l'intermédiaire du calcul du syndrome $\mathbf{s}$ de longueur $(N - K)$. Celui-ci résulte de la multiplication du message de source $\mathbf{x}$ de longueur N par la matrice de parité $\mathbf{H}$ de dimension $M \times N$ (avec $M = N - K$) :

$$\mathbf{s} = \mathbf{H}\mathbf{x}. \tag{B.22}$$

Les syndromes ainsi calculés sont par la suite transmis au décodeur via un canal non-bruité. Un taux de compression de $N : (N - K)$ est ainsi atteint.

Dans le cas binaire, l'utilisation d'une matrice $\mathbf{H}$ fournit $2^{(N-K)}$ syndromes distincts. Ces syndromes sont donc associés à un ensemble de 2^N messages d'information, formant alors une application surjective puisqu'un élément de l'ensemble d'arrivée peut correspondre à plusieurs éléments dans l'ensemble de départ. C'est également une raison d'apparition d'erreurs de décodage.

Décodage

En codage vidéo distribué, il est nécessaire d'apporter une modification à l'algorithme présenté en B.4 afin de tenir compte de l'information procurée par les syndromes [11]. En effet le décodeur doit pouvoir corriger une séquence $\mathbf{y}$ de longueur N de l'information latérale à partir d'un syndrome $\mathbf{s}$ de longueur $(N - K)$. L'équation des noeuds de contrôle (B.19) doit donc être modifiée comme suit [11] :

$$\eta_{m,n}^{[\ell]} = -2\tanh^{-1}\left((2s_m - 1)\prod_{j\in\mathcal{N}_{m,n}} \tanh\left(-\frac{\lambda_j^{[\ell-1]} - \eta_{m,n}^{[\ell-1]}}{2}\right)\right). \tag{B.23}$$

Par ailleurs, nous sommes ici en présence d'un canal binaire symétrique de probabilité d'entrecroisement ϵ. L'information intrinsèque s'écrit dans ce cas :

$$I_n^{int} = (-1)^{y_n}\log\left(\frac{1-\epsilon}{\epsilon}\right). \tag{B.24}$$

Lors d'applications pratiques, le décodeur devra estimer ϵ.

Une décision sur le rapport de log-vraisemblance permet ensuite de calculer le message décodé $\hat{\mathbf{x}}$:

$$\hat{x}_n = \begin{cases} 1, & \text{si } \lambda_n^{[\ell]} > 0, \\ 0, & \text{si } \lambda_n^{[\ell]} \leq 0. \end{cases} \tag{B.25}$$

Finalement, la condition $\mathbf{H}\hat{\mathbf{x}} = \mathbf{s}$ permet de s'assurer d'un décodage correct.

Remarque importante : Lors de l'utilisation de cet algorithme de décodage, nous avons remarqué que la dernière itération ne correspondait pas nécessairement au message présentant le nombre le plus faible d'erreurs. En effet, il se pouvait que le nombre d'erreurs décroisse en fonction de l'itération puis remonte pour des valeurs plus élevées sans jamais redescendre de manière significative. Ceci nuit en fin de compte aux performances globales et peut se manifester par une oscillation prononcée des courbes de probabilités d'erreur. Un constat simple fut finalement de comparer la distance de Hamming entre le syndrome voulu (soit le syndrome reçu au décodeur) puis le syndrome calculé lors de l'itération avec le nombre d'erreurs résultant sur le message. Il fut ensuite possible de voir qu'une distance de Hamming minimum correspondait à un nombre faible d'erreurs dans le message d'information. Par ailleurs, l'écart entre ce nombre puis le nombre minimal d'erreurs que l'on pourrait obtenir au cours des itérations suivantes s'avère assez faible. Du moins il est beaucoup plus faible en comparaison des erreurs à la dernière itération (souvent de plusieurs ordres de grandeurs). La conséquence directe de cette observation fut en contrepartie de modifier l'algorithme de décodage pour qu'il accepte le message équivalent au syndrome ayant la plus faible distance de Hamming avec celui recherché. Nous pourrions voir aussi dans cela un critère pour mettre fin aux itérations. Effectivement, une corrélation semble exister entre la distance de Hamming des syndromes et le nombre d'erreurs sur le message décodé.

B.5.2 Codes LDPCA[3]

Un des points clés de la bonne performance du codage vidéo distribué est bien évidemment le type de code employé pour la compression. Contrairement aux turbo-codes, les codes LDPC peuvent s'approcher efficacement de la limite de Slepian-Wolf tout en maintenant un écart avec cette limite à peu près constant pour différents débits [11]. Par ailleurs, les codes LDPC ne procurent pas une modification aisée du débit en plus d'avoir des longueurs très importantes. Un débit différent nécessite obligatoirement une matrice de parité différente.

