Codage Vidéo Distribué utilisant les Turbo Codes

Thèse

MOHAMED HAJ TAIEB

Doctorat en Génie Électrique
Philosophiæ doctor (Ph.D.)

Québec, Canada

Résumé

La charge de traitement de la compression vidéo est généralement assurée au niveau de l'émetteur dans les standards conventionnels de codage vidéo (MPEG, H.263, H.264/AVC [1]). Ce choix est dû au fait que l'émetteur dispose d'une connaissance totale de la source lui permettant d'assurer facilement et efficacement la compression. En outre, les applications habituelles de la transmission vidéo assurent un flux d'une station centralisée, disposant d'une capacité calculatoire élevée, vers un certain nombre de récepteurs. La charge de compression est ainsi effectuée une seule fois par une station adaptée à ce fait.

Avec l'accroissement de l'interactivité de la téléphonie mobile, les abonnés sont en mesure d'émettre des séquences vidéo autant que d'en recevoir. Le flux vidéo est alors généré par le téléphone mobile à capacité limitée vers une station de base à capacité élevée. La compression ne peut se faire selon le schéma conventionnel et la complexité de la compression doit être transférée au décodeur. Les résultats théoriques de Slepian-Wolf [2] et leur extension par les travaux de Wyner-Ziv [3] pour le cas de codage avec perte et en présence d'information latérale constituent les bases théoriques du codage de sources distribuées. Ces résultats théoriques ont donné naissance à un vaste champ d'applications comme le récent paradigme de codage vidéo distribué, établi il y a juste quelques années.

Dans cette thèse de doctorat, on présente une étude des divers systèmes de codage vidéo distribué dans le domaine pixel et dans le domaine transformé. Le décodeur exploite la corrélation entre la séquence à transmettre par le codeur et l'information latérale dont il dispose. Cette corrélation se traduit par un canal virtuel dont l'entrée est la trame à transmettre et la sortie est l'information latérale. Un code turbo est utilisé pour générer des bits de parité envoyés progressivement sous la demande du décodeur, pour corriger les erreurs de l'information latérale qui constitue une version bruitée de la trame originale. Dans ce travail, on implémente différents algorithmes de codage vidéo distribué basés sur les turbo-codes dans le but de s'approcher des performances de la compression vidéo conventionnelle.

Abstract

Most of the video compression processing is usually performed at the transmitter in the conventional video coding standards (MPEG, H.263, H.264/AVC [1]). This choice is due to the fact that the transmitter has full knowledge of its source to ensure easy and efficient compression. In addition, the usual applications of video transmission ensure a flow from a centralized station, with a higher computational capacity, to a number of receivers. The compression task is thus performed only once by a computationally adapted station.

However, with the emergence of wireless surveillance locally distributed cameras, the growth of cellular interactive video applications as well as many other applications involving several low cost video encoders at the expense of high complexity central decoder, the compression task can no longer be handled by the encoder and thus the compression complexity should be transferred to the decoder. Slepian and Wolf information theoretical result on lossless coding for correlated distributed sources [2] and its extension to the lossy source coding case with side information at the decoder, as introduced by Wyner and Ziv [3], constitute the theoretical basis of distributed source coding. These theoretical concepts have given birth to a wide field of applications as the recent distributed video coding paradigm, established a few years ago.

In this doctoral thesis, we present a study of various distributed video coding schemes in the pixel and transform domains. The decoder exploits the correlation between the video sequence to be transmitted by the encoder and the side information. This correlation can be seen as a virtual channel whose input is the frame to be transmitted and the output is the side information. Turbo coding is used to generate the parity bits which are sent, gradually upon decoder requests, to correct the errors in the side information considered as a noisy version of the original frame. In this work, we implement various algorithms for distributed video coding based on turbo codes in order to approach the efficiency of conventional video encoders.

Remerciements

Je tiens tout d'abord à remercier vivement mon directeur de recherche, le professeur Jean-Yves Chouinard, pour sa méthodologie de travail et sa disponibilité malgré un emploi de temps fort chargé. Le déroulement de mon projet de recherche a été sagement guidé par des réunions régulières aux cours desquelles il a su combiner sympathie et sérieux afin de me faire profiter de son expertise pour évaluer mes travaux de recherche et de ses conseils enrichissants pour diriger l'évolution de mon cursus académique ainsi celle de mon projet de recherche.

Mon projet de doctorat est effectué en collaboration avec le groupe de recherche sur les systèmes vidéo de pointe au Centre de Recherche sur les Communications d'Ottawa que je tiens à remercier et surtout mon co-directeur Dr. Demin Wang pour ses directives précises et efficaces pour l'avancement de mes recherches et pour la révision des divers rapports effectués. Ses conseils, ses idées innovatrices, sa patience et sa sympathie m'ont effectivement aidé pour progresser dans mon travail de recherche.

Je remercie aussi mon ami, le Dr. Grégory Huchet. Son projet de doctorat a aussi porté sur le codage vidéo distribué. Ainsi, je le consultais fréquemment au sujet de certains points : il n'a jamais hésité à consacrer son temps pour se pencher pleinement avec moi sur ces questions et à me faire part de toute son expertise.

J'adresse ensuite mes remerciements à tous les étudiants et amis du laboratoire de radiocommunication et de traitement de signal (LRTS) et tout le corps enseignant et administratif qui ont fait en sorte que le cadre de travail soit instructif, propice et agréable.

Finalement, je remercie spécialement tous les membres de ma famille pour la patience qu'ils m'ont accordée et les énormes sacrifices qu'ils ont faits pour que je sois là où je n'aurais jamais pu être sans eux.

Liste des symboles

b	Bande DCT
B	Trame bidirectionnelle
b_k^{SI}	Coefficient de la bande k de la DCT de la trame interpolée
b_k^{WZ}	Coefficient de la bande k de la DCT de la trame WZ
C	Capacité du canal [bits par seconde]
C_b	Composante de chrominance relative à la différence avec le bleu
C_r	Composante de chrominance relative à la différence avec le rouge
D	Distorsion
(dx_B, dy_B)	Vecteur de mouvement de la trame arrière X_B
(dx_F, dy_F)	Vecteur de mouvement de la trame avant X_F
E_i	Valeurs attendues ou valeurs théoriques dans le test statistique du χ^2
H	Entropie [Shannon ou bits]
I	Trame Intra
K	Trame clé
L	Longueur binaire du code ou de l'entrelaceur
N_0	Niveau de l'energie du bruit [watt $\times$ seconde]
n_{pi}	Nombre de bits de parité envoyé pour l'un des codeurs RSC de l'encodeur turbo par période de poinçonnage. $i = 1,2$ étant le numéro du codeur RSC
N_i	Nombre de bits seuil pour changer la matrice de poinçonnage
O_i	Valeurs observées ou empiriques dans le test statistique du χ^2
P	Trame prédictive
$P_{XY}(x,y)$	Distribution de probabilité jointe
P_{signal}	Puissance totale reçue sur la largeur de bande du signal
Q	Nombre de bits de quantification
R	Trame résiduelle
$\Re$	Région d'admissibilité
$R^*(D)$	Débit minimum requis pour une reconstruction de distorsion moyenne D
R_{min}	Débit minimum estimé par l'encodeur SW afin de réduire la sollicitation du canal de retour

R_X	Débit [bits par symbole]
Reg_k	Région numéro k de la bande DCT après subdivision
T	Transformation DCT de la trame résiduelle R
T_b	Les composantes DCT de la trame transformée T
U,V	Composantes de chrominance
X	Trame Wyner-Ziv
$\hat{X}$	Trame reconstruite
X_B	Trame vers l'arrière / *Backward frame*
X_F	Trame vers l'avant / *Forward frame*
$X_B\left(x+dx_B,x+dy_B\right)$	Version de la trame vers l'arrière compensée en mouvement / *Motion compensated version of the backward frame*
$X_F\left(x+dx_F,x+dy_F\right)$	Version de la trame vers l'avant compensée en mouvement / *Motion compensated version of the forward frame*
X,Y	Variables aléatoires
Y	Information latérale
Y	Composante de luminance
W	Largeur de bande du canal [Hertz]
α	Paramètre de la distribution Laplacienne
$\alpha_{l+1}^{*}\left(s\right)$	Métrique directe de branche au temps $l+1$ de l'état suivant s
$\beta_{l}^{*}\left(s'\right)$	Métrique inverse de branche au temps l de l'état précédent s'
$\Delta PSNR_{1g}^{EC}$	Différence du PSNR entre le groupe g et le groupe 1 utilisant le motif EC
$\Lambda\left(x_k\right)$	Rapport de vraisemblance du bit d'information x_k
$\gamma_k\left(s,s'\right)$	Métrique de branche entre l'état s et l'état s'
σ	Écart-type de la distribution Laplacienne
$\left\|.\right\|_2$	Norme euclidienne
∇	Opérateur du gradient
$\left(\cdot\right)^T$	Opérateur de la transposée

Abréviations

4D	*Four Diagonal blocks template* / Motif des 4 blocs diagonals
AC	*Alternating Current* / Courant alternatif
AN	*All Neighboring blocks template* / Motif de tous les blocs voisins
AWGN	*Additive White Gaussian Noise* / Bruit blanc additif gaussien
BME	*Backward Motion Estimation* / Estimation de mouvement vers l'arrière
BCJR	Algorithme de Bahl, Cocke, Jelinek et Raviv [4]
BER	*Bit Error Rate* / Taux d'erreur binaire
BiME	*BIdirectionnel Motion Estimation* / Estimation de mouvement bidirectionnelle
BiMESS	*BIdirectionnel Motion Estimation with Spatial Smoothing* / Estimation de mouvement bidirectionnelle avec lissage spatial
BP	*Rate estimation based on Bit-plane correlation* / Estimation du débit basée sur la corrélation entre les plans de bits
CDF	*Cumulative Distribution Function* / Fonction de distribution cumulative
CRC	*Communication Research Center Canada* / Centre de recherches sur les communications Canada
CRC	*Cyclic Redundancy Check* / Contrôle de redondance cyclique
DC	*Direct Current* / Courant direct
DCT	*Discrete Cosine Transform* / Transformée en cosinus discrète
DPCM	*Differential Pulse Code Modulation* / Modulation par impulsions et codage différentiel
DRC	*Decoder Rate Control* / Contrôle du débit au décodeur
DVC	*Distributed Video Coding* / Codage vidéo distribué
DWT	*Discrete Wavelet Transform* / Transformée en ondelettes discrète
EQM	Erreur Quadratique Moyenne
EC	*Empty Cross template* / Motif de la croix vide
FME	*Forward Motion Estimation* / Estimation de mouvement vers l'avant

FNC	*Final Number of parity bits Chunks* / Nombre final de fragments de bits de parité
fps	*Frames per second* / Trames par seconde
FSI	*Final Side Information* / Information latérale finale
GOP	*Group of pictures* / Groupe d'images
I	*Intra frame* / Trame intra
INC	*Initial Number of parity bits Chunks* / Nombre initial de fragments de bits de parité
ISI	*Initial Side Information* / Information latérale initiale
IST-PDWZ	*Instituto Superior Técnico - Pixel Domain Wyner-Ziv* / Codec WZ dans le domaine pixel de l'Instituto Superior Técnico
IST-TDWZ	*Instituto Superior Técnico - Transform Domain Wyner-Ziv* / Codec WZ dans le domaine transformé de l'Instituto Superior Técnico
JPEG	*Joint Photographic Experts Group*
K	*Key frame* / Trame clé
kbps	*Kilobits per second* / Kilobits par seconde
KL	Transformation de Karhunen-Loève
LDPC	*Low Density Parity Check code* Contrôle de parité à faible densité
L_a	*Likelihood a priori value* / Valeur de vraisemblance à priori
L_{ext}	*Likelihood EXTrinsic value* / Valeur de vraisemblance extrinsèque
log-MAP	log-Maximum À Postériori
LPF	*Low Pass Filter* / Filtre passe bas
LSB	*Least Significant Bit* / Bit de poids faible
MAD	*Mean Absolute Diffrence* / Différence absolue moyenne
MAE	*Mean Absolute Error* / Erreur absolue moyenne
MCR	*Motion-Compensation Restoration* / Restauration de la compensation de mouvement
MCTI	*Motion Compensated Temporal Interpolation* Interpolation temporelle à compensation de mouvement
MP3	*MPEG-1 Audio Layer 3*
MPEG	*Moving Picture Experts Group*
MSB	*Most Significant Bit* / Bit de poids fort
MSE	*Mean Squared Error* / Erreur quadratique moyenne
MV	*Motion Vector* / Vecteur de mouvement
pb	plan de bits
PDF	*Probability Density Function* / Fonction de densité de probabilité
PDWZ	*Partially Decoded WZ frame* / Trame WZ partiellement décodée

PSNR	*Peak Signal-to-Noise Ratio*
	Rapport signal à bruit de crête
QCIF	*Quarter Common Intermediate Format*
R-D	*Rate-Distortion performances* / Performances de débit-distorsion
RGB	*Red, Green, Blue*
RSC	*Recursive Systematic Convolutional code* /
	Codes convolutifs systématiques et récursifs
SER	*Symbol Error Rate* / Taux d'erreurs symboles
SI	*Side Information* / Information latérale
SSIM	*Structural SIMilarity* / Similarité structurelle
SW	Slepian-Wolf
TC	*Rate estimation based on Temporal Correlation*
	Estimation du débit basée sur la corrélation temporelle
T.O.S.	Termes d'ordre supérieur
tps	Trames par seconde
VHS	*Visual Human System* / Système visuel humain
WMVF	*Weighted Median Vector Filter* / Filtre médian vectoriel pondéré
WZ	Wyner-Ziv

Table des matières

Liste des tableaux

Liste des figures

Chapitre 1

Introduction générale

1.1 Mise en contexte et problématique

En codage vidéo, l'encodeur exploite la corrélation temporelle et spatiale pour réduire le débit de transmission. C'est l'encodeur qui se charge de l'évaluation des vecteurs de mouvement permettant au décodeur de reconstruire la vidéo, en toute simplicité. La complexité calculatoire de l'encodeur est ainsi de beaucoup supérieure à celle du décodeur. Cette asymétrie de complexité est parfaitement adaptée pour la plupart des applications de codage vidéo comme la télédiffusion numérique ou les services de téléchargement de vidéo par des mobiles. Dans ce type d'application, le flux de données est véhiculé d'une station de base vers un certain nombre de terminaux. La vidéo est donc compressée une seule fois au niveau de la station de base et décodée plusieurs fois au niveau de chaque récepteur. Ainsi l'encodeur est plus susceptible de supporter la complexité du codage que les récepteurs.

Cependant, certaines applications émergentes nécessitent un scénario dual où le codeur dispose de ressources relativement limitées et où le décodeur dispose de plus de puissance de traitement. Ce qui est le cas du téléversement où le flux vidéo est transmis du mobile vers la station de base. Pour effectuer ce transfert de complexité de l'encodeur vers le décodeur, la solution est d'exploiter les statistiques de la source au niveau du décodeur par l'application de la technique de décodage de sources distribuées avec information latérale telle que définie par Wyner-Ziv [3]. Dans le contexte de la transmission vidéo, cette méthode de codage est identifiée par l'appelation de codage vidéo distribué : elle reproduit la même technique pour le décodage de l'image avec information latérale (*side information*). La question qui se pose alors est : "en quoi consiste l'information latérale dans le cas de la transmission vidéo ?"

Comme la vidéo est une suite d'images en mouvement, deux images successives présentent généralement une forte similitude de telle sorte que l'on peut considérer la première image comme une version bruitée de la deuxième image et peut constituer alors l'information latérale au niveau du décodeur. Dans le cas où plusieurs caméras transmettent la même séquence vidéo à une station centralisée, cette dernière peut exploiter la corrélation entre les séquences vidéos reçues comme information latérale.

Plusieurs méthodes plus avancées peuvent être considérées pour la génération d'une information latérale de plus en plus corrélée avec l'information originale. Dans ce travail, on considère principalement l'interpolation temporelle à compensation de mouvement (MCTI) entre une trame antérieure (*backward frame*) et une trame postérieure (*forward frame*). La génération de cette trame interpolée s'effectue au niveau du décodeur et la tâche d'estimation de mouvement n'est plus supportée par l'encodeur. D'où le transfert de complexité de l'encodeur vers le décodeur. L'architecture DVC doit assurer une exploitation efficace de l'information latérale pour des performances de débit-distorsion (R-D) compétitives. L'architecture considère un canal bruité virtuel dont l'entrée et la sortie consistent en l'information originale et l'information latérale, respectivement. L'information latérale est alors appliquée à l'entrée d'un décodeur permettant de corriger supposément les erreurs introduites par ce canal virtuel. On utilise à cet effet un code canal performant comme les codes turbo et les codes LDPC atteignant des performances proches de la limite théorique établie par Claude Shannon [7].

1.2 Organisation de la thèse

Le but de ce travail est d'étudier, d'implémenter et d'analyser les performances des diverses architectures d'un codeur/décodeur vidéo Wyner-Ziv basé sur le codage et le décodage turbo [12]. Le codage vidéo distribué est, en fait, une nouvelle technique de codage et de compression vidéo basée sur le codage de source distribué. Ce dernier, quant à lui, est le résultat de l'évolution de trois principales découvertes en théorie de l'information : le résultat de Claude Shannon [7] sur la relation du débit atteignable en fonction de la distorsion tolérée, le codage sans perte de sources corrélées de Slepian-Wolf [2] et le codage de source avec perte de Wyner-Ziv [3]. À cet effet, on détaille, dans le deuxième chapitre ces principes de la théorie d'information pour expliquer comment ils ont été exploités dans le déploiement de ce nouveau paradigme de codage vidéo. L'architecture globale de ce codeur-décodeur telle que proposée dans la littérature est, ensuite, présentée en détaillant les différents blocs fonctionnels.

On présente, au chapitre 3, le principe du codage turbo, utilisant un codeur systématique récursif de rendement 4/5 et un codeur systématique récursif de rendement 1/2. On s'intéresse par la suite au processus de décodage turbo qui fait intervenir l'information latérale obtenue par interpolation des deux trames adjacentes à la trame objet du décodage. L'objectif d'utilisation de la technique turbo dans le contexte du codage vidéo distribué est de procéder à la correction des erreurs : non pas celles introduites par le canal de transmission mais plutôt celles qui résident dans la version interpolée de l'image constituant, dans le langage du codage distribué, l'information latérale. Le décodage turbo repose sur l'algorithme (BCJR) log-MAP, généralement déployé pour combattre les erreurs introduites par un canal blanc additif gaussien.

Toutefois, dans le contexte de codage vidéo distribué, le canal de transmission à considérer est un canal virtuel qui modélise l'erreur d'interpolation entre la trame originale et sa version estimée au niveau du décodeur. Pour déterminer le modèle de corrélation de ce canal virtuel, on effectue dans

le même chapitre 3 une étude statistique permettant d'évaluer la nature de la différence entre les trames originales et les trames interpolées. Cette étude nous a permis de constater que le modèle de corrélation suit une loi Laplacienne. On détaille ensuite dans ce même chapitre, le principe de l'algorithme (BCJR) log-MAP qui constitue la clé du déploiement des codes turbo. Le décodage turbo est implémenté pour le cas d'un code avec deux codeurs systématiques récursifs de rendement 1/2 et de rendement 4/5. Pour augmenter la compression, une matrice de poinçonnage est utilisée à la sortie du codeur turbo et les bits de parité sont transmis graduellement à la demande du décodeur. L'image interpolée est utilisée en premier lieu pour le décodage turbo afin de déterminer les symboles de quantification de la trame envoyée. En second lieu, l'image interpolée est utilisée pour la reconstitution de la trame envoyée en utilisant les symboles de quantification décodés. Les résultats de simulations présentent le niveau de distorsion de la séquence vidéo après reconstruction, en fonction du débit de transmission qui varie selon la précision du quantificateur (nombre de bits par pixel).

Dans le chapitre 4, on propose un nouveau schéma de codage vidéo distribué permettant d'optimiser, ou du moins améliorer, l'effet des bits de parité envoyés en les dirigeant vers les endroits du canal virtuel où ils sont le plus utiles. Ces endroits correspondent aux régions bruitées de la trame interpolée. Cette nouvelle technique permet de déterminer, au départ, les régions où l'interpolation est susceptible d'échouer. Ensuite, un nombre supérieur de bits de parité est dirigé vers ces régions pour combattre ces erreurs d'interpolation et un nombre moindre est envoyé vers les régions non bruitées. Ainsi, une réduction du nombre de bits de parité est réalisée permettant une meilleure compression.

Dans le chapitre 5, on s'intéresse à l'architecture du codec Wyner-Ziv dans le domaine transformé pour exploiter la redondance spatiale au niveau des trames. On implémente en premier lieu le codec dans le domaine pixel avec extraction des plans de bits. Par la suite on passe au domaine transformé avec la transformation en cosinus discrète (DCT). Le modèle de corrélation entre les coefficients DCT originaux et les coefficients DCT de l'information latérale est approximé par une distribution Laplacienne et une distribution gaussienne. Un test de χ^2 est effectué pour trancher entre ces deux distributions validant la pertinence du modèle Laplacien pour la caractérisation du canal virtuel dans le domaine transformé. On étudie aussi, dans ce chapitre, diverses dispositions des intervalles de quantification des différentes bandes des coefficients de la DCT. Les résultats de simulations démontrent bien le gain de performance obtenu par le passage du domaine pixel au domaine transformé.