Une solution serait vraisemblablement le recours à des techniques de poinçonnage. Des recherches récentes ont justement établit que l'utilisation possible de cette technique

3. *LDPC Accumulate codes.*

sur ce type de code [69]. L'emploi de ce procédé pourrait en toute quiétude avoir son intérêt pour le codage vidéo distribué. Néanmoins Varodayan et *al.* ont préalablement proposés dans [6] que l'utilisation de codes concaténés, alors basés sur les codes LDPC, permettent d'atteindre des longueurs plus faibles que ceux déjà rencontrés et d'obtenir par la même occasion une adaptation rapide à des débits voulus.

Le poinçonnage des syndromes permet d'adapter le débit de transmission vers le décodeur. Cependant, si la matrice de parité ne tient pas compte de ce procédé, les performances seraient susceptibles d'être très faibles. En effet, le graphe de Tanner correspondant contiendrait après poinçonnage des noeuds de variable connectés uniquement à un seul noeud de contrôle voire même à aucun. Avec le décodage itératif, cette structure empêcherait la transmission de l'information. Les codes présentés dans ce qui suit permettent l'adaptation du débit tout en possédant des performances très correctes pour le codage distribué [6].

Les codes LDPCA consistent en la concaténation de codes LDPC puis d'un accumulateur. La figure B.2 expose ce principe. Les bits de source $(x_1, x_2, \ldots, x_N)$ sont additionnés (modulo 2) en accord avec la structure du graphe afin de générer le syndrome $\mathbf{s} = [s_1, s_2, \ldots, s_M]^T$. Les bits de celui-ci sont ensuite accumulés (modulo 2) afin de produire le syndrome accumulé $\mathbf{a} = [a_1, a_2, \ldots, a_M]^T$. L'encodeur peut alors transmettre ces bits séquentiellement vers le décodeur.

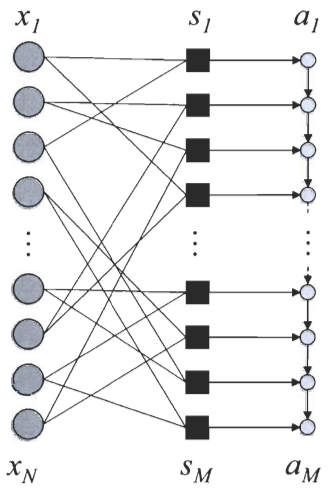


FIGURE B.2 – Principe de fonctionnement d'un code LDPCA [6].

L'adaptation du débit se fait par modification du graphe de décodage à chaque réception d'un bit additionnel du syndrome accumulé. Supposons premièrement que l'ensemble du syndrome accumulé soit reçu, la différence modulo 2 mène alors au syndrome $\mathbf{s}$. Le même algorithme de décodage qu'en B.5.1 s'emploie alors. La transmission intégrale du syndrome accumulé procure les mêmes performances que le codes LDPC utilisés sans accumulateur. L'ensemble du processus possède une représentation matri-

cielle équivalente :

$$\mathbf{a} = \mathbf{A}\mathbf{s} = \mathbf{A}\mathbf{H}\mathbf{x}. \tag{B.26}$$

où $\mathbf{A}$ est une matrice triangulaire inférieure $(N \times N)$ dont les termes valent 1. Elle représente l'accumulateur.

La modification du graphe de décodage est caractéristique de plus forts taux de compression. Prenons un exemple de 8 bits et considérons cette fois une compression de 0.5 bit/symb. qui peut par exemple correspondre à la transmission de l'ensemble des bits pairs du syndrome accumulé soit $(a_2, a_4, \ldots, a_8)$. La différence consécutive modulo 2 produit le syndrome *dé-accumulé* $\mathbf{s}^d = [s_1 + s_2, s_3 + s_4, \ldots, s_7 + s_8]^T$. Le graphe de Tanner correspondant maintient les connections avec l'ensemble des noeuds de variable, ce qui n'est pas le cas sans le recours de l'accumulateur (c.f. figure B.3).

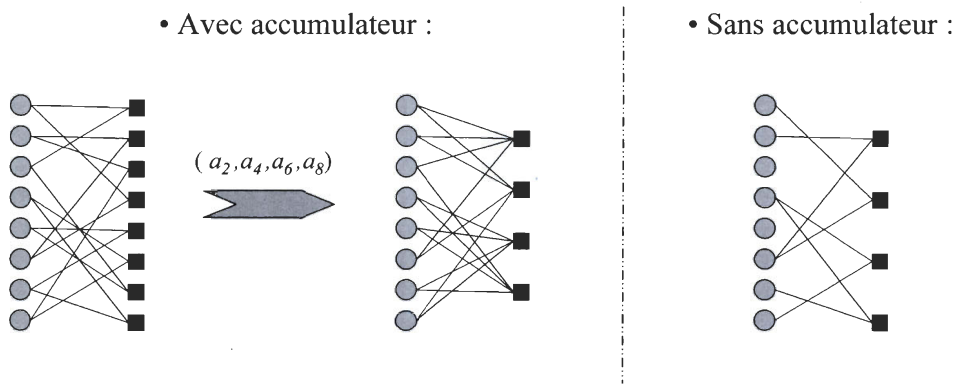


FIGURE B.3 – Syndrome réduit : avec et sans accumulateur [6].

Mathématiquement parlant, obtenir une compression $N : P$ où $P < N$ signifie que la matrice $\mathbf{H}$ se réduit en une matrice $\mathbf{H}_{red}$ de dimension $(N \times P)$ dont les lignes correspondent aux P bits de $\mathbf{s}^d$. Les lignes de la matrice $\mathbf{H}_{red}$ sont calculées de telle manière que les lignes correspondantes aux bits de syndrome constituant le syndrome accumulé reçu soient entremêlées pour conserver la position des 1. Par exemple, si le premier bit du syndrome désaccumulé $s_1^d = s_1 + s_2$, la première ligne de $\mathbf{H}_{red}$ résultera alors de l'opération logique OU (OR) appliquée entre la première et la seconde ligne de $\mathbf{H}$. Cette opération permet en effet de de préserver les liens entre les noeuds de contrôle et les noeuds de variable des lignes concernées.

Au décodage et si la matrice H était conservée telle quelle, cela équivaudrait à modifier l'équation des noeuds de contrôle (B.23) par :

$$\eta_{v,n}^{[\ell]} = -2\tanh^{-1}\left[\left(2\underbrace{\left(\sum_{i=u+1}^{v} s_i\right)}_{a_v - a_{u+1}} - 1\right)\prod_{i=u+1}^{v}\prod_{j\in\mathcal{N}_{i,n}}\tanh\left(-\frac{-\lambda_j^{[\ell-1]} - \eta_{v,n}^{[\ell-1]}}{2}\right)\right], \tag{B.27}$$

où $(g \in \mathcal{N}_{a,n}) \neq (h \in \mathcal{N}_{b,n})$ et $a, b \in [u+1, v]$. Il est en revanche plus aisé en pratique de modifier directement $\mathbf{H}$ et de garder le même algorithme de décodage.

Il a été démontré dans [6] que les codes LDPCA permettent d'atteindre des performances assez proches de la limite de Slepian-Wolf sans pour autant s'y écarter pour des débits plus élevés. Par ailleurs, les performances de ces codes ne diminuent pas significativement avec une longueur réduite. La figure B.4 expose un exemple des performances atteintes par des codes LDPCA réguliers et irréguliers avec une longueur $L = 396$. Pour obtenir ces courbes de performances, il est important de noter que seuls

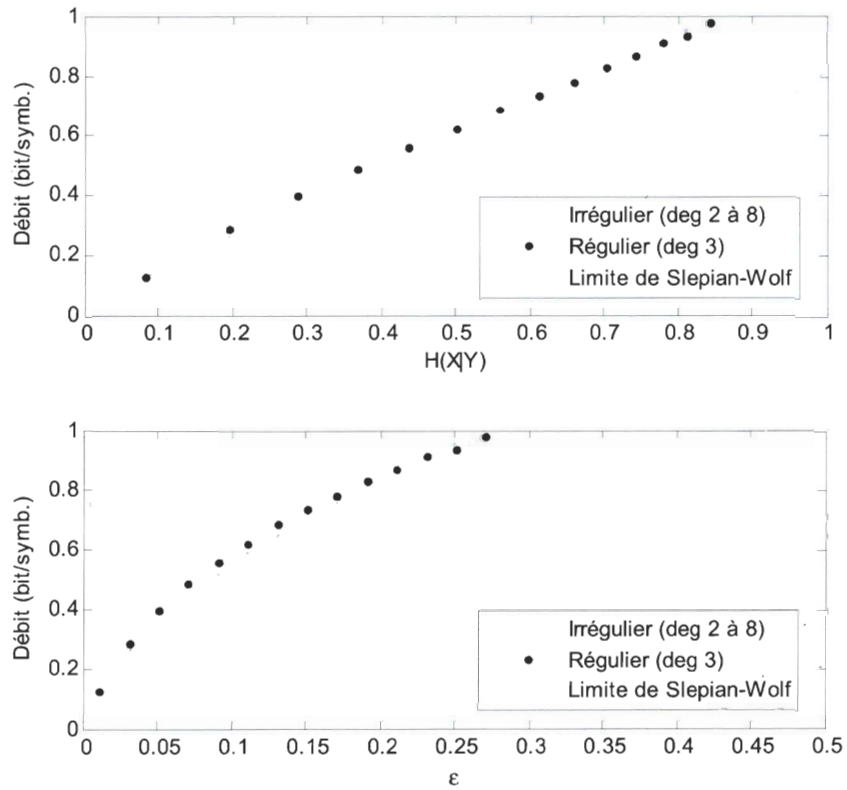


FIGURE B.4 – Performances des codes LDPCA de longueur $L = 396$. Débit d'encodage normalisé en fonction de l'entropie $H(X|Y) = -\epsilon \log_2(\epsilon) - (1-\epsilon)\log_2(1-\epsilon)$ puis en fonction de la probabilité d'entrecroisement ϵ du canal binaire symétrique.

les messages ayant pu être décodés sont pris en compte (pour une probabilité d'entrecroisement donnée). Des erreurs peuvent également apparaître pour des messages décodés correctement mais dans une proportion très faible. En revanche, lorsqu'un message n'a pas pu être décodé convenablement, l'encodeur devra transmettre des bits de syndrome supplémentaires. Par exemple, pour $\epsilon = 0.083$ (soit un débit binaire d'environ 0.5 bit/symbole), près de 57% des messages envoyés arrivent à être décodés. Le décodeur aura besoin en conséquence de plus de bits de syndrome pour les messages restants.