On s'intéresse, dans le chapitre 6, au module d'interpolation bidirectionnelle de l'architecture DVC qui diffère du module d'interpolation dans les encodeurs vidéo standard. En effet, l'estimation des vecteurs de mouvement est effectuée en se basant sur les trames avoisinantes sans aucune information sur la trame Wyner-Ziv. On implémente, en premier lieu, l'une des techniques d'interpolation les plus utilisées dans la littérature, l'estimation de mouvement bidirectionnelle avec lissage spatial (BiMESS) [13], et en second lieu on adapte la technique de Lukas-Kanade avec pyramide pour l'estimation du flux optique au contexte du codage vidéo distribué. Ces techniques d'interpolation permettent d'obtenir une amélioration considérable des performances débit-distorsion par rapport à la technique simple d'interpolation par moyennage. En effet, l'amélioration de la qualité de l'information latérale

revient à considérer un canal virtuel moins bruité.

Le chapitre 7 quant à lui se concentre principalement à la ré-implémentation de l'architecture "benchmark" du codage vidéo distribué adopté par le système Discover [9]. Ce projet est le résultat d'un travail de collaboration entre 6 groupes de recherches de différentes universités Européennes. Leurs efforts ont résulté en un codec DVC promouvant non seulement les performances en termes de débit-distorsion mais aussi l'aspect pratique. Ainsi le paradigme du codage vidéo distribué peut dépasser le stade de projet de recherche pour se voir potentiellement l'œuvre d'une réalisation pratique. Après avoir implémenté les divers mécanismes déployés dans Discover et reproduit des résultats de simulations similaires pour fins de validation, on s'intéresse dans les chapitres qui suivent à présenter des améliorations.

L'architecture Discover ne présente pas de mécanisme d'exploitation de la corrélation spatiale similaire au codage DPCM intra-trame pour maintenir une complexité d'encodage aussi faible que possible. Dans le chapitre 8, on propose une nouvelle architecture DVC progressive qui permet d'exploiter la corrélation spatiale au niveau du décodeur en divisant la trame en groupes de blocs spatialement corrélés. Ces groupes sont envoyés et décodés un après l'autre de telle sorte que le décodage d'un groupe donné permet d'améliorer la qualité de l'information latérale relative au groupe suivant étant donné que ces groupes sont spatialement corrélés. L'architecture progressive permet d'obtenir une qualité meilleure de l'information latérale et par conséquent une amélioration des performances débit-distorsion (R-D) allant jusqu'à 3 dB.

L'amélioration des performances R-D est aussi un premier objectif du chapitre 9 où l'on a misé plutôt sur l'étude du codec Wyner-Ziv et plus particulièrement sur le mécanisme de poinçonnage dans le domaine des transformées. Dans un premier temps on considère l'utilisation des bits systématiques. Ceux-ci sont écartés dans les architectures DVC et remplacés par l'information latérale pour le calcul des rapports de vraisemblance canal lors du décodage turbo. Dans ce chapitre, ces bits systématiques sont déployés suivant une nouvelle technique. Celle-ci fait varier la matrice de poinçonnage suivant la qualité de l'information latérale pour permettre un usage efficace des bits de parité et des bits systématiques. Dans un second temps, une technique de poinçonnage est mise en place afin de placer efficacement les bits de parités et systématiques au lieu de les disperser périodiquement le long du plan de bits. Les emplacements susceptibles d'être bruités sont détectés et les bits y sont dirigés. L'utilisation adaptative des bits générés par le code turbo a permis une réduction du débit Wyner-Ziv (W-Z) allant jusqu'à 20%, se traduisant en un gain de performance globale de 1.8 dB.

Le chapitre 9, traite aussi d'un aspect plutôt d'ordre pratique sans pour autant compromettre les performances R-D. Il s'agit de la réduction de la durée de latence du décodeur par l'incorporation d'un mécanisme d'estimation efficace du débit afin de réduire les boucles de décodage turbo et de demande de bits supplémentaires via le canal de retour. La précision de l'estimation permet de satisfaire un compromis visant à atténuer, à la fois, les cas de surestimation et les cas de sous-estimation des bits requis. Une réduction de la complexité du décodage allant jusqu'à 88%, avec une perte de perfor-

mances raisonnable, consistant en moins de 10% du débit W-Z.

On présente au chapitre 10 une conclusion générale récapitulant les travaux effectués dans cette thèse de doctorat. On y indique les différentes contributions et les publications qui en ont découlées. Des efforts restent à produire et des sentiers restent à explorer pour faire évoluer le paradigme de compression vidéo distribué, pour faciliter son implémentation et promouvoir son déploiement effectif dans l'industrie. On présente, à cet effet, quelques directions pour des investigations futures pouvant octroyer davantage au paradigme DVC de la compétitivité et de la facilité de déploiement.

Chapitre 2

Théorie de l'information et codage vidéo distribué

2.1 Introduction

La théorie d'information peut être considérée comme la "science" qui permet d'envoyer les données efficacement et avec fiabilité, le plus fidèlement et au moindre coût possible. Dans ce contexte, s'inscrivent les travaux de Claude Shannon qui en 1948, a fait de la théorie de l'information un domaine scientifique [7]. Ainsi, on lui décerne le statut du "père fondateur" de la théorie de communication ou encore de l'ère numérique. Les concepts proposés par Shannon en 1948 [7] et pertinents à notre recherche sont résumés dans les sections suivantes.

2.1.1 Capacité du canal et théorème du codage canal

Le taux binaire de transmission de données ne doit pas dépasser une certaine valeur connue sous le nom de la limite de Shannon, si l'on souhaite assurer la fiabilité du transfert de l'information : cette limite est la capacité du canal. Dans le cas d'un canal gaussien blanc à bruit additif, la capacité est donnée par [14] :

$$C = W \log_2 \left(1 + \frac{P_{signal}}{W N_0}\right) \tag{2.1}$$

où C désigne la capacité du canal en bits par seconde, W désigne la largeur de bande du canal en hertz (la largeur de la bande passante dans le cas d'un signal modulé), P_{signal} désigne la puissance totale reçue sur la largeur de bande du signal et elle est mesurée en watt ou en volt^2 et N_0 désigne le niveau de l'energie du bruit en watt $\times$ seconde. $\frac{P_{signal}}{WN_0}$ désigne alors le rapport signal sur bruit. Ce concept annonce qu'il existe, malheureusement, une limite au-dessus de laquelle on ne peut assurer une communication sans erreurs, indépendamment de la sophistication de la technique de codage canal. On ne peut pas transmettre plus rapidement que la capacité du canal sans perte d'information. D'un autre côté, ce concept annonce que pour un débit de transmission au-dessous de la capacité du

canal, il est possible de transmettre l'information aussi fiablement que l'on souhaite moyennant des codes de longueurs importantes.

2.1.2 Architecture d'un système de communications

Dans son article de 1948 [7], Shannon présente un modèle schématique de communication reproduit à la figure 2.1. On remarque à partir de ce modèle que l'on peut complètement séparer le design du codage source du design du codage canal. Le codage de source s'inscrit dans un but de compression et d'élimination de la redondance inutile et le codage canal s'inscrit dans un but d'ajout d'une redondance utile pouvant être exploitée au décodeur pour la correction d'erreurs.

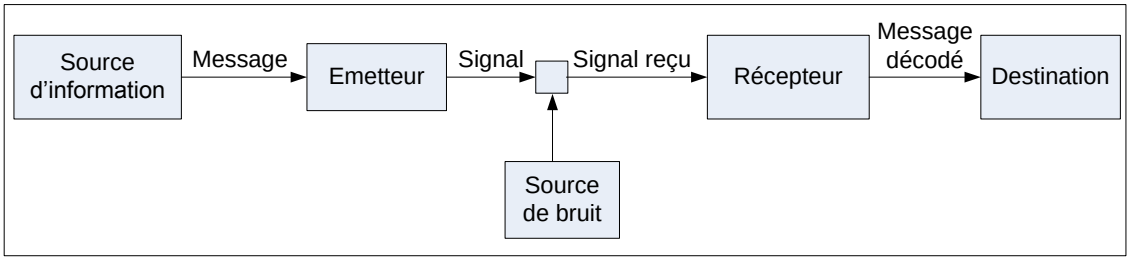

FIGURE 2.1: *Schéma bloc d'un système générique de communications [7].*

2.1.3 Représentation binaire

L'idée de base des travaux de Shannon repose sur la possibilité de formuler tout message en une série de 0s ou de 1s, ou encore le nombre de questions que l'on doit poser pour réduire le nombre de possibilités à juste une seule, éliminant ainsi toute incertitude [15] . Selon le théorème de Shannon (connu encore sous le nom théorème de Nyquist-Shannon) toute information peut être numérisée de façon fiable en autant que la fréquence d'échantillonnage soit supérieure au double de la plus haute fréquence du signal.

2.1.4 Codage source

L'objectif du codage source est de représenter efficacement les données avant de les transmettre en éliminant la redondance se trouvant de la source. Le codage source est actuellement considéré synonyme de compression. Le code Shannon-Fano introduit par Shannon n'était pas optimal dans la minimisation de la longueur du code. Trois années plus tard, David Huffman, un étudiant au cours du professeur Fano au MIT découvre un design de code tant attendu par Shannon. Le code de Huffman, est largement utilisé pour la compression sans perte des données. Les normes JPEG et MP3 ne sont que quelques exemples de méthodes de compressions avec pertes [16].

2.1.5 Entropie

Pour quantifier la mesure d'information contenue dans un message, Shannon a introduit la notion d'entropie, $H(X)$, comme évaluation de la quantité d'incertitude qui peut être éliminée lors de la réception du signal. Plus l'information est probable, plus l'entropie est faible et plus l'information est aléatoire plus l'entropie est importante. Considérons la transmission d'une séquence d'information aléatoire X. Pour assurer une reproduction fiable de X à la réception (transmission sans perte), on doit transmettre à un débit de R_X bits par caractère dépassant ou égalant son incertitude $H(X)$.

$$R_X \geq H(X) \tag{2.2}$$

2.2 Codage de sources corrélées sans pertes

Le paradigme de codage de source distribué, énoncé en 1965 par Slepian-Wolf [2], généralise le résultat de Shannon de la transmission fiable (sans perte) d'une seule séquence d'information discrète et aléatoire X, pour le cas d'une paire de séquences X et Y présentant une certaine corrélation. L'objectif du codage de sources distribuées est de parvenir à une compression optimale et sans perte de ces 2 sources corrélées mais se trouvant dans des endroits différents : il n'y a pas donc de communication mutuelle entre les sources lors de l'encodage mais supposément le décodage se fait de façon conjointe. Le cas d'un décodage disjoint, on retrouve le même contexte de compression présenté par le théorème de Shannon pour une seule source d'information.

Slepian et Wolf ont démontré qu'en exploitant la corrélation entre les 2 sources délocalisées (distribuées), on peut atteindre un résultat similaire à celui que l'on trouve lors de leur traitement conjoint au codage. Avec la seule connaissance de la corrélation entre les sources à l'émission, on peut réaliser un taux de compression optimal sans avoir à transmettre 2 fois l'information redondante se trouvant dans la première source comme dans la deuxième. Le théorème de Slepian-Wolf répond à la question suivante : comment peut-on connaître l'information redondante si les deux sources ne communiquent pas à l'émission ?

2.2.1 Résultat de Slepian-Wolf

Considérons deux séquences d'information X_i et Y_i générées par un tirage répétitif et indépendant d'une paire de variables aléatoires discrètes, X et Y, suivant une distribution jointe $P_{XY}(x,y)$. Le résultat de Slepian-Wolf [2], consiste à déterminer le nombre minimum de bits par caractère nécessaires pour transmettre fidèlement ces 2 séquences codées disjointement mais conjointement décodées, comme le montre la figure 2.2.

Les résultats seront présentés sous forme de régions d'admissibilité, tracées dans le plan $R_X - R_Y$, et délimitant la zone du plan dans laquelle on peut envoyer les deux séquences avec fiabilité. À titre d'exemple, considérons le scénario décrit à la figure 2.2 de codage disjoint et de décodage conjoint de

deux sources corrélées. La région d'admissibilité correspond à la partie hachurée de la figure 2.3, et définie par les conditions suivantes :

$$\begin{cases} R_X \geq H(X|Y) \\ R_Y \geq H(Y|X) \\ R_X + R_Y \geq H(X,Y) \end{cases} \tag{2.3}$$

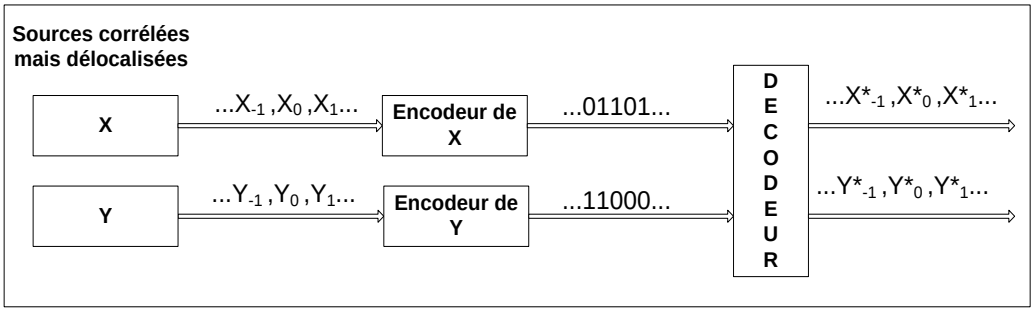

FIGURE 2.2: *Codage de sources corrélées mais délocalisées.*

À partir de *l'étoile bleue* se trouvant dans la région d'admissiblité de la figure 2.3, on constate que l'on peut transmettre la source X à un débit inférieur à son entropie $R_X < H(X)$ quitte à envoyer la source Y avec un débit respectant le critère de Shannon $R_Y > H(Y)$. Ainsi, malgré que chaque codeur ne voit que sa propre source (X ou Y), la corrélation entre les deux sources peut être exploitée au décodage. On n'est plus restreint au théorème de Shannon qui exige que chaque source soit transmise avec un débit supérieur ou égal à son entropie si le décodage des deux sources se fait conjointement.

2.2.2 Région d'admissibilité de deux sources corrélées

Le résultat de Slepian-Wolf définit les zones d'admissibilité des différentes situations du codage de deux sources corrélées soit localisées, soit délocalisées et de décodage conjoint ou disjoint. Ces différentes situations, au nombre de $2^4 = 16$, sont déterminées par la position des 4 interrupteurs de la figure 2.4.

Les zones d'admissibilité relatives aux différentes 16 positions des interrupteurs sont présentées dans la figure 2.5 sans oublier le cas le plus important de la figure 2.3.

2.2.3 Codage de source sans perte avec information latérale

Reconsidérons le cas particulier du schéma de compression de Slepian-Wolf (S-W) indiqué par *l'étoile bleue* à la figure 2.3. Le signal Y est envoyé en premier et reconstruit parfaitement au décodeur vu que le débit utilisée respecte Shannon : $R_Y > H(Y)$. Étant donné que les signaux X et Y sont corrélés, la reconstruction de Y, fournit une certaine information à propos de X. Dans ce scénario, Y peut être

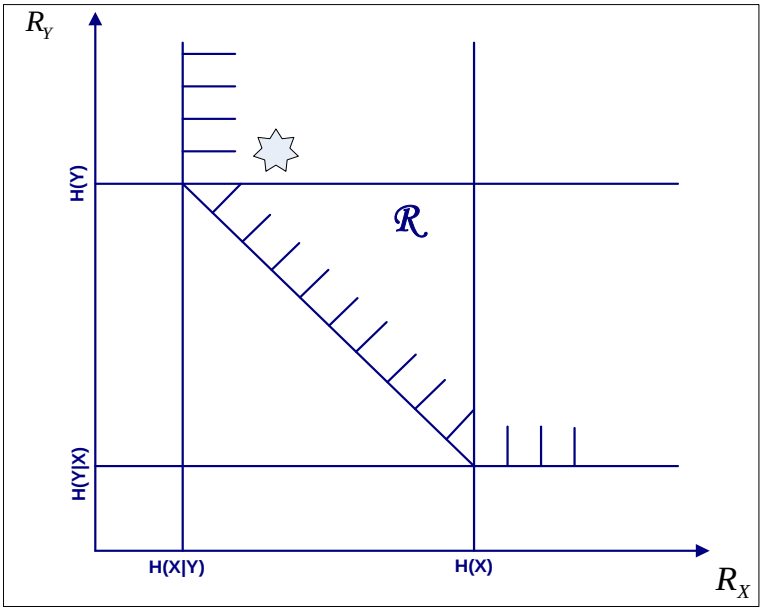

FIGURE 2.3: *Région de débit admissible correspondant à la figure 2.2.*

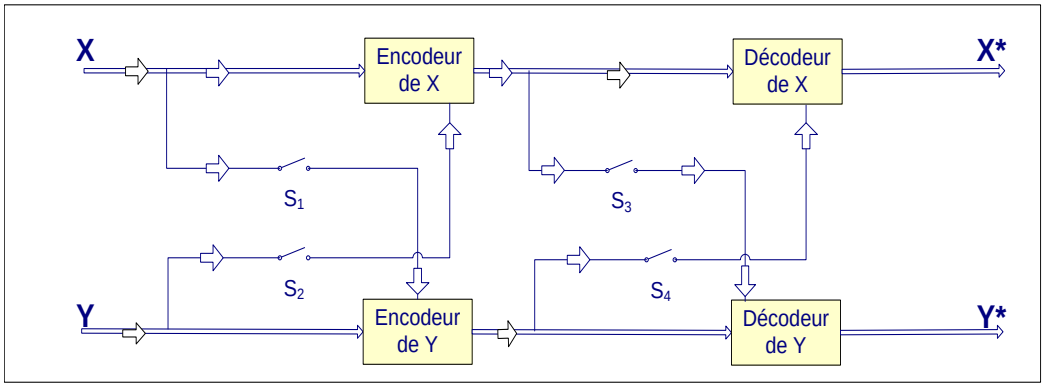

FIGURE 2.4: *Les 16 situations de codage de sources corrélées.*

dénoté *information latérale* disponible du côté de la réception pour le décodage de X. L'encodage de X nécessite, alors, un débit R_X tel que $H(X|Y) \leq R_X \leq H(X)$ si le décodeur dispose de l'information latérale Y. Sans cette information latérale le débit devait être $R_X > H(X)$. Ce scénario est connu sous le nom compression sans perte de X avec une information latérale Y, disponible uniquement au décodeur.

Étant donné que le signal X est statistiquement lié à l'information latérale Y, celui-ci peut être considéré comme la sortie systématique d'un canal virtuel avec X appliqué à l'entrée. Les techniques de codage de canal peuvent alors être utilisées pour générer les bits de parité et décoder, avec Y comme information latérale, la séquence X. Ici, les techniques de codage de canal sont ainsi utilisées à des

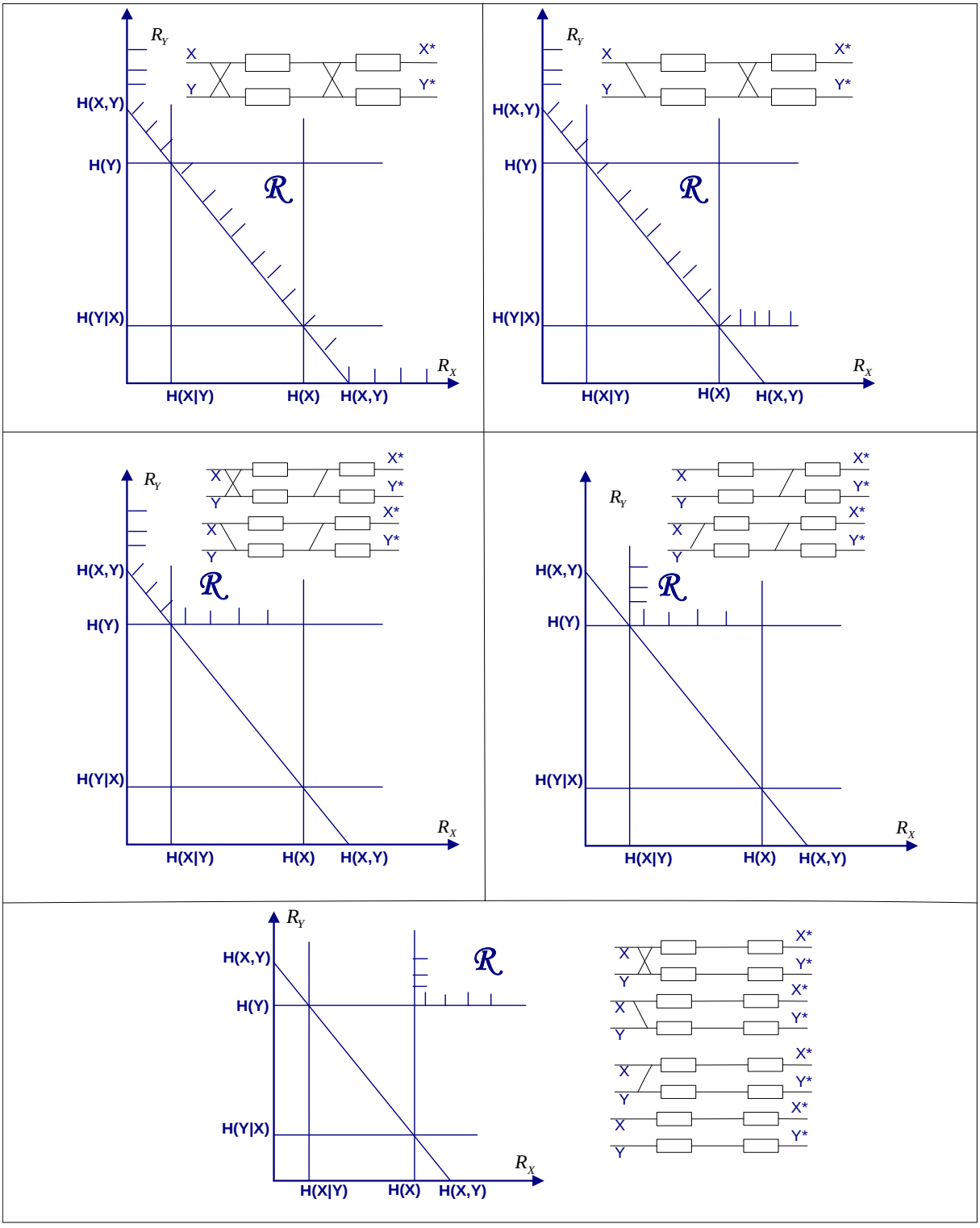

FIGURE 2.5: *Les régions d'admissibilité présentées dans [2].*

fins de codage de source.