Annexe C

Codes Gray et codage vidéo distribué

De leur utilisation pour le codage vidéo distribué, il a été vérifié dans [44] que les codes Gray sont capables d'augmenter significativement la corrélation entre la source et l'information latérale, réduisant ainsi la probabilité d'entrecroisement. Ces codes sont utilisés à la fois pour la représentation binaire de la source et de l'information latérale. La propriété des codes Gray peut être utilisée afin d'augmenter significativement l'efficacité de la compression de données distribuées. En effet, les codes Gray présentent cette particularité d'avoir les mots de codes adjacents différenciés d'un seul bit. Les résultats expérimentaux de [44] ont démontré que ces codes permettent d'améliorer la qualité de l'image jusque 4 dB et ce pour un débit équivalent.

Soit un nombre binaire $b_1 b_2 b_3 \ldots b_n$ avec b_1 le bit de poids fort, le code Gray correspondant se construit itérativement comme suit :

$$g_i = b_{i-1} \oplus b_i. \tag{C.1}$$

Pareillement la conversion d'un code Gray en nombre binaire s'opère ainsi :

$$b_i = b_{i-1} \oplus g_i. \tag{C.2}$$

Dans les encodeurs de compression vidéo distribué, la représentation binaire est souvent utilisée pour la représentation en *plan de bits* (*bit plane*). Ceci veut dire que si un pixel est représenté sur N bits, il est possible de décomposer une image selon N plans[1], où le plan i correspondra aux $i^{\text{ièmes}}$ bits des pixels. Il convient de souligner que les plans associés aux bits de poids forts présenterons naturellement la plus faible probabilité d'entrecroisement. Le problème majeur lié à la représentation binaire naturelle est que dans le cas de deux nombres adjacents mais dont la distance de Hamming diffère de 1 (comme celle existante entre les nombres 3 et 4), il peut potentiellement se produire

1. Un plan aura la même largeur et hauteur que l'image, mais sera de profondeur 1.

plus d'erreurs et, par la même occasion, dégrader la corrélation entre la source et l'information latérale. La modification provoquée par les codes Gray peut significativement réduire ces erreurs et améliorer la compression [44].

Annexe D

CRC-FRC

L'algorithme CRC-FRC (*Communications Research Centre - Frame Rate Converter*) permet une conversion de haute qualité du nombre d'images par seconde d'une séquence vidéo vers un nombre plus élévé. Il génére des trames à partir d'une compensation de mouvement interpolatrice fonctionnant d'après les *objets* internes à la séquence vidéo [64]. En plus de cela, il dispose d'un mécanisme de mesure de fiabilité du mouvement. L'algorithme est ainsi en mesure d'estimer précisément les trajectoires, de contrebalancer les occlusions et de produire des trames de haute qualité présentant un mouvement très régulier.

Les tests de performances effectués dans [64] par rapport à quatre autres convertisseurs ont montré que le CRC-FRC procure une bien meilleure qualité vidéo aussi bien subjective qu'objective.

Annexe E

Reconstruction optimale des coefficients de source quantifiés

(À des fins d'uniformité, nous réutiliserons ici les notations du chapitre 5.)

Cette annexe vise à présenter les relations d'obtention des coefficients de source $\hat{X}_{2k}^{DCT2}(i,j)$ à partir de ceux quantifiés $\hat{X}_{2k}^{Q}(i,j)$ et des coefficients de l'information latérale $Y_{2k}^{DCT2}(i,j)$ pour le système de codage vidéo distribué guidé par le coefficient DC. Nous y présenterons la reconstruction dite standard [24] ainsi que la reconstruction optimale [53].

E.1 Reconstruction standard

Nous adaptons ici les équations de la reconstruction standard (cf. § 3.3.2) des coefficients de source quantifiés en fonction des notations du chapitre 5. On obtient alors :

$$\hat{X}_{2k}^{DCT2}(i,j) = \begin{cases} Y_{2k}^{DCT2}(i,j) & \text{, si } I_{min} \leq Y_{2k}^{DCT2}(i,j) < I_{max}, \\ \text{round}\,(I_{min}) & \text{, si } Y_{2k}^{DCT2}(i,j) < I_{min}, \\ \text{round}\,(I_{max}) & \text{, si } Y_{2k}^{DCT2}(i,j) \geq I_{max}, \end{cases} \tag{E.1}$$

avec

$$Y_{2k}^{DCT2}(i,j) = \text{sign}\Big(Y_{2k}^{DCT1}(i,j)\Big).\Big[\big|Y_{2k}^{DCT1}(i,j)\big|\, M(Q_M,r)S(Q_M,r) >> 2^{15}\Big]. \tag{E.2}$$

Les limites des intervalles de quantifications se calculent différemment selon le type de coefficients considéré et selon sa valeur par rapport à zéro. Ainsi pour les coefficients DC, les relations sont :

$$I_{min} = \begin{cases} 0 & \text{, si } \hat{X}_{2k}^{Q}(i,j) = 0, \\ \left(\hat{X}_{2k}^{Q}(i,j) - f\right) \cdot S\,(Q_M,r) \cdot 2^{Q_E} & \text{, si } \hat{X}_{2k}^{Q}(i,j) > 0, \end{cases} \tag{E.3}$$

$$I_{max} = \left(\hat{X}^{Q}_{2k}(i,j) + 1 - f\right) \cdot S\left(Q_M, r\right) \cdot 2^{Q_E}, \text{ si } \hat{X}^{Q}_{2k}(i,j) \geq 0. \tag{E.4}$$

De la même manière pour les coefficients AC, nous avons :

$$I_{min} = \begin{cases} \left(\hat{X}^{Q}_{2k}(i,j) - f\right) \cdot S\left(Q_M, r\right) \cdot 2^{Q_E} & \text{, si } \hat{X}^{Q}_{2k}(i,j) > 0, \\ \left(\hat{X}^{Q}_{2k}(i,j) - 1 + f\right) \cdot S\left(Q_M, r\right) \cdot 2^{Q_E} & \text{, si } \hat{X}^{Q}_{2k}(i,j) \leq 0, \end{cases} \tag{E.5}$$

$$I_{max} = \begin{cases} \left(\hat{X}^{Q}_{2k}(i,j) + 1 - f\right) \cdot S\left(Q_M, r\right) \cdot 2^{Q_E} & \text{, si } \hat{X}^{Q}_{2k}(i,j) \geq 0, \\ \left(\hat{X}^{Q}_{2k}(i,j) + f\right) \cdot S\left(Q_M, r\right) \cdot 2^{Q_E} & \text{, si } \hat{X}^{Q}_{2k}(i,j) < 0. \end{cases} \tag{E.6}$$

E.2 Reconstruction optimale

Dans [53], l'auteur présente un procédé de reconstruction optimale au sens de la minimisation de l'erreur quadratique moyenne des coefficients reconstruits. Aucun détail concernant sa démonstration ne sera donné ici. Dans ce qui suit par contre, nous adaptons les formules selon les notations du chapitre 5 et les relations de quantification et de transformation de la norme H.264/AVC. La reconstruction optimale se décrit alors comme suit :

$$\hat{X}^{DCT2}_{2k}(i,j) = \begin{cases} \text{round}\left(Y^{DCT2}_{2k}(i,j) + a\right) & \text{, si } I_{min} \leq Y^{DCT2}_{2k}(i,j) < I_{max}, \\ \text{round}\left(I_{min} + b\right) & \text{, si } Y^{DCT2}_{2k}(i,j) < I_{min}, \\ \text{round}\left(I_{max} - b\right) & \text{, si } Y^{DCT2}_{2k}(i,j) \geq I_{max}, \end{cases} \tag{E.7}$$

avec

$$Y^{DCT2}_{2k}(i,j) = \text{sign}\left(Y^{DCT1}_{2k}(i,j)\right).\left[\left|Y^{DCT1}_{2k}(i,j)\right| M(Q_M, r) S(Q_M, r) >> 2^{15}\right]. \tag{E.8}$$

La quantité a se détermine ainsi :

$$a = \frac{\left(\gamma + \frac{1}{\alpha}\right) e^{-\alpha\gamma} - \left(\delta + \frac{1}{\alpha}\right) e^{-\alpha\delta}}{2 - \left(e^{-\alpha\gamma} + e^{-\alpha\delta}\right)}, \tag{E.9}$$

avec

$$\gamma = Y^{DCT2}_{2k}(i,j) - I_{min}, \tag{E.10}$$

et

$$\delta = I_{max} - Y^{DCT2}_{2k}(i,j). \tag{E.11}$$

De même la quantité b se définie comme :

$$b = \frac{1}{\alpha} + \frac{\Delta}{1 - e^{\alpha\Delta}}, \tag{E.12}$$

où α est le paramètre du modèle Laplacien des différences entre la source et l'information latérale (respectivement $X_{2k}^{DCT2}(i,j)$ et $Y_{2k}^{DCT2}(i,j)$ ici) et où Δ correspond à la largeur de l'intervalle de quantification :

$$\Delta = I_{max} - I_{min}. \tag{E.13}$$

Il est intéressant de noter ici (cf. [53]) que lorsque $\alpha \rightarrow 0$, $Y_{2k}^{DCT2}(i,j)$ ne porte aucune information concernant la source $X_{2k}^{DCT2}(i,j)$. Cette reconstruction équivaut alors à prendre la valeur médiane de l'intervalle de quantification $\hat{X}_{2k}^{DCT2}(i,j) = I_{mid} = (I_{max} + I_{min})/2$. De la même manière, lorsque $\alpha \rightarrow \infty$, la reconstruction devient similaire à la reconstruction standard donnée en première partie de cette annexe.