2.3 Codage de source à perte avec information latérale

2.3.1 Codage de source avec perte

La plupart des systèmes de compression tolèrent un certain niveau de distorsion entre l'information originale et l'information reçue. Wyner et Ziv se sont intéressés dans leurs travaux [3],[17], à ce système de compression avec pertes, en définissant la fonction débit distorsion $R^*(D)$ qui constitue le débit minimum requis pour pouvoir assurer une reconstruction ne dépassant pas en sa moyenne la distorsion D. Dans la communauté de la théorie d'information, on désigne ce principe par compression de source avec perte.

2.3.2 Résultat de Wyner-Ziv

Ce principe est encore généralisé lorsqu'une information latérale Y, présentant une certaine corrélation avec l'information à transmettre X, est disponible au niveau du récepteur. Le théorème de Wyner-Ziv (W-Z) stipule que cette information latérale peut être exploitée pour réduire le débit de transmission sans augmenter la distorsion. On parle du principe de codage de source à perte avec information latérale qui se base sur les théorèmes de S-W et de W-Z comme on peut le voir à la figure 2.6. $R_{X|Y}(D)$ désigne le débit d'encodage de la source X si l'information latérale est disponible à la fois à l'encodeur et au décodeur et $R^{WZ}_{X|Y}(D)$ désigne le débit d'encodage de la source X si l'information latérale est disponible uniquement au décodeur. La distorsion D dépend du pas de quantification. Wyner et Ziv ont démontré que :

$$R^{WZ}_{X|Y}(D) \geq R_{X|Y}(D), \tag{2.4}$$

avec l'égalité, i.e. $R^{WZ}_{X|Y}(D) = R_{X|Y}(D)$, si les sources X et Y sont conjointement gaussiennes et si la mesure de distorsion consiste en l'erreur quadratique moyenne.

2.3.3 Codec Wyner-Ziv

Un encodeur Wyner-Ziv est alors une extension du codeur Slepian-Wolf à l'entrée duquel on concatène un dispositif de transformation à perte, tel que montré à la figure 2.7. Le décodeur dispose d'une information latérale Y qui présente une certaine corrélation avec la source X. Cette corrélation peut être vue comme une forme de canal virtuel dont l'entrée est la source X et dont la sortie est l'information latérale Y. Dans ce cas Y est une version bruitée de X et peut servir à réduire le débit de transmission de X, assurant ainsi de la compression. Le codage vidéo est l'un des domaines pratiques où l'on peut faire appel aux principes du codage source distribué. En effet, on peut exploiter la redondance qui se trouve entre les trames reçues successivement pour la génération de l'infomation latérale. Dans la section suivante on expliquera ce point avec plus de détails en mettant l'accent sur le lien entre le codage source distribué et la compression vidéo.

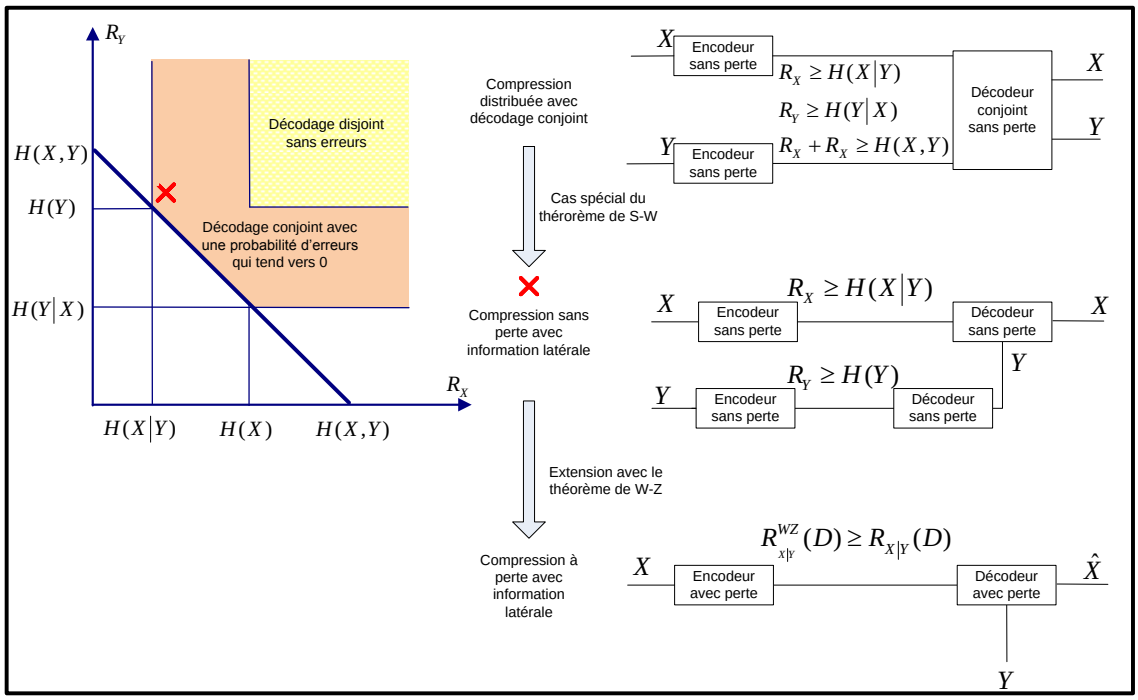

FIGURE 2.6: *Concept de compression de sources distribuées.*

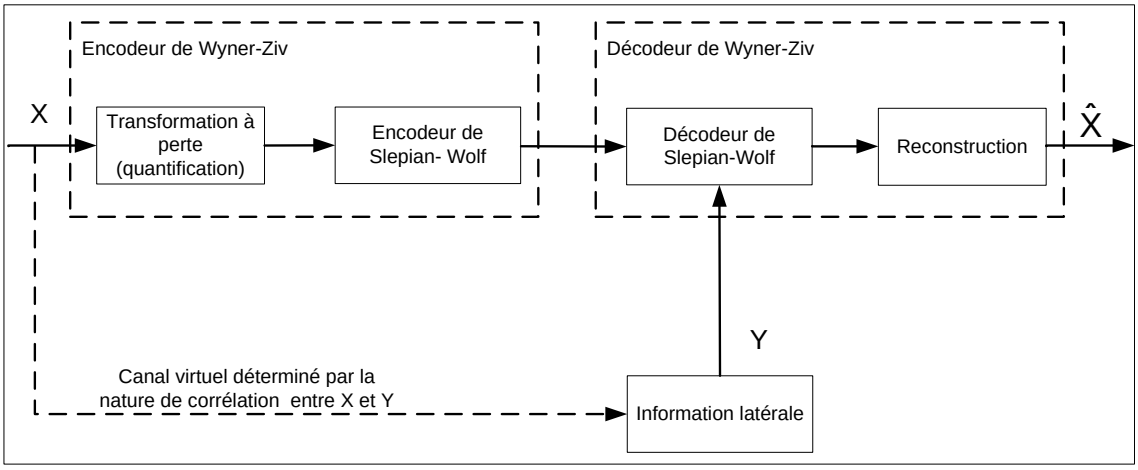

FIGURE 2.7: *Codeur et décodeur Wyner-Ziv.*

2.4 Vidéo numérique

Une séquence vidéo est une succession d'images défilant à une cadence dépassant la capacité de distinction de l'oeil humain qui est de 20 images par seconde. Ainsi il est possible de tromper la perception oculaire et lui faire croire à une image animée.
La vidéo numérique vient remplacer la vidéo analogique qui représente l'information comme un flux

continu de données qui va s'afficher à l'écran de télévision par le principe de balayage. Le standard vidéo numérique dispose de plus de flexibilité de traitement contrairement à son précédent. Cette flexibilité est primordiale pour assurer les attentes croissantes du public en matière d'audiovisuel. Le public audiovisuel se fait de plus en plus interactif avec la vidéo : on peut visionner des vidéos soit devant un écran de téléviseur soit à travers un écran d'un cellulaire, on peut aussi télécharger ou téléverser des vidéos. Les communications vidéo en temps réel sont devenues très courantes.

La vidéo numérique est le médium idéal pour ce genre d'applications car elle se prête mieux au traitement, au stokage et à la compression. L'image numérique, quant à elle, est divisée en pixels codés dans un espace colorimétrique RGB (Red, Green, Blue), correspondant aux 3 couleurs primaires dont la combinaison permet de produire à l'écran une grande partie du spectre visible. Chaque couleur est codée sur 8 bits spécifiant 256 niveaux d'intensité. Chaque pixel est donc codé par 8×3 =24 bits pour un choix de couleurs de 2^{24} possibilités soit 16 millions de couleurs.

2.4.1 Sous-échantillonnage

Le système YUV est un espace de colorimétrique qui encode une image ou une vidéo couleur en prenant en compte la perception humaine, réduisant ainsi la bande passante pour les composantes de chrominance (Cr,Cb), ce qui permet généralement de masquer des erreurs de transmission et les artefacts de compression à la perception humaine basée sur une représentation RGB [18]. L'oeil humain est moins sensible aux variations des composantes de chrominance U et V. Pour assurer un certain degré de compression, on utilise une technique de sous-échantillonnage en chrominance. Cette technique consiste à supprimer certaines valeurs de la chrominance dans un groupe de 4×4 pixels [19]. Les types de sous-échantillonnages de la chrominance sont [20] :

- 4 :4 :4 $\Rightarrow$ qu'il n'y a aucun sous-échantillonnage.
- 4 :2 :2 $\Rightarrow$ que pour chaque 2×2 échantillons de luminance Y, on a seulement 2 composantes de chrominance Cr et 2 composantes de chrominance Cb.
- 4 :2 :0 $\Rightarrow$ que pour chaque 2×2 echantillons de luminance Y, on a 1 composante de chrominance CR et 1 composante de chrominance Cb.

2.4.2 Compression vidéo

La compression vidéo consiste en la réduction de la quantité de données numériques utilisées pour la représentation des images vidéo. Elle se base sur une combinaison de la compression spatiale des images et de la compensation temporelle du mouvement.
La compression vidéo opère généralement sur des groupes de pixels voisins de forme carrée, souvent appelé *macroblocs*. L'encodeur compare ces groupes de pixels dans deux trames successives et envoie seulement les différences entre ces blocs. Pour effectuer cet encodage temporel il existe 3 types d'images [21] :

– Les images I ou images intra : elles sont appelées ainsi parce qu'elles peuvent être décodées indépendamment de toute autre trame.

– Les images P ou images prédictives : elles sont aussi appelées inter images. Ce type d'images permet d'améliorer la compression en exploitant la redondance temporelle d'une séquence vidéo. Les images P contiennent seulement la différence entre l'image qui la précède qui peut être soit de type I ou de même type P.

– Les images B est une abréviation pour les images bidirectionnelles. Ce type d'image est semblable aux images P sauf qu'elles sont décodées en utilisant à la fois l'image précédente et l'image suivante.

2.4.3 Compression vidéo distribuée

La compression vidéo est généralement assurée par l'encodeur qui exploite les statistiques de la source pour réduire le débit de transmission. L'encodeur calcule la différence entre les trames et envoie les images Intra suivies des images prédictives contenant cette différence. Ce principe est parfaitement convenable pour les applications de transmission vidéo standard où l'encodeur est assez puissant pour supporter la complexité d'encodage. Cependant dans certains cas, comme par exemple l'envoi de séquences vidéo d'un cellulaire vers une station de base fixe, le scénario dual de compression vidéo est requis. Le traitement de compression ne peut plus être assuré par la caméra où la mémoire et la capacité calculatoire sont faibles. Dans cette situation, le système a besoin d'un encodeur à faible complexité au détriment d'un décodeur à forte complexité.

Selon les deux résultats de la théorie d'information de Slepian-Wolf et Wyner-Ziv, on peut assurer le transfert de complexité en utilisant un encodeur produisant des images intra (ou encore des images Wyner-Ziv) et un décodeur inter image disposant d'une information adjointe : une image intercorrélée avec la source. Dans ce travail, on applique le codage Wyner-Ziv à un cas réel de transmission de séquence vidéo en utilisant un turbo code comme codeur et décodeur Slepian-Wolf. A la section suivante, on présente en détail les différents composants du système de codage Wyner-Ziv.

2.5 Encodeur et décodeur vidéo de Wyner-Ziv

On présente à la figure 2.8 un encodeur intra image, connu sous l'appelation encodeur Wyner-Ziv, et un décodeur inter image appelé décodeur Wyner-Ziv. Ainsi, la complexité de l'exploitation de la corrélation entre les trames successives est assurée par le décodeur. L'émetteur se contente d'encoder les images de façon indépendante sans avoir à supporter le calcul des vecteurs de mouvement comme c'est le cas dans le schéma de codage conventionnel. D'où le transfert de la complexité calculatoire de l'encodeur vers le décodeur. L'encodeur Wyner-Ziv se compose d'un quantificateur (transformation à perte) et d'un encodeur Slepian-Wolf (i.e. sans pertes). Quant au décodeur Wyner-Ziv, il est constitué d'un décodeur Slepian-Wolf et d'un système de reconstruction. Le codeur et décodeur Slepian-Wolf utilisés reposent ici sur un code turbo et l'information latérale est utilisée par le turbo décodeur sans

qu'elle ne soit quantifiée [22].

Soient X_1, X_2,...,X_N, les trames d'une séquence vidéo. Les trames impaires $\{X_{2i+1}\}$ sont des trames intra envoyées suivant le standard vidéo habituel ; par exemple, selon la norme H.264. Pour simplifier le problème, on suppose que ces trames sont parfaitement reconstruites au niveau du décodeur. Les trames paires $\{X_{2i}\}$ sont encodées indépendamment des images intra et des autres trames paires.

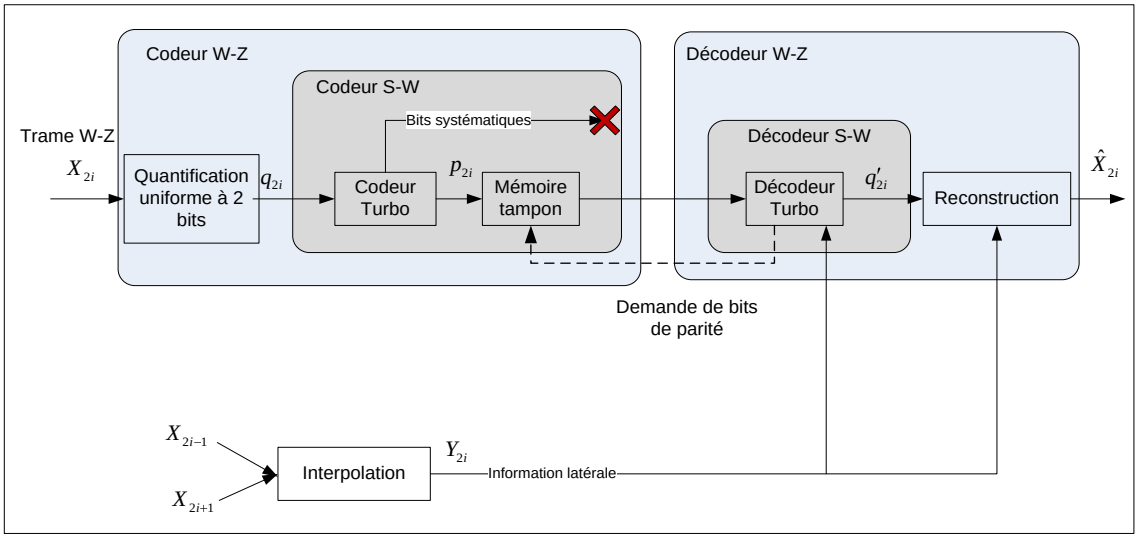

FIGURE 2.8: *Schéma bloc de la compression vidéo distribuée.*

2.5.1 Quantification et turbo codage

Les trames paires $\{X_{2i}\}$ sont introduites pixel par pixel (chaque pixel étant représenté par un mot de 8 bits) dans un quantificateur uniforme (voir figure 2.9 pour $2^M = 4$ niveaux). L'utilisation du code de Gray permet d'améliorer les performances du système dans la mesure que deux niveaux de quantification voisins ne diffèrent que d'un seul bit mais pour un but de comparaison et validation on garde le même schéma de quantification uniforme proposé par [22]. La séquence binaire est ensuite encodée avec un turbo codeur formé de deux codeurs convolutifs systématiques identiques comme le montre la figure 2.10. Avant de passer la séquence binaire au second codeur convolutif, un entrelacement est effectué au niveau symboles formés par deux bits chacun. La fonction de permutation prend alors en entrée les bits deux à deux. Finalement, seuls les bits de parités sont transmis au décodeur.

2.5.2 Turbo décodage avec information latérale

La préparation et l'utilisation de l'information latérale de la trame X_{2i} se fait comme suit. Le bloc d'interpolation se trouvant au niveau du décodeur prend les trames adjacentes X_{2i-1} et X_{2i+1} et effectue une interpolation temporelle résultant en l'information latérale Y_{2i}. Cette trame interpolée est par la suite utilisée par le turbo décodeur pour remplacer le rôle des bits systématiques dans l'opération de décodage. Il faut bien remarquer que dans ce système, la trame interpolée n'est pas quantifiée et

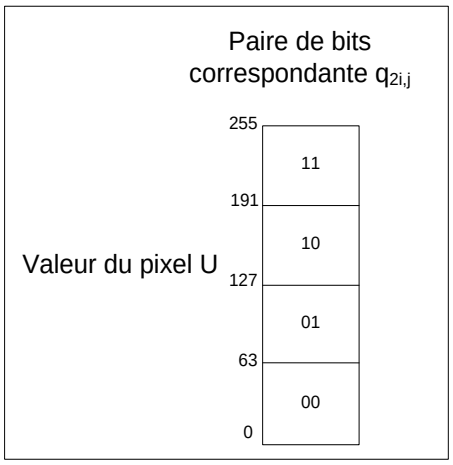

FIGURE 2.9: *Quantification uniforme du pixel avec 2 bits.*

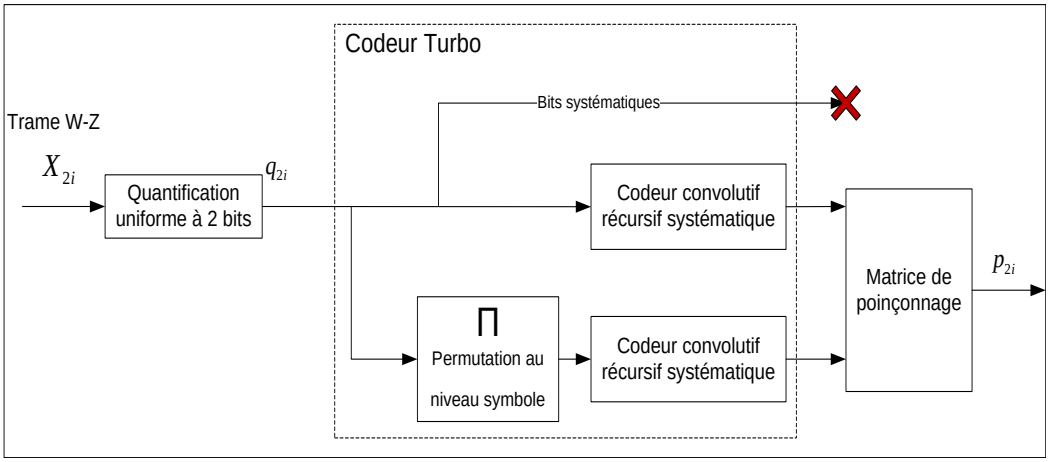

FIGURE 2.10: *Codeur turbo.*

sa valeur correspond à la valeur du pixel en question. Pour raffiner cette version bruitée, le turbo décodeur utilise aussi les bits de parité envoyés effectivement dans un canal réel. Mais en premier lieu, cette version bruitée Y_{2i} va être utilisée par le turbo décodeur afin de déterminer la valeur du symbole envoyé, q_{2i}. On présente dans le chapitre suivant un exemple détaillé expliquant comment le turbo décodage est effectué et comment l'information latérale sous forme d'une séquence de pixels peut être utilisée pour le décodage d'une séquence de bits.

Dans ce système, l'envoi des bits de parité est supposé sans erreur. Pour assurer une meilleure adaptation au changement statistique entre l'information latérale et la trame originale, l'envoi des bits de parité se fait graduellement sous demande du décodeur, par le biais d'un système de poinçonnage. Avec cette structure le codeur n'envoie que le minimum des bits de parité nécessaires pour décoder correctement la séquence binaire. Si le décodeur ne peut pas assurer une probabilité d'erreur au-

dessous d'un seuil, il demande au codeur, par l'utilisation d'un canal de retour, l'envoi de plus de bits de parité.

2.5.3 Reconstruction avec information latérale

Après le décodage turbo, les symboles décodés q'_{2i} et l'information latérale sont utilisés pour la reconstruction de chaque pixel tel que : $\hat{X}_{2i,j} = E\left(X_{2i,j} \middle| q'_{2i,j}, Y_{2i,j}\right)$. La figure 2.11 montre la fonction de reconstruction utilisée pour un nombre de niveaux de quantification $2^M = 4$. Si la valeur du pixel , $Y_{2i,j}$, de l'information latérale correspond au symbole décodé $q'_{2i,j}$, le pixel reconstitué $\hat{X}_{2i,j}$ va prendre la valeur du pixel de la trame interpolée $Y_{2i,j}$. Le cas contraire équivaut à dire que le pixel interpolé $Y_{2i,j}$ ne se trouve pas dans le même intervalle de quantification que le pixel original $X_{2i,j}$. En d'autres termes, l'interpolation de ce pixel a échoué. Ainsi la reconstruction se basera sur la valeur du symbole décodé et le pixel reconstitué $\hat{X}_{2i,j}$ prendra la valeur limite se trouvant dans l'intervalle correspondant à $q'_{2i,j}$. Ainsi la fonction de reconstruction permet de limiter l'amplitude de la distorsion à une valeur maximale. Cette propriété permet d'éviter les erreurs perceptuelles importantes qui peuvent affecter la qualité objective des images. Ainsi quand la corrélation entre l'information latérale et la trame originale est faible, la fonction de reconstruction se basera sur le symbole décodé et non pas sur l'information latérale.

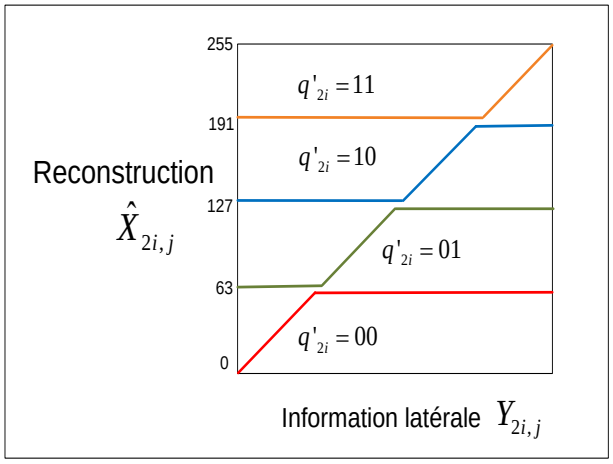

FIGURE 2.11: *Fonction de reconstruction pour un modèle de corrélation Laplacienne.*

2.6 Fondement et évolution du codage vidéo distribué

Le codage vidéo numérique conventionnel n'a pas cessé d'évoluer afin d'atteindre des performances de compression de plus en plus élevées en mettant en oeuvre des techniques sophistiquées et complexes d'estimation de mouvement. Ces techniques sont accomplies par l'encodeur qui doit être adapté à cette tâche calculatoire importante. Le décodeur, cependant, peut facilement reconstituer la vidéo en exploitant les vecteurs de mouvement calculés par l'encodeur. Cette répartition de tâches est bien

adaptée aux applications courantes de transfert des séquences vidéo comme la diffusion vidéo, la diffusion de flux vidéo en continue (streaming). En effet l'encodeur bénéficie d'une puissance calculatoire élevée pour assurer la compression, à une seule reprise, de la séquence vidéo envoyée à plusieurs appareils à puissance calculatoire limitée. Nonobstant, depuis l'émergence des caméras de surveillance vidéo sans fils localement distribuées et la croissance d'utilisation des services vidéo dans les cellulaires et de nombreuses autres applications impliquant plusieurs encodeurs vidéo à faible coût aux dépends d'un décodeur centralisé à haute complexité, la norme de codage vidéo traditionnelle (standard H.264/AVC [1]) a été révisée et la répartition des tâches inversée.

Le résultat de Slepian-Wolf sur le codage sans perte de sources distribuées [2] et son extension pour le cas de codage de source avec perte en présence d'une information latérale au décodeur par le résultat de Wyner-Ziv [3], constitue les bases théoriques du codage de sources distribuées. Ces résultats théoriques datant des années 70, ont donné naissance à un champ vaste d'applications récentes dont la principale consiste en le paradigme du codage vidéo distribué établi en 2002. L'objectif fondamental du codage vidéo distribué est d'établir une technique de transmission de flux vidéo permettant le transfert de la complexité calculatoire de compression de l'encodeur vers le décodeur. Ce dernier se chargera de l'estimation et de la compensation de mouvement par la génération de l'information latérale et assumera ainsi les complexités de calcul inhérentes. À cet effet, une première solution a été proposée par un groupe de recherche de l'Université de Californie à Berkeley [23, 24] et une seconde solution a été proposé par un groupe de recherche de l'Université de Stanford [22, 25]. Cette dernière solution a suscité le plus d'intérêt de la part des groupes de recherche et a été intensivement étudiée. Plusieurs directions ont été poursuivies afin de parvenir à de meilleures performances de débit distorsion dont les principales consistent en ce qui suit.

La première architecture de codage vidéo distribué développée à l'Université de Standford considère la séquence vidéo dans le domaine pixel ([22]). Chaque pixel est passé dans un quantificateur scalaire à 16 niveaux (4 bits) qui consiste en une transformation à perte réduisant la taille du pixel de 8 bits à 4 bits. Ces 4 bits sont ensuite transférés aux entrées d'un codeur turbo formé de deux encodeurs convolutifs récursifs systématiques de rendement 4/5. L'extraction des différents plans de bits dans ce schéma n'est pas bien appropriée car les 4 bits de quantification d'un pixel donné sont considérés ensemble par chacun des deux encodeurs de rendement 4/5 et un bit de parité leur est alloué. Le fait de garder ensemble les 4 bits de quantification d'un pixel, permet au décodeur turbo, utilisant la valeur du pixel dans la trame interpolée à la place des bits systématique, d'éviter la marginalisation lors du calcul des valeurs de vraisemblance canal. Ce groupe de recherche précise dans [26] que le codage turbo au niveau symbole (avec 2 RSCs 4/5) et que le codage turbo avec extraction des plans de bits (avec 2 RSC 1/2), des valeurs des pixels quantifiés donnent des résultats similaires. Dans cet article [26], les auteurs étendent l'architecture de codage vidéo distribué dans le domaine transformé avec la même transformée en cosinus discret (en bloc 4×4) utilisée dans le standard H.264. Cette transformation donne la possibilité d'exploiter les dépendances statistiques à travers une trame pour atteindre de meilleures performance de débit-distorsion. En effet la majeure partie de l'information

est concentrée dans le coin supérieur gauche du bloc 4×4. Cette information correspond aux basses fréquences auxquelles le système visuel humain (VHS : Visual Human System) est le plus sensible [20].

Les auteurs de [11], ont proposé une amélioration du système de codage Wyner-Ziv dans le domaine transformée en remplaçant le quantificateur scalaire par un quantificateur avec un intervalle symétrique autour de zéro pour les coefficients AC et en ajustant le pas de quantification en fonction de la plage dynamique de chaque bande de coefficient DCT. Les coefficients DC, par contre, prennent en général des valeurs positives élevées vu qu'ils représentent l'énergie moyenne du bloc 4×4. Ils sont quantifiés, alors, en utilisant un quantificateur scalaire uniforme sans un intervalle de quantification symétrique autour de zéro.

Par ailleurs, l'amélioration de la technique d'interpolation pour la génération de l'information latérale constitue incontestablement l'un des moyens les plus efficaces pour obtenir de meilleures performances. En effet, l'amélioration de la trame interpolée revient à faire augmenter la corrélation entre l'information latérale et la trame originale. Autrement dit, ceci revient à diminuer le niveau de bruit du canal virtuel. Dans [13], les auteurs proposent une technique bidirectionnelle d'estimation de mouvement et un algorithme de lissage spatial de mouvement pour l'élimination des vecteurs de mouvement erronés. D'autres techniques plus élaborées de génération de l'information latérale sont proposées dans [27, 28, 29].

La modélisation du bruit de corrélation du canal virtuel est aussi un axe prometteur pour l'amélioration des performances du schéma DVC. La corrélation entre la trame interpolée et la trame originale est régie par une distribution Laplacienne de paramètre α souvent considéré de valeur égale à 1/2. Une évaluation précise de ce paramètre favorisera la convergence du processus de décodage turbo par l'utilisation de moins de bits de parité. Pour déterminer le paramètre α, il faut observer la différence entre la trame originale et la trame interpolée, sauf que le décodeur n'a pas accès à l'information originale et l'encodeur n'a pas accès à l'information latérale. Donc, ni l'un ni l'autre ne sera en mesure d'évaluer ce paramètre. Dans les travaux effectués dans [8] et [30], les auteurs présentent une technique pour la modélisation du bruit de corrélation au niveau du décodeur. L'estimation du paramètre α se fait par l'exploitation des trames clés précédentes et suivantes. Dans ces travaux ce paramètre est estimé pour différents niveaux de granularité : pour chaque trame, pour chaque bloc 8 par 8 et pour chaque pixel.

Divers autres axes de recherche se rapportant au codage vidéo distribué figurent dans la littérature pour améliorer chacun des différentes composantes de l'architecture globale. Dans ce contexte, le bloc de reconstruction a été aussi l'objet d'étude et d'amélioration [31, 32, 33]. L'objectif des travaux sur le codage vidéo distribué ne se limite pas uniquement à l'amélioration des performances débit-distorsion mais en plus englobe aussi la considération de l'aspect pratique pour pouvoir implémenter cette architecture dans des applications réelles. Le canal de retour est certainement l'élément architectural le plus controversé vu qu'il introduit un délai important de décodage. Pour soulever ce problème, des travaux ont été proposés dans la littérature en transférant le contrôle du débit de transmission à l'encodeur

[30] et [34].

2.7 Conclusion

Dans ce chapitre, on a présenté les 3 principaux apports de la théorie d'information qui constituent le fondement du codage source distribué et qui présentent les limites théoriques que peut atteindre l'efficacité d'un système de communication en matière de réduction du débit nécessaire pour une distorsion donnée. Ces limites théoriques motivent les chercheurs dans le domaine du codage distribué à trouver de meilleures techniques de compression pour s'en approcher. Les techniques actuelles les plus performantes pour remplacer le codeur et le décodeur Slepian-Wolf (voir la figure 2.7) utilisent les turbo-codes [40] et les codes LDPC [41]. On a aussi démontré comment la compression vidéo puisse être un domaine qui permette de mettre en pratique la théorie du codage de source distribuée. L'axe de recherche qui rassemble ces données théoriques avec la compression vidéo est connu sous le nom de codage vidéo distribué. On a présenté alors dans la dernière partie de ce chapitre les fondements, les travaux et les progrès effectués dans cette direction de recherche relativement récente. On s'intéressera à partir du chapitre suivant à l'implémentation de l'architecture de codage vidéo distribué en s'intéressant, tout d'abord, au codeur/décodeur Slepian-Wolf basé sur les codes-turbos.

Chapitre 3

Codeur/décodeur Slepian-Wolf basé sur les turbos codes

3.1 Introduction

Dans ce chapitre on s'intéresse à la partie la plus délicate du schéma de codage vidéo Wyner-Ziv à savoir le codeur et le décodeur Slepian-Wolf. Pour ce genre d'applications les turbo codes sont les mieux adaptés. On commence tout d'abord par l'implémentation de la partie encodage pour deux codeurs turbo différents. Le premier codeur correspond au codeur turbo 3GPP utilisé dans [42] : il est constitué de deux codeurs convolutifs systématiques récursifs de rendement 1/2. Le second codeur est constitué de deux codeurs convolutifs systématiques récursifs de rendement 4/5 et correspond au codeur turbo utilisé dans [22]. Le codeur turbo avec deux RSCs de rendement 4/5 offre une meilleure compression au prix d'une complexité élevée au décodage. Un système de poinçonnage est introduit à la sortie de l'encodeur turbo pour assurer une transmission progressive, selon le besoin, des bits de parité. La période de la matrice de poinçonnage pour un code turbo avec 2 RSCs 4/5 est de 8 alors qu'elle est de 32 pour un code turbo avec 2 RSCs 1/2. Une étude des statistiques du canal virtuel est aussi effectuée dans ce chapitre en examinant la différence entre les trames interpolées et les trames originales pour la totalité de la séquence vidéo `Foreman`. Cette étude démontre que le modèle de corrélation est bien représenté par une distribution Laplacienne.

Par la suite on s'intéresse au processus de décodage turbo dont la partie centrale du décodage turbo repose sur l'algorithme BCJR log-MAP [4]. L'application de cet l'algorithme dans le cas du décodeur S-W proposé dans [22], présente deux particularités. La première étant le fait que le canal virtuel affecte la trame "*reçue*" avec un bruit de nature Laplacienne. La deuxième différence, quant à elle, réside dans le fait que le décodeur log-MAP utilise les valeurs des pixels de la trame interpolée (reçue par le canal virtuel) pour décoder une séquence binaire et non les bits systématiques envoyés dans un canal. On s'intéressera à l'implémentation de l'algorithme de décodage BCJR log-MAP qui diffère selon qu'il s'agisse d'un code turbo avec 2 RSCs de rendement 1/2 ou de rendement 4/5 en traitant

chaque cas à part.

3.2 Codeur convolutif récursif systématique

3.2.1 Codeur RSC 1/2

On étudie dans cette partie, le codeur convolutif récursif systématique de rendement 1/2 que l'on trouve dans le codeur turbo 3GPP (3rd Generation Partnership Project : mobiles de 3e génération). Ce codeur RSC 1/2 est défini par les polynômes direct et de retour suivants :

$$\begin{aligned} &\text{Coefficients directs :} \quad 1+D+D^3 \rightarrow 1101 \rightarrow 15\,(\text{en octal}) \\ &\text{Coefficients de retour :} \quad\ 1+D^2+D^3 \rightarrow 1011 \rightarrow 13\,(\text{en octal}) \end{aligned} \tag{3.1}$$

La matrice génératrice correspondante est donnée par :

$$G = \left[1 \quad \frac{1+D+D^3}{1+D^2+D^3}\right] \tag{3.2}$$

Dans la figure 3.1, on trouve une représentation du codeur RSC 1/2 tel décrit :

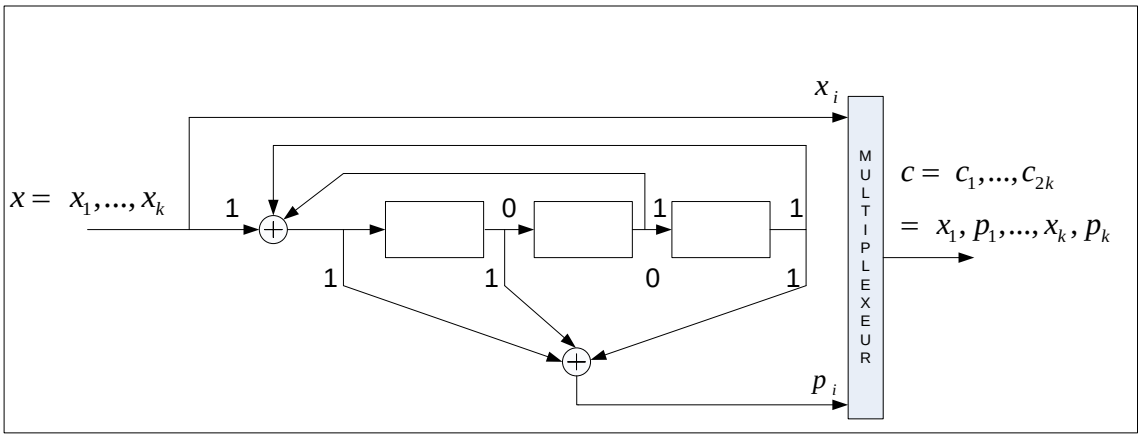

FIGURE 3.1: *Codeur RSC 1/2.*

La représentation du treillis du codeur RSC 1/2 est donné par le tableau 3.1 de l'état suivant et le tableau 3.2 de la sortie générée. Le premier tableau correspond à l'état suivant lorsqu'on part de l'état actuel pour une entrée donnée. Le deuxième tableau correspond à la sortie générée lorsqu'on part de l'état actuel pour une entrée donnée. On ne considère que le bit de parité généré car les bits systématiques ne seront pas envoyés.

3.2.2 Codeur RSC 4/5

Le codeur turbo utilisé dans [22] est constitué de deux codeurs convolutifs systématiques récursifs de taux 4/5. Les coefficients directs et de retour du codeur sont donnés sous forme octale dans le tableau 3.3 [43].

Bits d'entrée État actuel	0	1
000	0	4
001	4	0
010	5	1
011	1	5
100	2	6
101	6	2
110	7	3
111	3	7

Tableau 3.1: *État suivant du diagramme d'états du codeur RSC de taux 1/2.*

Bits d'entrée État actuel	0	1
000	0	1
001	0	1
010	1	0
011	1	0
100	1	0
101	1	0
110	0	1
111	0	1

Tableau 3.2: *Sortie du diagramme d'états du codeur RSC de taux 1/2.*

	$k=0$	$k=1$	$k=2$	$k=3$	$k=4$
$h_k(D)$	23	35	31	37	27

Tableau 3.3: *Coefficient du codeur RSC 4/5.*

La matrice génératrice de ce codeur est donnée par :

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & \frac{D^4+D^3+D^2+1}{D^4+D+1} \\ 0 & 1 & 0 & 0 & \frac{D^4+D^3+1}{D^4+D+1} \\ 0 & 0 & 1 & 0 & \frac{D^4+D^3+D^2+D+1}{D^4+D+1} \\ 0 & 0 & 0 & 1 & \frac{D^4+D^2+D+1}{D^4+D+1} \end{bmatrix} \tag{3.3}$$

Le codeur introduit un délai de 4 coups d'horloge à chacune des 4 entrées. Chaque entrée est alors munie d'une mémoire de 4 cellules. En tout, la taille de la mémoire du codeur est $M = 4 \times 4 = 16$ cellules. Le diagramme d'état comprendra $2^{16} = 65536$ états. Pour simplifier l'implémentation de ce code on utilise la représentation canonique [44] donnée par la figure 3.2. Cette représentation permet

Symboles d'entrée État actuel	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0000	0	2	6	4	5	7	3	1	7	5	1	3	2	0	4	6
0001	9	11	15	13	12	14	10	8	14	12	8	10	11	9	13	15
0010	1	3	7	5	4	6	2	0	6	4	0	2	3	1	5	7
0011	8	10	14	12	13	15	11	9	15	13	9	11	10	8	12	14
0100	2	0	4	6	7	5	1	3	5	7	3	1	0	2	6	4
0101	11	9	13	15	14	12	8	10	12	14	10	2	9	11	15	13
0110	3	1	5	7	6	4	0	2	4	6	2	0	1	9	7	5
0111	10	8	12	14	15	13	9	11	13	15	11	9	8	10	14	12
1000	4	6	2	0	1	3	7	5	3	1	5	7	6	4	0	2
1001	13	15	11	9	8	10	14	12	10	8	12	14	15	13	9	11
1010	5	7	3	1	0	2	6	4	2	0	4	6	7	5	1	3
1011	12	14	10	8	9	11	15	19	11	9	13	15	14	12	8	10
1100	6	4	0	2	3	1	5	7	1	3	7	5	4	6	2	0
1101	15	13	9	11	10	8	12	14	8	10	14	12	13	15	11	9
1110	7	5	1	3	2	0	4	6	0	2	5	4	5	7	3	1
1111	14	12	8	10	11	9	13	15	9	11	15	13	12	14	10	8

Tableau 3.4: *État suivant du diagramme d'états du codeur RSC de rendement 4/5.*

de réduire la taille de la mémoire du codeur à $M = 4$ correspondant à un diagramme d'état de $2^4 = 16$ états.

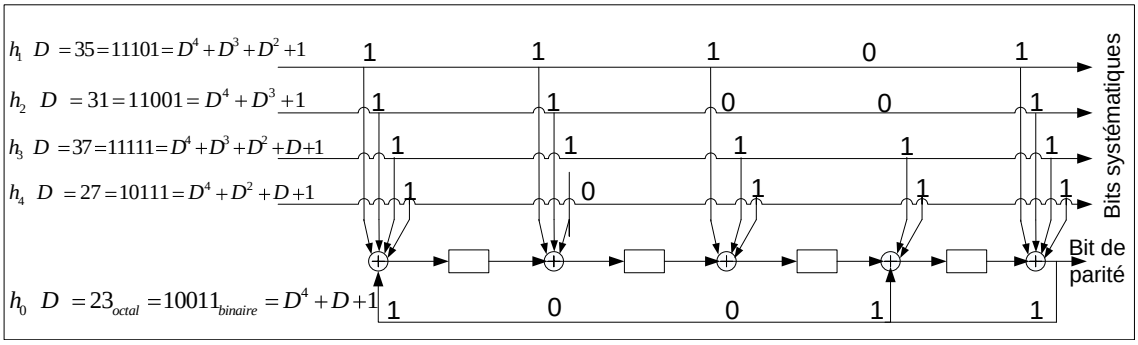

FIGURE 3.2: *Codeur convolutionel récursif systématique sous forme canonique.*

L'implémentation du codeur RSC de rendement 4/5 correspondant à la figure 3.2 requiert la préparation du treillis du code dont la représentation est donnée, comme pour le cas du codeur RSC 1/2, par les deux tableaux 3.4 et 3.5. Le premier tableau correspond à l'état suivant lorsqu'on part de l'état actuel pour une entrée donnée. Le deuxième tableau correspond à la sortie générée lorsqu'on part de l'état actuel pour une entrée donnée. On ne considère que le bit de parité généré car les bits systématiques ne seront pas envoyés.