Remarque : La reconstruction décrite au § 5.2.2 est encore différente. Elle correspond à la reconstruction optimale avec $\alpha \rightarrow 0$ et $b = 0$. L'information latérale garde ainsi son importance lorsque $Y_{2k}^{DCT2}(i,j) \in [I_{min},\ I_{max})$.

Bibliographie

[1] D. Slepian et J. Wolf, « Noiseless Coding of Correlated Information Sources », *IEEE Transactions on Information Theory*, vol. 19, juillet 1973.

[2] A. Wyner et J. Ziv, « The Rate-Distortion Function for Source Coding With Side Information at the Decoder », *IEEE Transactions on Information Theory*, vol. 22, janvier 1976.

[3] P. List, A. Joch, J. Lainema, G. Bjontegaard et M. Karczewicz, « Adaptative Deblocking Filter », *IEEE Transactions on Circuits and Sytems for Video Technology*, vol. 13, juillet 2003.

[4] I. E. G. Richardson, *H.264 and MPEG-4 Video Compression : Video Coding for Next-generation Multimedia.* John Wiley & Sons, 2003.

[5] C. Brites, J. Ascenso et F. Pereira, « Improving Transform Domain Wyner-Ziv video Coding Performance », Toulouse, France, IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), mai 2006.

[6] D. Varodayan, A. Aaron et B. Girod, « Rate Adaptative Distributed Source Coding using Low-Density Parity Check codes », *EURASIP Signal Processing Journal, Special Issue on Distributed Source Coding*, vol. 86, novembre 2006.

[7] D. Salomon, *Data Compression. The Complete Reference.* Springer, 2004.

[8] B. Girod, A. M. Aaron, S. Rane et D. Rebollo-Monedero, « Distributed Video Coding », *Proceedings of the IEEE*, vol. 93, janvier 2005.

[9] J.-Y. Chouinard, « Information Theory : Course notes », School of Information Technology and Engineering, University of Ottawa, avril 2001.

[10] T. M. Cover et J. A. Thomas, *Elements of Information Theory.* Wiley Series in Telecommunications, A Wiley-Interscience publication, 1991.

[11] A. D. Liveris, Z. Xiong et C. N. Georghiades, « Compression of Binary Sources With Side Information at the Decoder Using LDPC Codes », *IEEE Communications Letters*, vol. 6, octobre 2002.

[12] A. Wyner, « On Source Coding With Side Information at the Decoder », *IEEE Transactions on Information Theory*, vol. 21, mai 1975.

[13] Y. Q. Shi et H. Sun, *Image and Video Compression for Multimedia Engineering. Fundamentals, Algorithms, and Standards.* CRC Press, 2000.

[14] A. K. Salkintzis et N. Passas, *Emerging Wireless Multimedia. Services and Technologies.* A Wiley-Interscience publication, 2006.

[15] K. R. Rao et P. C. Yip, *The Transform and Data compression Handbook.* CRC Press, 2001.

[16] « Wikipédia - DCT », 2008.

[17] T. Wedi et S. Wittmann, « Quantization Offsets for Video Coding », *IEEE International Symposium on Circuits and Systems (ISCAS)*, vol. 1, p. 324–327, mai 2005.

[18] « Wikipédia - Quantification (signal) », 2008.

[19] T. Wiegand, G. J. Sullivan, G. Bjontegaard et A. Luthra, « Overview of the H.264/AVC Video Coding Standard », *IEEE Transactions on Circuits and Sytems for Video Technology*, vol. 13, juillet 2003.

[20] *ISO/IEC 14496-10 and ITU-T Rec. H.264, Advanced Video Coding*, 2003.

[21] « Wikipédia - H.264 », 2008.

[22] A. Aaron, S. Rane, E. Setton et B. Girod, « Transform-Domain Wyner-Ziv Codec for Video », San Jose, California, USA, Visual Communications and Image Processing Conference (VCIP), 2004.

[23] C. Berrou, A. Glavieux et P. Thitimajshima, « Near Shannon Limit Error-Correcting Coding and Decoding : Turbo Codes », mai 1993.

[24] A. Aaron, R. Zhang et B. Girod, « Wyner-Ziv Coding of Motion Video », Pacific Grove, California, USA, IEEE Asilomar Conference on Signals, Systems and Computers, novembre 2002.