Symboles d'entrée État actuel	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0000	0	1	1	0	1	0	0	1	1	0	0	1	0	1	1	0
0001	1	0	0	1	0	1	1	0	0	1	1	0	1	0	0	1
0010	0	1	1	0	1	0	0	1	1	0	0	1	0	1	1	0
0011	1	0	0	1	0	1	1	0	0	1	1	0	1	0	0	1
0100	0	1	1	0	1	0	0	1	1	0	0	1	0	1	1	0
0101	1	0	0	1	0	1	1	0	0	1	1	0	1	0	0	1
0110	0	1	1	0	1	0	0	1	1	0	0	1	0	1	1	0
0111	1	0	0	1	0	1	1	0	0	1	1	0	1	0	0	1
1000	0	1	1	0	1	0	0	1	1	0	0	1	0	1	1	0
1001	1	0	0	1	0	1	1	0	0	1	1	0	1	0	0	1
1010	0	1	1	0	1	0	0	1	1	0	0	1	0	1	1	0
1011	1	0	0	1	0	1	1	0	0	1	1	0	1	0	0	1
1100	0	1	1	0	1	0	0	1	1	0	0	1	0	1	1	0
1101	1	0	0	1	0	1	1	0	0	1	1	0	1	0	0	1
1110	0	1	1	0	1	0	0	1	1	0	0	1	0	1	1	0
1111	1	0	0	1	0	1	1	0	0	1	1	0	1	0	0	1

Tableau 3.5: *Sortie du diagramme d'états du codeur RSC de rendement 4/5.*

3.3 Codeur Wyner-Ziv basé sur les codes-turbo

Dans la partie codage, on considère les trames impaires sous leur état pixel (8 bits pour 256 valeurs). Après quantification, la séquence binaire est acheminée aux deux codeurs RSC pour générer les bits de parité. Les bits systématiques sont écartés étant donné que le décodeur dispose déjà d'une information latérale pouvant remplacer leur rôle au décodage. Les différents blocs du codeur Wyner-Ziv sont détaillés ci-dessous et représentés à la figure 3.3.

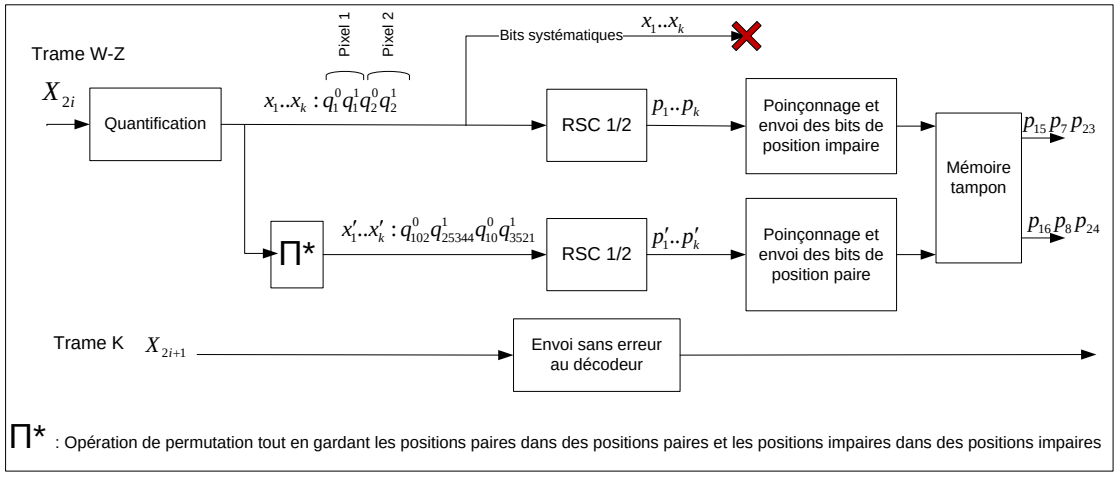

FIGURE 3.3: *Schéma bloc du codeur Wyner-Ziv pour le cas d'un turbo code avec 2 RSC 1/2.*

3.3.1 Quantification

Pour la compression des trames paires, le premier bloc de l'émetteur (voir fig. 3.3) consiste en la représentation des pixels, prenant 256 valeurs possibles, avec un nombre moindre de symboles de quantifications (2, 4 ou 16 niveaux de quantification). Cette transformation à perte permet de réaliser de la compression au prix d'une certaine distorsion. Comme la séquence des bits de quantification est déterminée par la trame à envoyer, une version bruitée de cette trame permet de donner une certaine information sur les bits quantifiés. Cette constatation va nous servir dans la partie décodage utilisant une information latérale.

3.3.2 Entrelacement

L'objectif de l'entrelaceur se trouvant à l'entrée du deuxième codeur RSC consiste à espacer et à disperser les erreurs. Les deux codeurs RSC 4/5 du code turbo prennent à leurs entrées 4 bits à la fois. Pour une quantification à 16 niveaux, ces 4 bits correspondent alors au résultat de quantification d'un pixel. Pour maintenir ce fait au second codeur après entrelacement, la permutation doit se faire au niveau symbole. C'est-à-dire qu'on permute des blocs de 4 bits regroupés ensemble et correspondant à un pixel donné. Le principe de décodage qu'on détaillera dans le chapitre suivant repose sur l'échange avec permutation des valeurs de probabilité extrinsèque entre les deux décodeurs. La sortie du premier décodeur RSC 4/5 fournira 16 valeurs extrinsèques pour chaque pixel. La permutation au niveau symbole revient alors à permuter les pixels de la trame à envoyer. Pour le cas du turbo-code avec deux codeurs RSC 1/2, l'entrelacement se fera au niveau binaire. Après quantification, les symboles sont introduis bit par bit à chacun des deux codeurs RSC 1/2. Au décodage chaque valeur extrinsèque est relative à un bit unique. Donc il suffit de réaliser une simple permutation binaire des valeurs extrinsèques avant l'échange entre les deux décodeurs.

3.3.3 Poinçonnage

Les bits quantifiés sont fournis au codeur turbo qui va écarter les bits systématiques pour ne transmettre que les bits de parité. Pour l'envoi de ces bits, on utilise un schéma de poinçonnage de période 32 dans le cas d'un code-turbo avec deux codeurs RSC 1/2 et de période 8 dans le cas d'un code-turbo avec deux codeurs RSC 4/5. Ce choix est justifié par la différence du nombre de bits à l'entrée de chaque type de turbo-code. En effet, le code-turbo avec RSC 1/2 prend en son entrée 2 bits à la fois (un pour le premier codeur RSC 1/2 et un autre pour le deuxième codeur après permutation) et génère deux bits de parité (un de chaque codeur RSC 1/2). Quant au turbo-code avec RSC 4/5, celui-ci prend en son entrée 8 bits à la fois et génère aussi deux bits de parités. Le codeur RSC 4/5 assure alors 4 fois plus de compression que le codeur RSC 1/2 donc sa matrice de poinçonnage devrait être de longueur 4 fois moins pour pouvoir comparer entre les performances des deux schémas de codage turbo.

Les bits de parité sont stockés dans une mémoire tampon et sont envoyés progressivement à la demande du décodeur. Ainsi on n'envoie que le nombre de bits de parité nécessaires pour la convergence du décodeur turbo. Pour assurer de meilleure performance de décodage, on s'assure que le bit de parité

relatif à une entrée donnée n'est pas envoyé à la fois par le premier et le second codeur RSC 1/2. Pour implémenter cette idée on fixe l'envoi des bits de parité des entrées impaires par le premier codeur et les bits de parité des entrées paires par le second codeur. La fonction d'entrelacement pseudo-aléatoire doit alors répondre à la certaine contrainte : un pixel dans une position paire est permuté dans une autre position paire choisie d'une façon aléatoire et un pixel dans une position impaire est permuté dans une autre position impaire. Cette technique ne permet pas d'assurer une meilleure compression autant qu'elle accélère la convergence du décodeur turbo résultant en un nombre moindre d'itérations.

3.4 Statistique du canal virtuel

Pour l'implémentation de l'algorithme de décodage turbo, on applique en général l'algorithme log-MAP (algorithme BCJR avec métriques de vraisemblance logarithmiques) tout en considérant que le canal est additif à bruit blanc gaussien. Dans le schéma de codage vidéo distribué proposé, le codeur n'envoie que les bits de parité nécessaires à travers le canal pour augmenter la compression. La réception des bits de parité est supposée sans erreur. Pour le calcul des valeurs de vraisemblance du canal au niveau de l'algorithme log-MAP, l'information latérale remplacera le rôle des bits systématiques qui n'ont pas été envoyés.

Avant d'implémenter le décodage turbo, il faut alors déterminer le modèle de corrélation entre la trame originale et la trame interpolée par le décodeur. Ce modèle caractérise le canal virtuel dont l'entrée est la trame originale et dont la sortie est la trame interpolée bruitée. Un exemple d'affichage d'une trame et de sa version interpolée est donné à la figure 3.4 .

FIGURE 3.4: *Une trame et sa version interpolée par moyennage.*

Pour déterminer la nature statistique de la corrélation entre la trame originale et la trame interpolée, on considère la séquence vidéo test *Foreman* format QCIF (144 $\times$ 176 pixels). On parcourt l'ensemble

des 361 trames et on compare chacune d'entre elles avec la version interpolée en fonction des deux trames qui lui sont adjacentes. Cet algorithme est décrit à la figure 3.5.

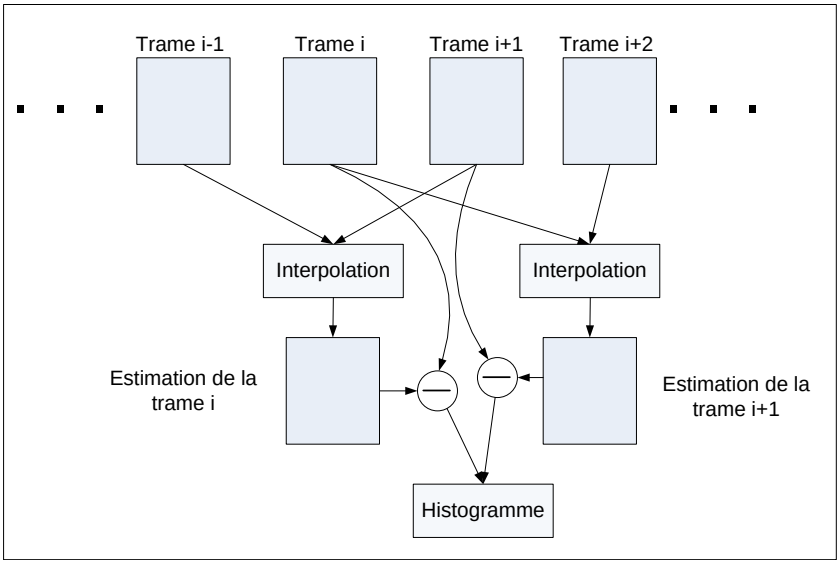

FIGURE 3.5: *Détermination de la corrélation entre la trame originale et la trame interpolée.*

On remarque à l'aide du traçage de l'histogramme de la figure 3.6 que la différence entre chaque trame et son interpolation peut être approximée par une loi de probabilité Laplacienne de paramètre $\alpha = 1/2$. Le modèle de corrélation entre la trame originale et la trame interpolée caractérise le canal

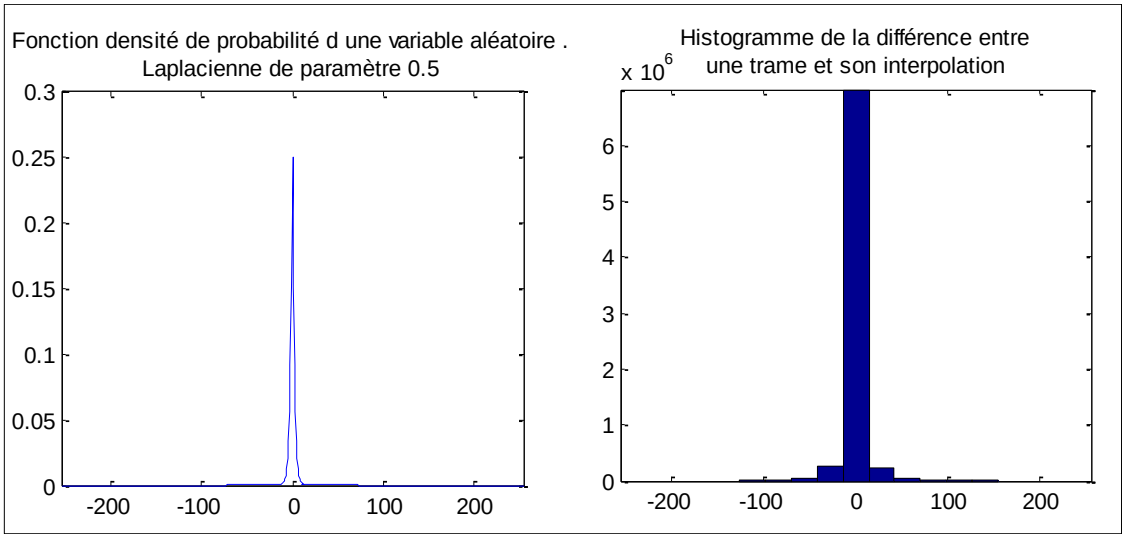

FIGURE 3.6: *Modèle de corrélation Laplacienne entre la trame originale et la trame interpolée.*

virtuel qui sera considéré pour l'implémentation de l'algorithme de décodage log-MAP.

3.5 Décodage log-MAP pour un code turbo avec 2 RSCs 1/2

Pour comprendre le fonctionnement de l'algorithme de décodage MAP pour le cas d'un codeur turbo avec 2 RSCs 1/2 on se base sur le schéma bloc donné à la figure 3.7.

Dans ce qui suit, on donne un exemple détaillé pour le calcul des métriques de branche. On commence tout d'abord, par présenter les notations qui vont être utilisées. L'entrée au codeur RSC de rendement 1/2 est dénotée par $x = [x_1, ..., x_K]$. La sortie du codeur RSC est $c = \left[\underline{c_1}, ..., \underline{c_K}\right]$ où chaque vecteur $\underline{c_k} = [x_k, p_k]$, et où p_k est le bit de parité généré par le codeur RSC au temps discret k. Le vecteur reçu au décodeur est représenté par $y = \left[\underline{y_1}, ..., \underline{y_K}\right]$, avec chaque $\underline{y_k} = [y_k^x, y_k^p]$. y^x représente la valeur reçue du bit systématique x_k et y^p représente la valeur reçue du bit de parité p_k. Finalement, l'état de l'encodeur au temps k est noté par s_k. Le décodeur BCJR MAP calcule les rapports de vraisemblance

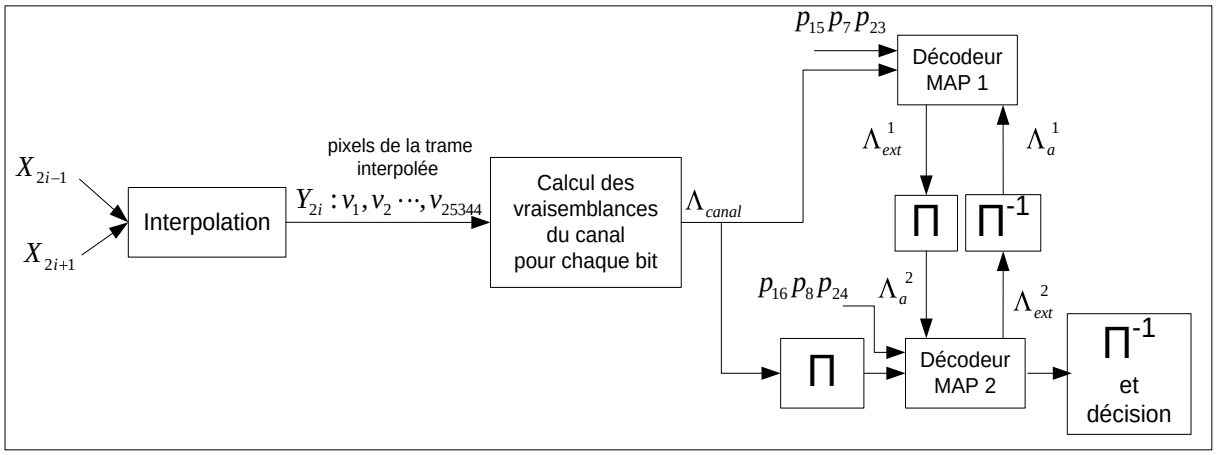

FIGURE 3.7: *Décodeur turbo avec 2 RSCs 1/2.*

pour chaque bit d'information comme suit :

$$\Lambda(x_k) = \log \frac{P(x_k = 0|y)}{P(x_k = 1|y)} \tag{3.4}$$

La probabilité à postériori $P(x_k = 0|y)$ est déterminé par le calcul des métriques de branche entre un état s' et un état s :

$$\gamma_k(s', s) = \underbrace{P(x_k)}_{\substack{\text{prob. à priori : information} \\ \text{extrinsèque de l'autre décodeur}}} \quad \underbrace{P(y_k^x|x_k)}_{\substack{\text{vraisemblance} \\ \text{du canal}}} \quad \underbrace{P\left(y_k^p\middle|p_k\right)}_{\substack{\text{vraisemblance} \\ \text{du bit de parité}}} \tag{3.5}$$

On dénote par $P(x_k)$ la probabilité à priori de l'information x_k. Après chaque itération de décodage turbo, cette valeur est mise à jour en prenant la valeur de l'information extrinsèque fournie par l'autre décodeur (voir [45]). La probabilité conditionnelle, $P\left(y_k^p\middle|p_k\right)$, représente la vraisemblance du bit de

parité. Dans le cas du système de codage vidéo Wyner-Ziv, les bits de parité sont soit poinçonnés soit envoyés au décodeur sans erreurs. On obtient alors :

$$P\left(y_k^p \middle| p_k\right) = \begin{cases} 1, y_k^p = p_k, & p_k \text{ non poinçonné} \\ 0, y_k^p \neq p_k, & p_k \text{ non poinçonné} \\ 0.5, & p_k \text{ poinçonné} \end{cases} \tag{3.6}$$

Il reste maintenant à effectuer la détermination de la probabilité de $P\left(y_k^x \middle| x_k\right)$ qui est connue sous l'appelation de la valeur de vraisemblance du canal. Notons que les bits systématiques x_k ne sont pas transmis et que l'information latérale v_l remplace les bits reçus y_k^x sans toutefois être quantifiée. Dans ce cas, on obtient l'approximation suivante :

$$P\left(y_k^x | x_k\right) \approx P\left(v_l | q_l^m\right) \tag{3.7}$$

Dans cette équation, la variable v_l désigne la valeur du pixel de l'information latérale ($v_l \in [0 \cdots 255]$). On a aussi $\bar{x} = [x_1, .., x_k, .., x_K] = \left[\underbrace{q_1^0, q_1^1}_{\text{pixel 1}}, .., \underbrace{q_l^m = x_k}_{\text{bit m du pixel } l}, .., q_K^0, q_K^1\right]$.

Pour fixer les idées, on considère l'exemple numérique décrit par la figure 3.8.

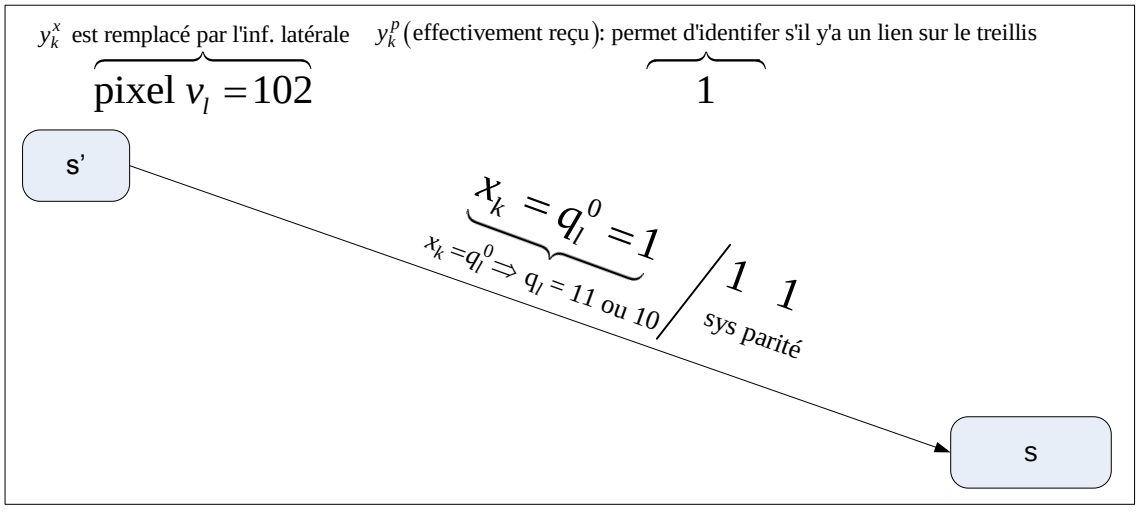

FIGURE 3.8: *Exemple numérique de calcul de la vraisemblance du canal.*

$$
\begin{aligned}
P\left(y_k^x=0\middle|\, x_k=1\right) &\approx P\left(v_l=102\middle|\, q_l^{m=0}=1\right)\\
&= \frac{P\left(q_l^0=1\middle|v_l=102\right)P(v_l=102)}{P\left(q_l^0=1\right)} \text{ (Règle de Bayes)}\\
&= \frac{\left(\sum\limits_{q_l \text{ tel que } q_l^0=1} P(q_l|v_l=102)\right)\cdot P(v_l=102)}{P\left(q_l^0=1\right)} \text{ (Marginalisation)}\\
&= \frac{(P(q_l=10|v_l=102)+P(q_l=11|v_l=102))\cdot P(v_l=102)}{P\left(q_l^0=1\right)}\\
&= \frac{(P(u_l\in[127..191]|v_l=102)+P(u_l\in[191..255]|v_l=102))\cdot\{P(v_l=102)=1/256\}}{\left\{P\left(q_l^0=1\right)=1/2\right\}}
\end{aligned}
\tag{3.8}
$$

Ici u_l désigne la valeur du pixel dans la trame originale. Finalement, on a :

$$
\begin{aligned}
&P\left(v_l=102\middle|\, q_l^{m=0}=1\right)=\\
&\tfrac{2}{256}\left(\left[F_\eta\left(191-102\right)-F_\eta\left(128-102\right)\right]+\left[F_\eta\left(255-102\right)-F_\eta\left(192-102\right)\right]\right)
\end{aligned}
\tag{3.9}
$$

avec $F_\eta(n)=\int\limits_{-\infty}^{n} p_\eta(n)\,dn$ est la fonction de distribution cumulative (fonction de répartition) de la loi Laplacienne et $p_\eta(n)=\frac{\alpha}{2}e^{-\alpha|n|}$ est la fonction de densité de distribution d'une variable aléatoire Laplacienne.

Un dernier point reste à préciser avant de pouvoir implémenter le décodeur turbo basé sur l'algorithme BCJR modifié [1]. D'après [42], on a :

$$
\Lambda_{ext}^{(1)}(x_l)=\Lambda^{(1)}(x_l)-\left[\Lambda_c r_l^{(0)}+\Pi^{-1}\left(\Lambda_{ext}^{(2)}(x_l)\right)\right] \tag{3.10}
$$

r_l désigne la valeur reçue par le canal et elle est déterminée à partir de la valeur du pixel v_l. Cette formule est applicable pour le cas où le canal est un canal additif à bruit blanc gaussien. Dans notre cas et pour déterminer l'équivalent de $\Lambda_c r_l^{(0)}$, on doit revenir à sa définition avant la supposition d'un canal AWGN. Cette valeur correspond en fait à $\ln\left[\frac{p\left(r_l^{(0)}\middle|x_l=+1\right)}{p\left(r_l^{(0)}\middle|x_l=0\right)}\right]$.
Dans le modèle étudié dans ce travail ceci revient à calculer : $\ln\left[\frac{p(v_l|x_l=+1)}{p(v_l|x_l=0)}\right]$.

Considérons la transition décrite dans la figure 3.8 entre l'état s' et l'état s du treillis du codeur RSC 1/2. Chaque transition correspond à une valeur d'entrée $x_k=0$ ou $x_k=1$. La métrique logarithmique de branche correspondant à cette transition est donnée par :

$$
\gamma_l^*(s',s)=\log\left(\gamma_l(s',s)\right) \tag{3.11}
$$

1. L'appelation modifiée vient de [42]

On en déduit les métriques directes et inverses comme suit :

$$\begin{aligned}
\alpha_{l+1}^{*}(s) &= \max_{s' \in \sigma_l}{}^{*}\left(\gamma_l^{*}(s',s) + \alpha_l^{*}(s)\right) \quad \text{avec } \alpha_0^{*}(s) = \begin{cases} 0, & \text{pour } s = S_0 \\ -\infty, & \text{pour } s \neq S_0 \end{cases} \\
\beta_l^{*}(s') &= \max_{s \in \sigma_{l+1}}{}^{*}\left(\gamma_l^{*}(s',s) + \beta_{l+1}^{*}(s)\right) \quad \text{avec } \beta_0^{*}(s) = \begin{cases} 0, & \text{pour } s = S_f \\ -\infty, & \text{pour } s \neq S_f \end{cases}
\end{aligned} \tag{3.12}$$

où σ_l désigne l'ensemble des états possibles au temps l et σ_{l+1} désigne l'ensemble des états possibles au temps $l+1$. Ici la fonction max* est défini par : $\max^{*}(x,y) = \ln\left(e^x + e^y\right)$. Mais pour la stabilité numérique lors de l'implémentation il faut éviter le calcul des exponentielles avec des arguments élevés donc on utilise plutôt cette propriété :

$$\max^{*}(x,y) = \ln\left(e^x + e^y\right) = \max(x,y) + \ln\left(1 + e^{-|x-y|}\right) \tag{3.13}$$

Après codage de la séquence binaire quantifiée, on doit rajouter des bits de terminaison afin de remettre le codeur à l'état initial S_0. Sauf que les bits de terminaison pour un codeur RSC doivent être déterminés selon l'état final S_f du codeur puis envoyés au décodeur. Cependant, dans le contexte de codage vidéo distribué, les bits systématiques ne sont pas envoyés et ils sont remplacés par les pixels de la trame interpolée. Donc au lieu de remettre l'état final à S_0 par l'ajout de bits de terminaison, il suffit tout simplement d'envoyer l'état final de l'encodeur S_f. Sinon on peut toujours initialiser les valeurs de la métrique inverse de façon équiprobable :

$$\beta_0^{*}(s) = \log(1/8),\ \forall s \tag{3.14}$$

Après le calcul des métriques logarithmiques de branches et les métriques directes et inverses, on peut déterminer le rapport de vraisemblance logarithmique de l'entrée à un instant donné :

$$\Lambda(x_l) = \max_{s' \to s=1}{}^{*}\left(\alpha_l^{*}(s) + \gamma_l^{*}(s',s) + \beta_{l+1}^{*}(s)\right) - \max_{s' \to s=0}{}^{*}\left(\alpha_l^{*}(s) + \gamma_l^{*}(s',s) + \beta_{l+1}^{*}(s)\right) \tag{3.15}$$

La fonction max* pour plusieurs arguments est donnée par :

$$\max^{*}(x,y,z) = \max^{*}(x, \max^{*}(y,z)) \tag{3.16}$$

3.6 Décodage log-MAP pour un code turbo avec 2 RSCs 4/5

Dans le cas d'un turbo code avec 2 RSCs 1/2, l'entrée au codeur est binaire : soit 0 soit 1. Le décodeur décide 0 si le rapport logarithmique de vraisemblance $\Lambda(x_l) < 0$ et décide 1 si $\Lambda(x_l) > 0$. Pour un turbo code avec 2 RSCs 4/5 l'entrée au codeur est un symbole formé de 4 bits. Le décodeur doit alors décider parmi une valeur à l'entrée allant de 0 à 15. Le décodeur turbo avec 2 RSCs 1/2 fonctionne dans le domaine des rapports de vraisemblance logarithmique (log-likelihood ratio : LLR) alors que le décodeur turbo avec 2 RSC 4/5 fonctionne dans le domaine des vraisemblances logarithmiques (log-likelihood). A chaque coup d'horloge, l'entrée au codeur turbo avec 2 RSCs 4/5 est un vecteur

de 4 bits $\underline{X}_i = (x_{4i-4}, x_{4i-3}, x_{4i-2}, x_{4i-1})$ avec $x_i \in \{0,1\}$. La valeur de vraisemblance pour une entrée $\underline{X}_k$ est donnée par :

$$\Lambda'(x_k = q) = \log P(\underline{X}_k = q|y) \quad avec\, q = 0, \ldots, 15. \tag{3.17}$$

Le décodeur décide par exemple, $\hat{\underline{X}}_k = 5$ si $\Lambda'(x_k = 5) > \Lambda'(x_k = q), \forall q \neq 5$. L'expression de la métrique de branche est donnée par :

$$\gamma_k(s', s) = \underbrace{P(\underline{X}_k = q)}_{\substack{\text{prob. à priori : information} \\ \text{extrinsèque de l'autre décodeur}}} \underbrace{P(\underline{Y}_k^x | \underline{X}_k = q)}_{\substack{\text{vraisemblance} \\ \text{du canal}}} \underbrace{P(y_k^p | p_k)}_{\substack{\text{vraisemblance} \\ \text{du bit de parité}}} \tag{3.18}$$

La valeur de la vraisemblance du bit de parité est donnée de façon similaire que précédemment :

$$P(y_k^p | p_k) = \begin{cases} 1, y_k^p = p_k, & p_k \text{ non poinçonné} \\ 0, y_k^p \neq p_k, & p_k \text{ non poinçonné} \\ \frac{1}{16}, & p_k \text{ poinçonné} \end{cases} \tag{3.19}$$

La valeur de vraisemblance canal est donnée par $P(\underline{Y}_k^x | \underline{X}_k = q) = P(v_k | \underline{X}_k = q)$, avec v_k est le pixel information latérale. Un exemple de calcul de la vraisemblance canal est donné ci-dessus.

$$\begin{aligned} & P(\underline{Y}_k^x | \underline{X}_k = q) = P(v_k = 102 | \underline{X}_k = 5) \\ & = \frac{P(\underline{X}_k = 5 | v_k = 102) P(v_k = 102)}{P(\underline{X}_k = 5)} \text{(Règle de Bayes)} \\ & = \frac{[F_\eta(80-102) - F_\eta(95-102)] P(v_k = 102)}{P(\underline{X}_k = 5)}, \\ & \text{avec :} \quad F_\eta = \int_{-\infty}^{\eta} p_\eta(\eta)\, d\eta = \int_{-\infty}^{\eta} \frac{\alpha}{2} e^{-\alpha|\eta|} d\eta. \cdot \end{aligned} \tag{3.20}$$

La valeur extrinsèque est calculée comme suit :

$$\Lambda'_{ext}(\underline{X}_k = q) = \Lambda'(\underline{X}_k = q) - \log(P(\underline{Y}_k^x | \underline{X}_k = q)) - \log(P(\underline{X}_k = q)) \tag{3.21}$$

Lorsque le nombre de niveaux de quantification est 16, chaque pixel est quantifié sur 4 bits qui sont fournis au codeur RSC 4/5. Et comme la permutation est effectuée au niveau symbole, le deuxième codeur RSC 4/5 reçoit à chaque coup d'horloge 4 bits correspondant à un pixel donné. Le décodeur log-MAP génère des valeurs de vraisemblance extrinsèques relatives à chaque symbole formé des 4 bits de quantification d'un pixel donné. Lors de l'échange extrinsèque, il suffit d'effectuer la même permutation que celle effectuée pour les symboles avant d'être passés au deuxième codeur.

Par contre l'échange extrinsèque ne s'effectue pas de la même façon lorsqu'il s'agit d'un quantificateur à 4 niveaux où chaque pixel est quantifié en deux bits. L'entrée du premier codeur RSC 4/5

correspond alors aux bits de quantification de deux pixels à la fois. La permutation est effectuée en gardant regroupés les 4 bits de quantification de deux pixels consécutifs. Ce choix de permutation est dû au fait que les valeurs extrinsèques générées par le décodeur log-MAP correspondent à un symbole formé par 4 bits de quantification de deux pixels adjacents. Ainsi le transfert des valeurs extrinsèques se fait par la même permutation effectuée au codage.

Pour expliquer davantage les différences du codeur/décodeur Wyner-Ziv lorsque le nombre de niveaux de quantifications varie, on détaille dans les sous-sections suivantes les opérations de codage et décodage turbo pour une quantification à 16, 4 et 2 niveaux.

3.6.1 Quantification à 16 niveaux

Chaque pixel représenté par 8 bits est quantifié avec seulement ses 4 bits les plus significatifs. Ces 4 bits sont fournis en un coup d'horloge vers le premier codeur RSC 4/5 et après permutation au niveau symbole vers le deuxième codeur RSC 4/5. Les deux codeurs RSC 4/5 génèrent chacun un bit de parité pour chaque pixel, soit 4 bits de quantification. Ces bits de parité sont poinçonnés pour adapter le débit à la variation des statistiques du canal virtuel. La partie de codage avec 16 niveaux de quantification tel que décrite, est schématisée dans la figure 3.9.

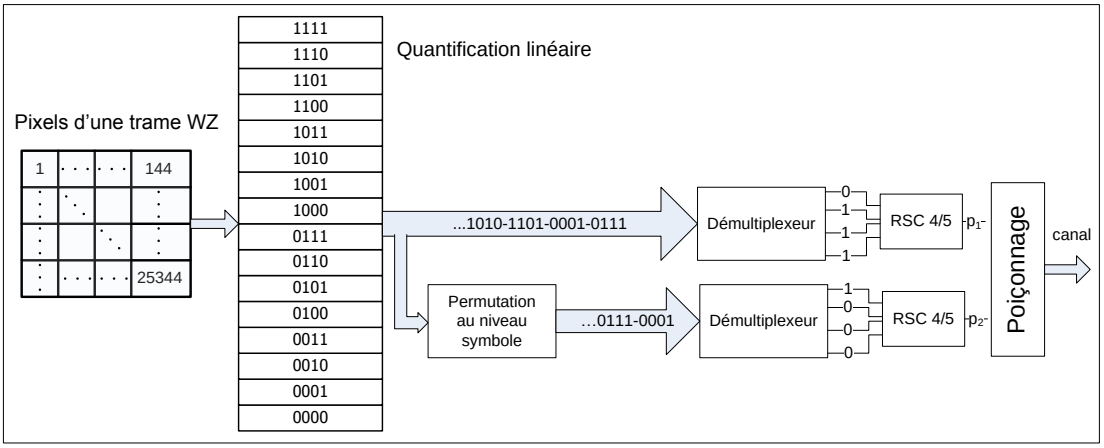

FIGURE 3.9: *Codeur turbo avec 2 RSCs 4/5 pour une quantification à 16 niveaux.*

Le calcul des valeurs de vraisemblance canal se base sur les pixels de la trame interpolée en remplaçant les bits systématiques pour une économie de débit. Les deux décodeurs log-MAP de la figure 3.10, séparés par un entrelaceur au niveau symbole calculent 16 valeurs de vraisemblances relatives aux 16 valeurs possibles des symboles à 4 bits. La permutation au niveau symbole permet de maintenir ensemble les 4 bits de quantification d'un pixel donné. Le calcul des vraisemblances canal s'effectue alors sans marginalisation (voir l'équation 3.8) contrairement au cas d'un codeur turbo avec 2 RSCs 1/2 (voir l'équation 3.8). Les bits décodés q' sont ensuite utilisés pour raffiner la trame interpolée qui a déjà servie dans le décodage turbo. On obtient finalement une trame reconstruite à partir de la trame

interpolée où les erreurs d'interpolation ont été inhibées par l'utilisation des bits décodés.

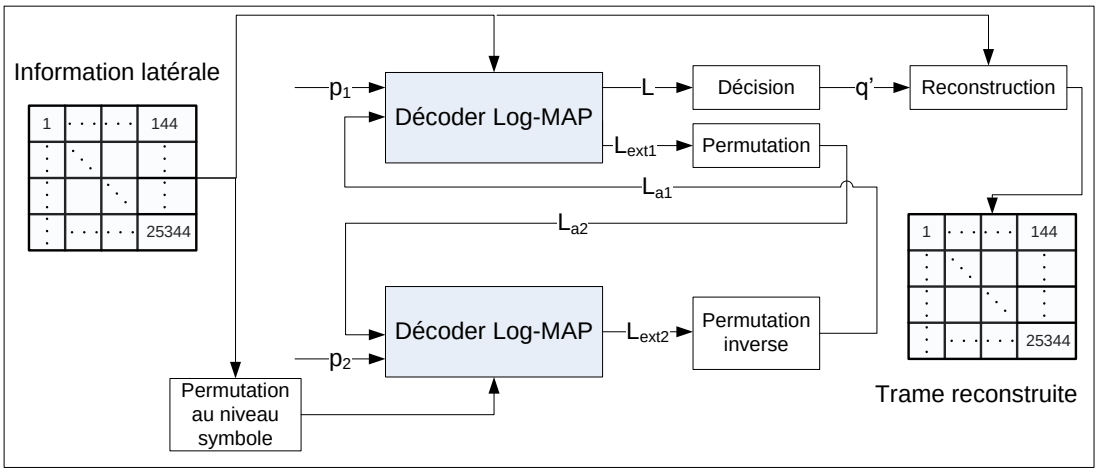

FIGURE 3.10: *Décodage turbo à 2 RSCs 4/5 et reconstruction pour 16 niveaux de quantification.*

L'entrelacement au niveau symbole contrecarre le problème de marginalisation mais ne permet pas d'obtenir un grand niveau de décorrélation de l'information extrinsèque échangée entre les deux décodeurs log-MAP. En effet, le but de l'entrelaceur interposé entre les deux décodeurs log-MAP est de disperser l'information extrinsèque générée par chacun des décodeurs avant de l'échanger. L'entrelaceur au niveau symbole est limité par la contrainte de garder ensemble les 4 bits de quantification de chaque pixel lors de l'échange extrinsèque présenté dans la figure 3.11.

3.6.2 Quantification à 4 niveaux

Pour ce type de quantification, chaque pixel est représenté par 2 bits et le flux binaire est fourni à l'encodeur 4/5. Cet encodeur regroupe les 4 bits de quantification de 2 pixels consécutifs à chaque coup d'horloge pour générer le bit de parité correspondant. L'entrelaceur précédant le deuxième encodeur doit alors veiller à garder ensemble chacun des 4 bits des deux pixels consécutifs comme le montre la figure 3.12. Le fait de garder ensemble deux pixels adjacents après la permutation risque de dégrader les performances au décodage. D'un côté les erreurs ne sont pas dispersées d'une façon optimale si on garde chaque deux pixels adjacents ensemble, et d'un autre côté la décorrélation de l'information extrinsèque sera affectée. L'opération de décodage telle décrite est schématisée dans la figure 3.10.

3.6.3 Quantification à 2 niveaux

La sous optimalité de dispersion des erreurs décrite ci-dessus est encore plus accentuée pour le cas d'un quantificateur à 2 niveaux. En effet ce quantificateur associe à chaque pixel un seul bit (correspondant à son bit le plus signficatif). Par la suite, les 4 bits sont regroupés et présentés à l'encodeur 4/5. L'entrelacement gardera regroupés les 4 bits de chacun des 4 pixels adjacents comme le montre la figure 3.13. Pour remédier à ce regroupement des pixels adjacents lors de l'entrelacement et pour

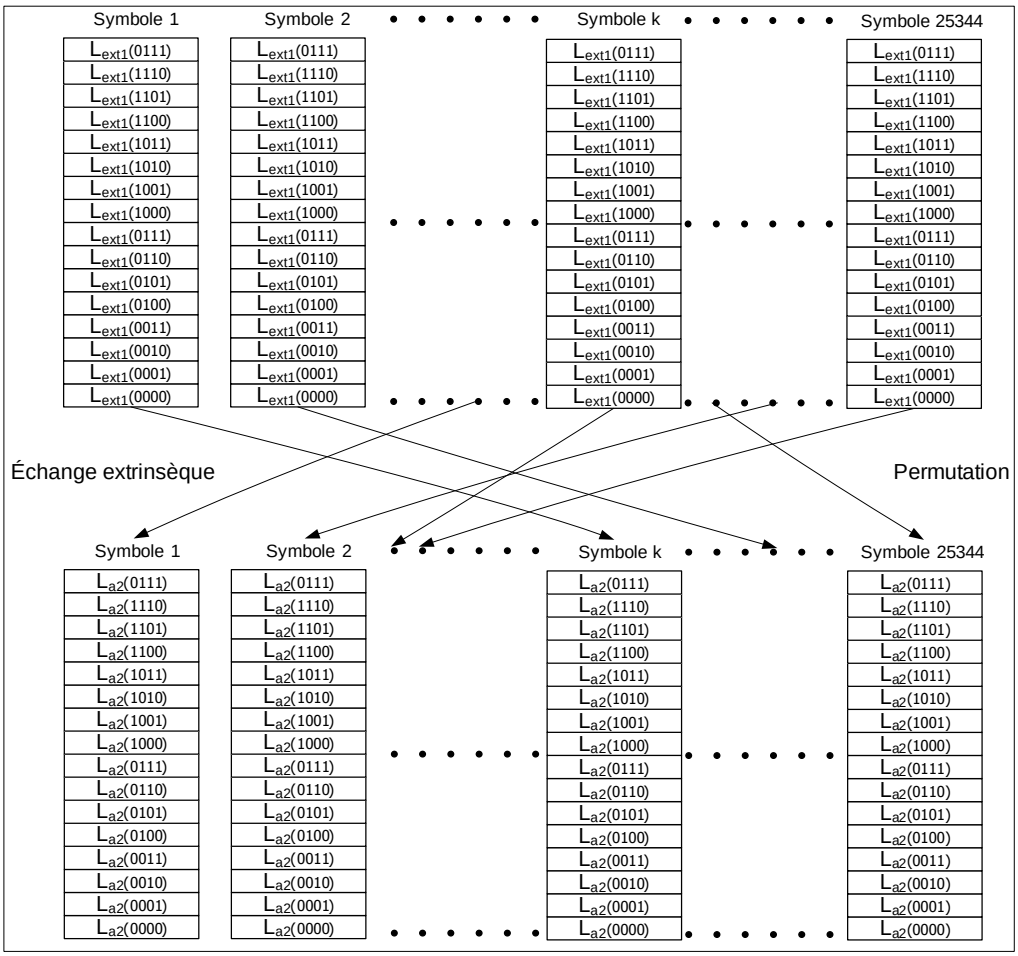

FIGURE 3.11: *Échange extrinsèque entre les deux décodeurs log-MAP pour une permutation au niveau symbole.*

assurer une meilleure dispersion d'erreur, on peut mélanger les pixels de la trame originale avant la quantification et le codage.