[25] S. Pradhan et K. Ramchandran, « Distributed Source Coding Using Syndromes (DISCUS) : Design and Construction. », Snowbird, UT, IEEE Data Compression Conference (DCC), mars 1999.

[26] J. Garcias-Frias et Y. Zhao, « Compression of Correlated Binary Sources Using Turbo Codes », *IEEE Communications Letters*, vol. 5, octobre 2001.

[27] J. Garcias-Frias et Y. Zhao, « Compression of Binary Memoryless Sources Using Punctured Turbo Codes », *IEEE Communications Letters*, vol. 6, septembre 2002.

[28] J. Bajcsy et P. Mitran, « Coding for the Slepian-Wolf Problem With Turbo Codes », vol. 2, IEEE Global Telecommunications Conference (GLOBECOM), novembre 2001.

[29] J. Bajcsy, B. Xu, P. Mitran et C.Beainy, « Turbo Coding in the Slepian-Wolf and Multi-Access Channel Problems : Issues of Duality », vol. 1, IEEE Asilomar Conference on Signals, Systems and Computers, novembre 2002.

[30] A. Aaron et B. Girod, « Compression With Side Information Using Turbo Codes », IEEE Data Compression Conference (DCC), avril 2002.

[31] D. J. C. MacKay, « Fountain Codes », *IEEE Proceedings on Communication*, vol. 152, décembre 2005.

[32] A. Shokrollahi, « Raptor Codes », *IEEE Transactions on Information Theory*, vol. 52, p. 2551–2567, 2006.

[33] A. Aaron, S. Rane, R. Zhang et B. Girod, « Wyner-Ziv Coding for Video : Applications to Compression and Error Resilience », IEEE Data Compression Conference (DCC), 2003.

[34] A. Aaron, E. Setton et B. Girod, « Towards Practical Wyner-Ziv Coding of Video », Barcelona, Spain, IEEE International Conference on Image Processing (ICIP), 2003.

[35] R. Puri et K. Ramchandran, « PRISM A New Robust Video Coding Architecture Based on Distributed Compression Principles », Allerton, IL, Allerton Conference on Communication, Control, and Computing, 2002.

[36] J. E. Fowler, M. Tagliasacchi et B. Presquet-Popescu, « Wavelet-Based Distributed Source Coding of Video », *in Proceedings of the European Signal Processing Conference*, Antalya, Turkey, septembre 2005.

[37] F. Yang, G. Ding, Q. Dai et Y. Ying, « Wavelet Based Scalable Wyner-Ziv Video Coding », Beijing, China, Picture Coding Symposium (PCS), avril 2006.

[38] M. Flierl et P. Vandergheynst, « Distributed Coding of Highly Correlated Image Sequences with Motion-Compensated Temporal Wavelets », *EURASIP Journal on Applied Signal Processing*, 2006.

[39] J. Ascenso, C. Brites et F. Pereira, « Improving Frame Interpolation with Spatial Motion Smoothing for Pixel Domain Distributed Video Coding », Smolenice, Slovak Republic, EURASIP Conference on Speech and Image Processing, MultimediaCommunications and Services, juillet 2005.

[40] J. Sun et H. Li, « Motion Compensated Wyner-Ziv Video Coding », 2005.

[41] L. Lu et V. Sheinen, « Side Information Generation for Low Complexity Video Coding Systems Based on Wyner-Ziv Theorem », 2006.

[42] Z. Gan, L. Qi et X. Zhu, « A Novel Improved Method of Pixel-Domain Wyner-Ziv Video Coding », Beijing, China, Picture Coding Symposium (PCS), avril 2006.

[43] M. Tagliasacchi, A. Trapanese, S.Tubaro, J. Ascenso, C. Brites et F. Pereira, « Exploiting Spatial Redundancy in Pixel Domain Wyner-Ziv Video Coding ». 2006.

[44] Z. He, L. Cao et H. Cheng, « Correlation Estimation and Performance Optimisation for Distributed Image Compression », San Jose, California, USA, Visual Communication and Image Processing Conference (VCIP), janvier 2006.

[45] A. Trapanese, M. Tagliasacchi, S. Tubaro, J. Ascenso, C. Brites et F. Pereira, « Improved Correlation Noise Statistics Modelling in Frame-based Pixel Domain Wyner-Ziv Video Coding », Sardinia, Italy, International Workshop on Very Low Bitrate Video Coding, septembre 2005.

[46] D. Kubasov, K. Lajnef et C. Guillemot, « A Hybrid Encoder/Decoder Rate Control for a Wyner-Ziv Video Codec with a Feedback Channel », Chania, Crete, IEEE Multimedia Signal Processing Workshop, octobre 2007.

[47] R. P. Westerlaken, S. Borchert, R. K. Gunnewiek et R. L. Lagendijk, « Analyzing Symbol and Bit Plane-Based LDPC in Distributed Video Coding », San Antonio, Texas, USA, IEEE International Conference on Image Processing (ICIP), septembre 2007.

[48] C. Brites et F. Pereira, « Encoder Rate Control for Transform Domain Wyner-Ziv Video Coding », San Antonio, USA, IEEE International Conference on Image Processing (ICIP), septembre 2007.

[49] A. Adikari, W. Fernando et W. Weerakkody, « Iterative Wyner-Ziv Decoding for Unidirectional Distributed Video Coding », *IEEE Electronic Letters*, vol. 47, p. 93–95, janvier 2007.

[50] S. Ye, M. Ouaret, F. Dufaux, M. Ansorge et T. Ebrahimi, « Error Resiliency of Distributed Video Coding in Wireless Video Communication », *in SPIE Optics and Photonics*, Proceedings of SPIE, Bellingham, WA, SPIE, 2008.

[51] C. Yaacoub, J. Farah et B. Pesquet-Popescu, « Feedback Channel Suppression in Distributed Video Coding with Adaptive Rate Allocation and Quantization for Multiuser Applications », *EURASIP Journal on Wireless Communications and Networking*, 2008.

[52] J. Martinez, C. Holder, G. Fernandez, H. Kalva et F. Quiles, « DVC Using a Half-Feedback Based Approach », p. 1125–1128, IEEE International Conference on Multimedia and Expo, juin 2008.

[53] D. Kubasov, J. Nayak et C. Guillemot, « Optimal Reconstruction in Wyner-Ziv Video Coding with Multiple Side Information », Chania, Crete, IEEE Multimedia Signal Processing Workshop, octobre 2007.

[54] F. Pereira, J. Ascenso et C. Brites, « Studying the GOP Size Impact on the Performance of a Feedback Channel-Based Wyner-Ziv Video Codec », Santiago, Chile, IEEE Pacific Rim Symposium on Image Video and Technology (PSIVT), décembre 2007.

[55] W.-J. Chien, L. J. Karam et G. P. Abousleman, « Block-Adaptive Wyner-Ziv Coding for Transform-Domain Distributed Video Coding », Honolulu, Hawaii, USA, IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), avril 2007.

[56] H. S. MALVAR, A. HALLAPURO, M. KARCZEWICZ et L. KEROFSKY, « Low-Complexity Transform and Quantization in H.264/AVC », *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, juillet 2003.

[57] D. ZIGHED et R. RAKOTOMALALA, « Extraction de connaissances à partir de données (ECD) », *in Techniques de l'Ingénieur*, novembre 2002.

[58] I. H. WITTEN et E. FRANK, *Data Mining : Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, 2005.

[59] Y. WANG et I. H. WITTEN, « Induction of Model Trees for Predicting Continuous Classes », *in Proceedings of the Poster Papers of the European Conference on Machine Learning*, University of Economics, Faculty of Informatics and Statistics, Prague, 1997.

[60] J. QUINLAN, « Learning with Continuous Classes », *in Proceedings of the Australian Joint Conference on Artificial Intelligence*, Singapore, World Scientific, 1992.

[61] L. BREIMAN, J. H. FRIEDMAN, R. A. OLSHEN et C. STONE, *Classification and Regression Trees*. 1984.

[62] C. BRITES, J. ASCENSO et F. PEREIRA, « Feedback Channel in Pixel Domain Wyner-Ziv Video Coding Myths and Realities », Florence, Italy, European Signal Processing Conference (EUSIPCO), septembre 2006.

[63] J. PEDRO, C. BRITES, J. ASCENSO et F. PEREIRA, « Studying the Feedback Channel in Transform Domain Wyner-Ziv Video Coding », Peniche, Portugal, 6th Conference on Telecommunications (ConfTele), mai 2007.

[64] P. BLANCHFIELD, D. WANG, A. VINCENT, F. SPERANZA et R. RENAUD, « Advanced Frame Rate Conversion and Performance Evaluation », *The SMPTE Motion Imaging Journal*, p. 153–159, avril 2006.

[65] G. HUCHET, J.-Y. CHOUINARD, D. WANG et A. VINCENT, « Decision Trees for Denoising in H.264-AVC Video Sequences », San Jose, California, USA, Visual Communication and Image Processing Conference (VCIP), janvier 2008.

[66] S. CHUNG, G. D. FORNEY, T. J. RICHARDSON et R. URBANKE, « On the Design of Low-Density Parity-Check Codes Within 0.0045 dB of the Shannon Limit », *IEEE Communications Letters*, vol. 5, p. 58–60, février 2001.

[67] T. K. MOON, *Error Correction Coding : Mathematical Methods and Algorithms*, chap. 15 : Low-Density Parity-Check Codes. Wiley Interscience, 2005.

[68] D. J. C. MACKAY, *Information Theory, Inference and Learning Algorithms*, chap. 47 : Low-Density Parity-Check Codes, p. 557–573. Cambridge, 2003.

[69] J. HA, J. KIM, D. KLINC et S. W. MCLAUGHLIN, « Rate-Compatible Punctured Low-Density Parity-Check Codes With Short Block Lengths », *IEEE Transactions on Information Theory*, vol. 52, février 2006.