3.7 Implémentation et simulation

Après avoir présenté les divers composants du schéma Wyner-Ziv de codage et décodage vidéo basés sur les turbos codes, on s'intéresse dans cette section à leur implémentation et simulation. Deux schémas de code turbo ont été étudiés, le premier avec deux codeurs RSC 1/2 et le second avec deux codeurs RSC 4/5, le premier turbo code étant moins complexe que le deuxième. Cependant les performances du second turbo code dépassent celles du premier. Pour pouvoir effectuer une comparaison équitable, nous reproduisons dans nos simulations les mêmes conditions considérées par Aaron, Zhang et Girod [22] avec un code turbo à 2 RSC 4/5. Le premier objectif des simulations est de trouver des résultats similaires avec ceux présentes dans [22] ce qui permet de valider la phase implémentation.

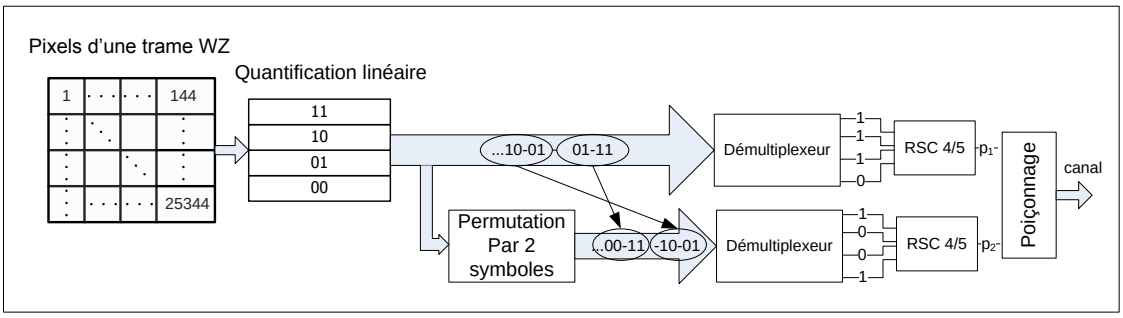

FIGURE 3.12: *Codeur turbo avec 2 RSCs 4/5 pour une quantification à 4 niveaux.*

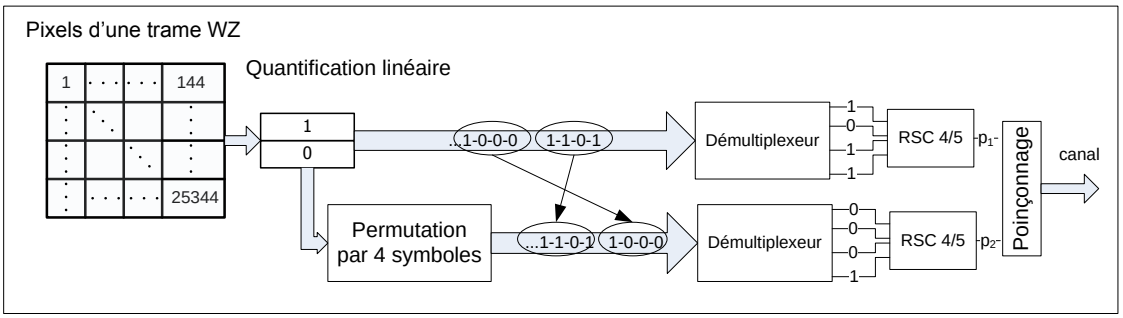

FIGURE 3.13: *Codeur turbo avec 2 RSCs 4/5 pour une quantification à 2 niveaux.*

Par la suite, on compare les performances de compression d'un turbo code avec 2 codeurs RSC 4/5 et 2 codeurs RSC 1/2.

Dans [22], on considère les 101 premières trames de la séquences vidéo `Carphone`, et les 361 premières trames de la séquence vidéo `Foreman`. On envoie progressivement les bits de parité jusqu'à atteindre une probabilité d'erreur symbole inférieure à 10^{-3}. Par la suite, on effectue la reconstruction de la trame envoyée à l'aide de la trame interpolée et des bits de parité décodés. On détermine le rapport signal sur bruit de pointe (PSNR) de la trame reçue pour différents niveaux de quantification.

3.7.1 Implémentation du turbo code

Le décodage turbo repose sur l'échange des valeurs de vraisemblance extrinsèques entre deux décodeurs log-MAP, et ce, un certain nombre de fois avec la condition d'arrêt l'atteinte d'un nombre maximal d'itérations présélectionné ou l'atteinte d'une probabilité d'erreur symbole suffisamment faible (voir figure 3.14). Si le décodeur ne parvient pas à atteindre une probabilité d'erreur faible au bout du nombre maximal d'itérations, on envoie un bit de parité supplémentaire pour chacun des codeurs. Ainsi, le débit s'adapte au changement des caractéristiques du canal virtuel. On fait varier dans les simulations, le nombre d'itérations maximal selon la probabilité d'erreur par symbole atteinte : si on constate que cette probabilité est élevée, on diminue le nombre d'itérations maximal et, dans le

cas contraire, on l'augmente dans le but d'atteindre la probabilité d'erreur seuil sans demande de bits de parité additionnels. L'estimation du taux d'erreurs symbole (SER) est effectuée par accès direct à l'information originale. Dans le chapitre 7, le SER sera estimé à partir des rapports de vraisemblance et le décodage sera validé par un code de redondance cyclique (CRC).

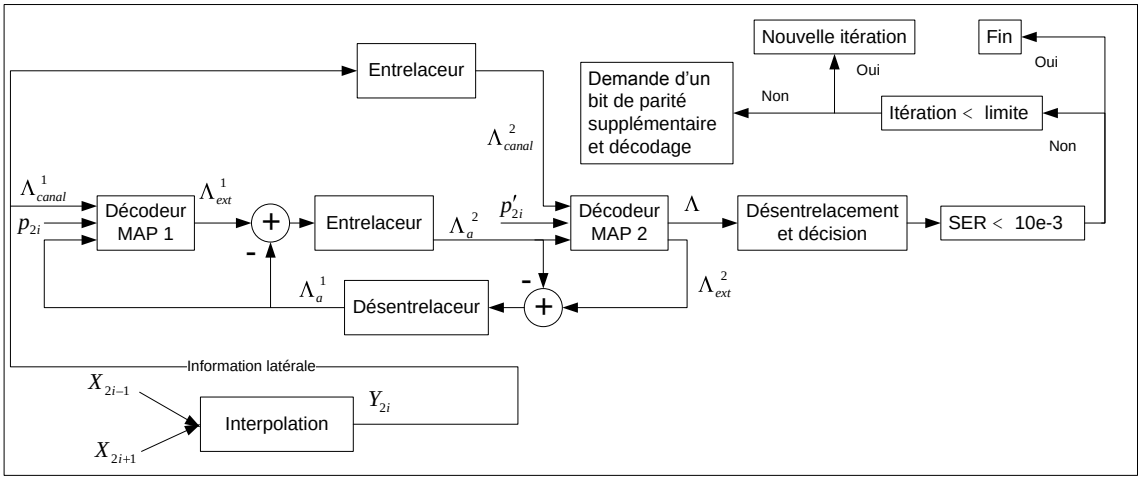

FIGURE 3.14: *Schéma bloc du décodeur turbo.*

3.7.2 Résultats des simulations

Pour valider les codeur et décodeur turbo, on a reproduit les mêmes conditions de simulation que dans [22]. On considère les 101 premières trames de la séquence `Carphone`. On envoie les 50 trames paires et on considère que les 51 trames impaires (trames intra) sont disponibles au décodeur. Après le processus de quantification des trames paires (trames inter), on regroupe les bits en blocs de longueur $L = 25344 * Q$ bits, Q étant le nombre de bits de quantification, et on les fournit au codeur turbo qui va envoyer uniquement les bits de parité provenant des deux codeurs RSC 1/2 ou RSC 4/5. Pour réduire les délais de simulation du décodage, on observe les statistiques de l'information latérale pour fixer un nombre de bits de parité à envoyer initialement. Si l'information latérale est fortement bruitée, on fixe un nombre initial important car on sait que le décodeur va avoir besoin d'un nombre élevé de bits de parité pour pouvoir décoder correctement la séquence de bits.

La figure 3.15 présente le résultat des différentes simulations effectuées dans le but d'améliorer les performances du codeur Wyner-Ziv. La performance est évaluée par la courbe du PSNR (en dB) de la séquence vidéo reconstruite en fonction du débit de transmission en kbps. Les différents points sur les courbes correspondent aux différents niveaux de quantification : i.e., 0, 2, 4, 16 niveaux. Lorsque l'on augmente le nombre de niveaux de quantification, le débit augmente et la distorsion diminue.

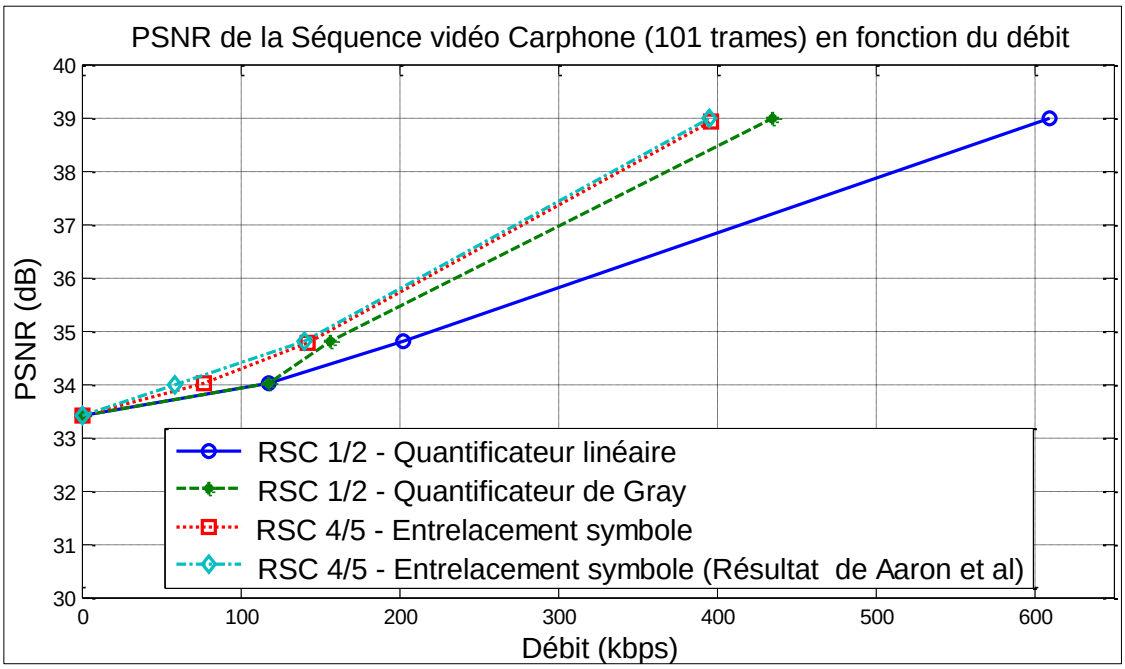

FIGURE 3.15: *Comparaison entre les performances de différents schémas de codage vidéo distribué.*

3.8 Discussion et interprétation des résultats de simulations

Chaque point des courbes tracées dans la figure 3.15 constitue un schéma de codage et de décodage Wyner-Ziv avec des paramètres particuliers. Ces paramètres sont le nombre de niveaux de quantification, le type d'entrelacement et la nature du code turbo et du quantificateur.

3.8.1 Code turbo avec 2 codeurs RSC 1/2

Un codeur RSC 1/2 prend un seul bit à la fois. Chaque transition dans le treillis correspond alors à un seul bit. Le calcul du rapport de vraisemblance canal considère la valeur du pixel de la trame interpolée qui contient le bit en question à une position donnée. Ce pixel est le centre de la Laplacienne et repose sur le principe de marginalisation sur tous les symboles contenant à cette position la valeur du bit. Ce principe est explicité à la figure 3.16 dans le cas d'un quantificateur à 16 niveaux. Le pixel de la trame interpolée correspond à une valeur de 0 à 255 et qui vaut, dans cet exemple, 102. Le bit à décoder occupe la deuxième position.

Le décodeur log-MAP décode ainsi les bits un à un et génère des valeurs de vraisemblance et les valeurs extrinsèques pour un seul bit à la fois, d'où la possibilité de bénéficier des avantages de la permutation au niveau binaire qui vise à la décorrélation de l'échange extrinsèque entre les deux décodeurs log-MAP.

L'utilisation du code de Gray à la place du code linéaire augmente considérablement les performances

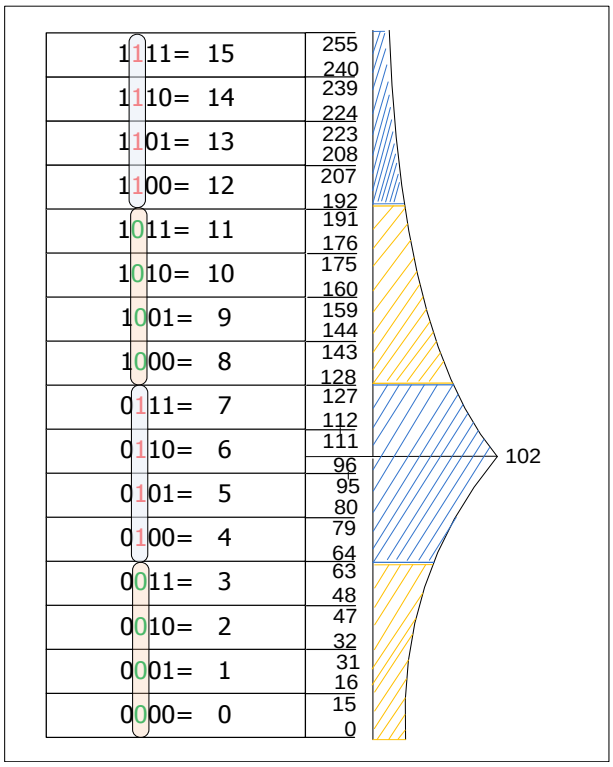

FIGURE 3.16: *Principe de marginalisation pour le calcul de la vraisemblance canal pour un décodeur turbo à 2 RSC 1/2.*

du système. En effet, pour le cas d'une quantification à 16 niveaux, si le pixel interpolé sort de la plage de quantification du pixel original, on affecte au décodage plus qu'un bit si on utilise une quantification linéaire. Pour code de Gray, un seul bit change entre deux symboles adjacents. Donc dans le cas de décalage d'un seul niveau du pixel interpolé, un seul pixel sera affecté et le décodeur log-MAP pourra plus aisément le corriger.

3.8.2 Code turbo avec 2 codeurs RSC 4/5

A chaque coup d'horloge, le codeur récursif systématique de rendement 4/5 prend 4 bits, au lieu d'un seul bit (cas du codeur de rendement 1/2), et leur associe un bit de parité. Un tel turbo code assure une meilleure compression que le turbo code précédent mais au prix d'une complexité plus élevée au décodage. L'amélioration des performances d'un tel système est, entre autres, due au fait qu'il n'y a plus de dégradation causée par la marginalisation lors du calcul des valeurs de vraisemblances comme on le voit dans la figure 3.17.

Cependant l'entrelacement est au niveau symbole et ne peut être effectué au niveau binaire sans faire face au problème de marginalisation lors de l'échange extrinsèque. À noter aussi que si l'entrelacement se fait au niveau binaire, le calcul des valeurs de vraisemblance canal se fait sans marginalisation au premier décodeur log-MAP alors qu'au niveau du second décodeur, qui est précédé d'une permu-

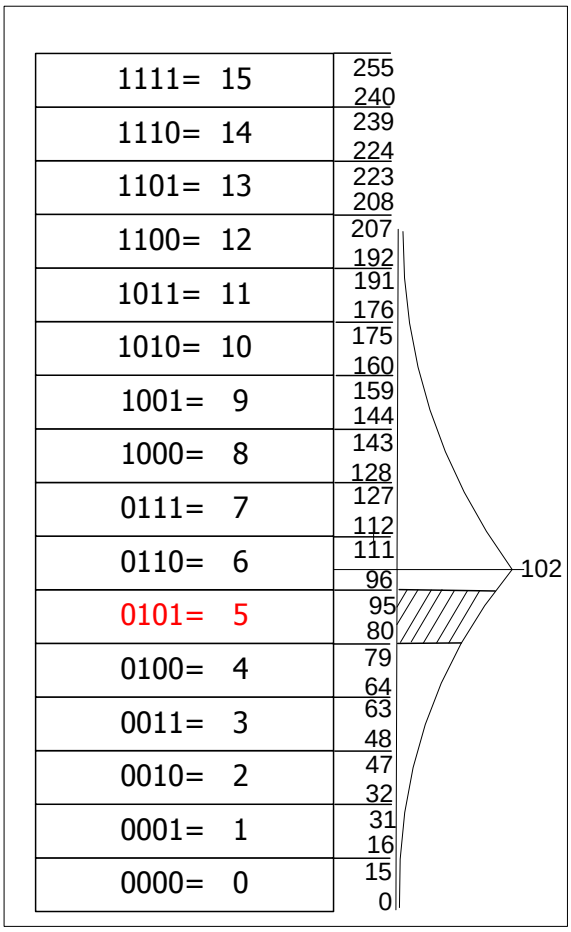

FIGURE 3.17: *Calcul de la vraisemblance canal par un décodeur turbo à 2 RSC 4/5 évitant la marginalisation.*

tation au niveau binaire, la marginalisation est obligatoire pour le calcul de la vraisemblance canal.

Les essais effectués pour l'implémentation de l'entrelacement au niveau binaire avec un turbo code à 2 codeurs RSC 4/5 nous a permis de conclure que la perte générée par la marginalisation quant à l'échange extrinsèque et par la marginalisation au niveau du deuxième décodeur surpasse le gain dû à l'échange d'une information plus décorrélée entre les deux décodeurs log-MAP. Il n'est donc pas avantageux d'effectuer un entrelacement binaire pour les codes RSC 4/5.

3.8.3 Décodage et reconstruction

La trame interpolée au niveau du décodeur est utilisée en premier lieu pour décoder les symboles de quantification des pixels de la trame originale. Dans l'opération de décodage turbo, cette trame remplace les bits systématiques écartés pour des raisons de compression. À la sortie du décodeur turbo, on obtient les symboles de quantification de chacun des pixels de la trame originale. À rappeler que la quantification est une opération de compression rajoutant de la distorsion. La trame interpolée va être

FIGURE 3.18: *Reconstruction pour une quantification à 2,4 et à 16 niveaux de la trame 92 de la séquence vidéo* `Carphone`.

à nouveau utilisée pour contrecarrer la distorsion ajoutée après quantification. En effet elle constitue une version bruitée de la trame originale et les symboles décodés peuvent être utilisés pour son raffinement. Dans les simulations effectuées on a considéré un quantificateur à 2, 4 et 16 niveaux offrant un niveau de distorsion décroissant. Les résultats de la reconstruction pour ces divers niveaux de quantification sont présentés à la figure 3.18. On remarque à partir de cette figure que le déplacement de la ligne marquée par des flèches n'a pu être détecté qu'avec 16 niveaux de quantification. On remarque aussi l'amélioration de la qualité de la trame reconstruite lorsqu'on augmente le nombre de niveau de quantifications.

3.9 Conclusion

Dans ce chapitre on a présenté et détaillé le bloc codeur/décodeur Slepian-Wolf (SW) de l'architecture du système de codage vidéo distribué. Ce bloc est basé sur les codes turbo qui sont connus par des performances tendant vers les limites théoriques d'un code canal. Les turbo codes quant à eux se basent sur l'algorithme log-MAP (BCJR) qu'on a aussi présenté et adapté à l'architecture vidéo distribué. Le codec SW constitue le noyau du codec Wyner-Ziv qui désigne la partie du codec vidéo distribué permettant de coder les trames WZ. On rappelle que dans le système de codage vidéo distribué présenté, les trames clés K sont envoyées en utilisant les standards conventionnels de compression vidéo (i.e.

H.263 ou H.264).

On a aussi présenté, dans ce chapitre, les différentes simulations implémentées pour étudier les performances du codeur/décodeur Wyner-Ziv. Le choix de codeurs RSC de rendement 1/2 était fait dans le but de faciliter la partie décodage. Même si la détermination des valeurs de vraisemblance se fait par marginalisation, on arrive tout de même à avoir de bonnes performances avec un entrelacement au niveau binaire. Avec les codeurs RSC 4/5, on atteint de meilleures performances car ils permettent de regrouper les 4 bits d'un symbole, évitant ainsi la marginalisation lors du décodage.

Les simulations effectuées dans ce chapitre sont principalement une reprise de certains résultats figurant déjà dans la littérature. Mais le développement et l'implémentation d'un schéma bloc de codage vidéo distribué se basant sur la technique de codage canal turbo constitue un élément indispensable pour une compréhension minutieuse du fonctionnement du codec Wyner-Ziv. Ceci permet, en outre, d'apporter des améliorations par la suite. En effet, l'objet du chapitre suivant se rapporte à munir le système DVC décrit précédemment d'un dispositif permettant d'assurer un schéma de poinçonnage adaptatif. Ainsi un nombre supérieur de bits de parité est envoyé lorsque l'interpolation (le canal virtuel) éprouve des difficultés, alors qu'un nombre réduit de bits de parité est envoyé lorsque l'interpolation donne de bons résultats.

Chapitre 4

Amélioration des performances par poinçonnage adaptatif

4.1 Introduction

Dans les chapitres précédents, le schéma de codage vidéo distribué dans le domaine pixel a été étudié et implémenté en se basant sur l'un des premiers travaux de cet axe de recherche [22]. Les résultats de simulation obtenus sont semblables à ceux présentés dans [22], validant la phase d'implémentation d'un système de codage vidéo distribué utilisant les turbo-codes. On étend, dans ce qui suit, ces travaux en présentant une nouvelle technique de poinçonnage adaptative permettant d'envoyer plus de bits de parité lorsque le canal virtuel est bruité et moins de bits de parité dans le cas contraire. On rapporte une amélioration du niveau de compression en comparaison avec les techniques usuelles de poinçonnage. Ce chapitre est organisé comme suit. On présente, en premier lieu, les techniques de poinçonnage utilisées dans la littérature et on explique l'impact des bits de parité dans la correction des erreurs. Par la suite, on présente un nouveau schéma de codage vidéo distribué permettant d'effectuer un poinçonnage adaptatif. Finalement, on implémente ce schéma et on présente les résultats de simulation.

4.2 Dispositif de poinçonnage conventionnel

On reprend dans la figure 4.1, le schéma de codage vidéo distribué utilisé dans [22]. Les trames impaires sont considérées parfaitement reconstituées au niveau du décodeur et vont servir à la génération de l'information latérale pour décoder les trames paires. Ces dernières sont introduites à un quantificateur scalaire uniforme à 16 niveaux et par la suite, à l'encodeur turbo composé de deux RSCs de rendement 4/5 présenté dans le chapitre 3 et dont une portion du treillis est donné dans la figure 4.2. Ce treillis est composé de 16 états et comporte des transitions doubles avec deux bits de parité différents (une transition génère 0 comme bit de parité et l'autre transition génère 1 comme bit de parité).

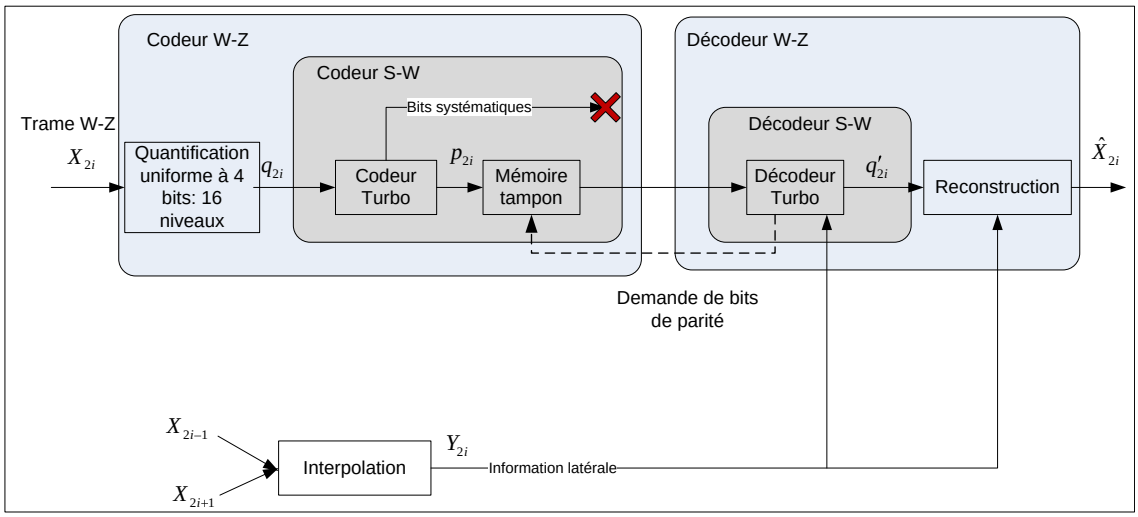

FIGURE 4.1: *Schéma bloc de la compression vidéo distribuée.*

Le processus de décodage turbo, détaillé dans le chapitre 3, se base sur le calcul des métriques de branche $\alpha_k(s)$, $\beta_k(s)$ et $\gamma_k(s',s)$ pour chacune des transitions entre les états s' et s. On rappelle ici l'expression de la métrique de branche :

$$\gamma_k(s',s) = \underbrace{P(\underline{X}_k = q)}_{\substack{\text{prob. à priori : information} \\ \text{extrinsèque de l'autre décodeur}}} \underbrace{P(\underline{Y}_k^x | \underline{X}_k = q)}_{\substack{\text{vraisemblance} \\ \text{du canal}}} \underbrace{P(y_k^p | p_k)}_{\substack{\text{vraisemblance} \\ \text{du bit de parité}}} \tag{4.1}$$

On s'intéresse dans ce chapitre au rôle des bits de parité lors du décodage turbo dont la valeur de vraisemblance est donnée par :

$$P(y_k^p | p_k) = \begin{cases} 1, y_k^p = p_k, & p_k \text{ non poinçonné} \\ 0, y_k^p \neq p_k, & p_k \text{ non poinçonné} \\ \frac{1}{16}, & p_k \text{ poinçonné} \end{cases} \tag{4.2}$$

En observant le treillis donné dans la figure 4.2, on remarque que la réception d'un bit de parité réduit de moitié le nombre potentiel de symboles d'entrée. Ce fait est plus utile lorsque le bit systématique est erroné que lorsque qu'il est correct. Dans le contexte de codage vidéo distribué, ceci revient à dire qu'un bit de parité est plus utile lorsque la génération de l'information latérale échoue que lorsqu'elle est en bon état.

Le schéma de poinçonnage employé dans [22], possède un patron périodique de 8 bits de parité et fonctionne comme décrit dans la figure 4.3 . À chaque période de 8 pixels un certain nombre de bits de parité, dénoté n_{p_1} pour le premier décodeur et n_{p_2} pour le second décodeur, sont envoyés. Le reste des bits de parité est poinçonné. Ainsi le nombre total des bits de parité est égal à :

$$nb_{total} = (n_{p_1} + n_{p_2}) \times \frac{25344}{8} \tag{4.3}$$

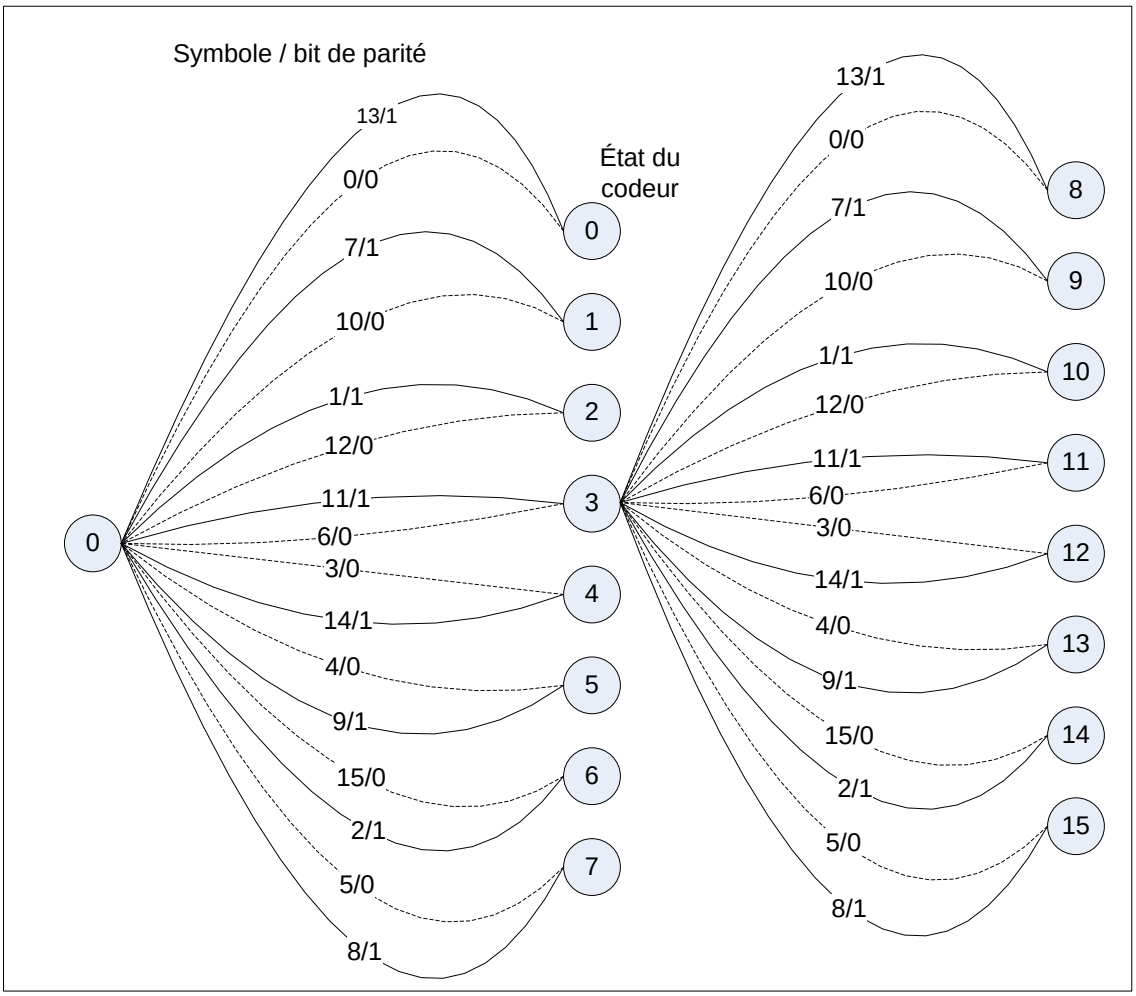

FIGURE 4.2: *Portion du treillis d'un codeur RSC de rendement 4/5.*

où 25344 désigne le nombre total des pixels dans la trame. Si le décodeur ne parvient pas à converger avec ce nombre de bits de parité, une demande est envoyée à l'encodeur qui va transmettre un nombre de bits de parité additionnels correspondant à un bit de parité supplémentaire par période de 8 symboles. Le nombre de bits de parité supplémentaires correspond donc à :

$$nb_{added} = \frac{25344}{8} = 3168 \text{ bits} \tag{4.4}$$

Ce système ne permet pas d'avoir une granularité fine de contrôle de débit. Pour cette raison, il est plus approprié de comparer les performances d'un schéma de poinçonnage adaptatif avec un dispositif de poinçonnage permettant d'assurer une granularité plus fine comme, par exemple, l'utilisation d'un patron de poinçonnage aléatoire sur l'ensemble de toute la trame. Ainsi le nombre de bits de parité envoyé sera mieux ajusté.

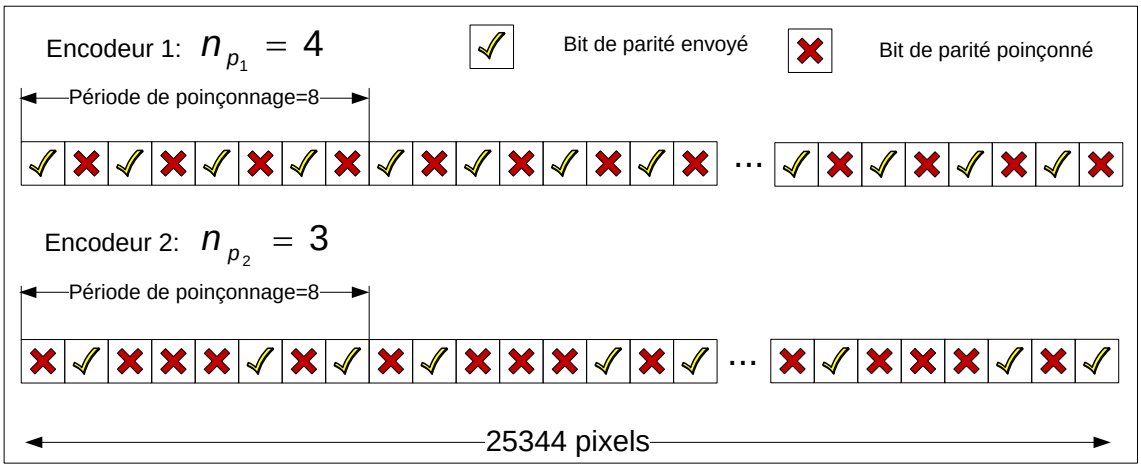

FIGURE 4.3: *Schéma de poinçonnage de période 8.*

4.3 Dispositif de poinçonnage adaptatif

4.3.1 Description de l'architecture du codeur Wyner-Ziv avec poinçonnage adaptatif

Le comportement d'un canal de propagation habituel est généralement aléatoire et imprévisible. Le poinçonnage ne peut donc être ajusté en fonction de ses variations. Cependant dans le contexte de codage vidéo distribué, le comportement du canal peut être estimé au niveau du décodeur. En effet, en observant les trames clés voisines, le décodeur peut évaluer approximativement les régions de l'information latérale où l'interpolation est plus susceptible d'échouer. Ceci correspond aux emplacements où la différence entre la valeur du pixel dans la trame clé précédente et sa valeur dans la trame clé suivante est élevée. Ainsi, le décodeur peut avoir une certaine connaissance de la variation du bruit affectant le canal virtuel. Néanmoins, cette information se rapportant au comportement du canal virtuel, doit être transmise à l'encodeur pour qu'il puisse procéder à un poinçonnage adaptatif, en évitant de poinçonner les bits de parité relatifs aux pixels les plus bruités.

On propose une nouvelle architecture de codage vidéo distribué, présentée à la figure 4.4, permettant la minimisation de la sollicitation du canal de retour, par lequel le décodeur informe l'encodeur à propos de l'estimation des pixels bruités.

En réalité, l'échec de l'interpolation se trouve à être généralement regroupé dans certains blocs de l'image interpolée comme le montre la figure 4.5. Ainsi, l'idée consiste à découper la trame en un certain nombre de régions rectangulaires, à estimer la valeur du bruit pour chacune de ces régions et finalement retransmettre le nombre de bits de parité nécessaire pour chaque région. Dans ce travail, on effectue un découpage en 44 régions, soit 4 rangées par 11 colonnes (voir 4.5). Chaque région comporte alors $36 \times 16 = 576$ pixels. Pour chaque région, le décodeur détermine le nombre de bits de parité nécessaires par période de 8 bits de parité. Par exemple, la région 1 de la figure 4.5 ne contient qu'un petit nombre d'erreurs, le nombre de bits de parité par période de 8 bits de parité est alors fixé

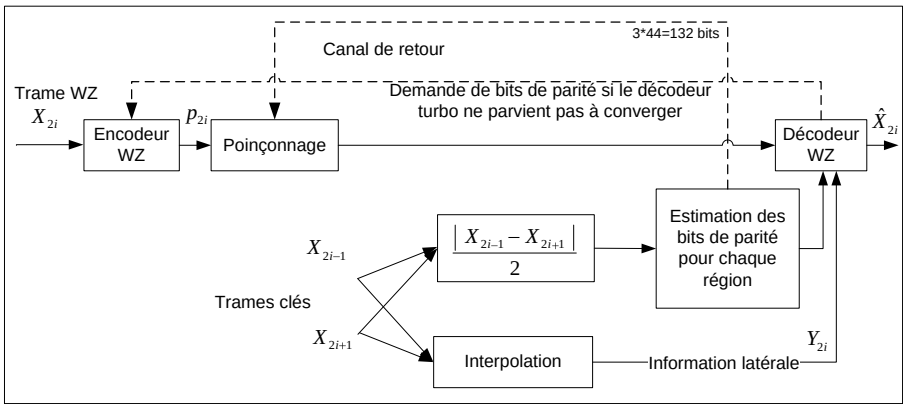

FIGURE 4.4: *Codec Wyner-Ziv avec poinçonnage adaptatif.*

à uniquement 2 (voir tableau 4.1). C'est-à-dire que pour chaque période de 8 bits de parité, seulement 2 bits de parité sont envoyés ce qui revient à un nombre total de $\frac{2}{8} \times 576 = 144$ bits de parité pour la première région. Le reste des 432 bits de parité est poinçonné. Cependant les régions 9, 20, 21, contiennent des blocs d'erreurs dues certainement à un mouvement dans la trame. Le nombre de bits de parité alloué à ces régions est de 8 d'après le tableau 4.1. Ce qui revient à dire que tous les bits de parité relatifs à ces régions sont envoyés sans aucun poinçonnage.

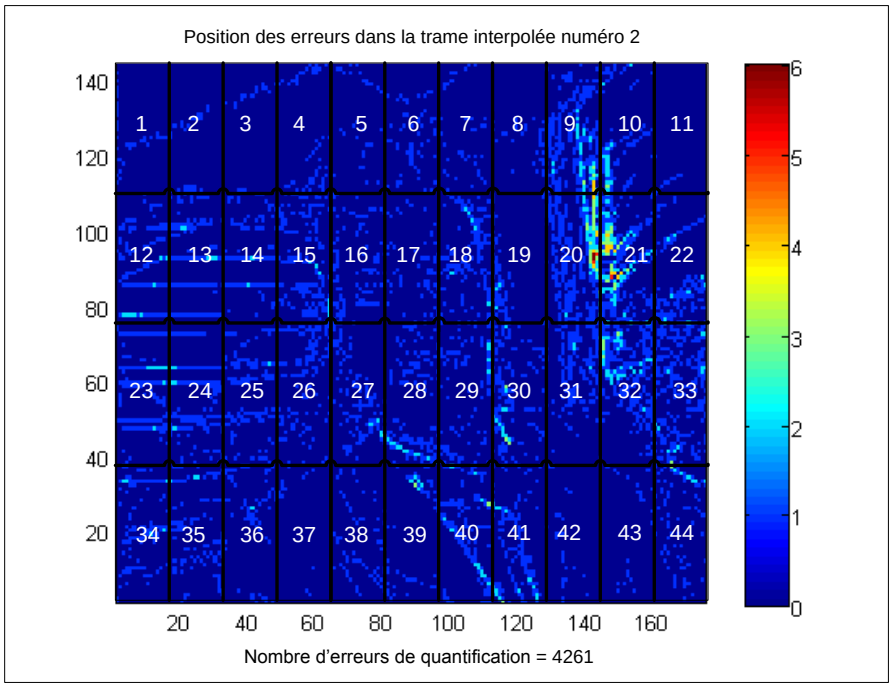

FIGURE 4.5: *Découpage de la trame en 44 régions.*

Une fois que le nombre de bits de parité par région est déterminé, le décodeur doit envoyer cette

répartition au décodeur à travers le canal de retour. Le nombre des bits de parité pour le premier encodeur par période de 8 prennent des valeurs dans l'intervalle $[1,\ 8]$. Ainsi, 3 bits sont utilisés pour désigner le nombre de bits de parité pour chaque région, soit un total de $3 \times 44 = 132$ bits que le décodeur doit transmettre à l'encodeur pour effectuer le poinçonnage.

4.3.2 Utilisation du canal de retour dans le fonctionnement du poinçonnage adaptatif

Le canal de retour constitue, en fait, l'un des inconvénients de l'architecture du codage vidéo distribué et plusieurs travaux se sont intéressés à son élimination ([30, 33]). Le rôle du canal de retour est d'assurer un contrôle du débit en permettant l'envoi progressif des bits de parité suivant le besoin du décodeur. Ce dernier va recevoir un certain nombre des bits de parité et va procéder au décodage turbo qui nécessite un délai temporel considérable. Si le décodeur turbo ne parvient pas à converger il demande à l'encodeur, par le biais du canal de retour, d'envoyer un nombre supplémentaire de bits de parité. Par la suite il recommence à nouveau le processus de décodage engendrant un délai temporel additionnel. Ce schéma est réalisé plusieurs fois jusqu'à la convergence du décodage turbo.

L'architecture de poinçonnage adaptatif présentée dans ce travail permet, non seulement de réduire le nombre de bits nécessaires pour la convergence du décodeur turbo mais aussi de réduire les délais temporels du processus de décodage turbo. En effet, un estimé initial du nombre des bits de parité nécessaire est envoyé à l'encodeur ce qui favorisera une convergence turbo dès les premières tentatives.

4.3.3 Détermination du nombre de bits de parité nécessaire pour chaque région

Comme spécifié ci-dessus, le décodeur détermine les bits de parité nécessaire pour la convergence turbo par l'observation de la différence entre les deux trames avoisinantes. En effet, lorsqu'il y a une grande différence entre ces deux trames, le processus d'interpolation est plus susceptible d'échouer et un nombre important de bits de parité doit être envoyé pour la correction des erreurs d'interpolation. Le décodeur détermine seulement la répartition des bits de parité pour le premier encodeur $n_{p_1} \in [1,\ 8]$ et qui est donné par :

$$n_{p_1}\left(region = r\right) = \min\left\{ \text{ceiling}\left[\underset{r}{\text{mean}}\left(\frac{|X_{2i-1} - X_{2i+1}|}{2}\right)\right], 8\right\} \tag{4.5}$$

où "ceiling" désigne la fonction qui permet d'arrondir vers l'entier le plus proche vers plus l'infini et la moyenne de la valeur absolue de la différence entre les pixels, $\underset{r}{\text{mean}}$, est à travers une région r de 36×16 =576 pixels. Par la suite le nombre des bits de parité pour le second encodeur est déduit selon la relation suivante :

$$\text{Si} \quad n_{p_1} \geq 4,\ n_{p_2} = \min\left\{\left(n_{p_1} + 1\right), 8\right\} \quad \text{sinon} \quad n_{p_2} = \max\left\{\left(n_{p_1} - 2\right), 0\right\} \tag{4.6}$$

4.4 Simulation et discussion

Dans cette section on décrit l'implémentation du schéma DVC avec poinçonnage adaptatif. Après la réception des deux trames clés avoisinantes, le décodeur commence par l'estimation du bruit dans

chacune des 44 régions et évalue le nombre de bits de parité nécessaire par une période de 8 bits, pour chaque région. On présente au tableau 4.1, la répartition des bits de parité utilisée pour la deuxième trame dans la séquence vidéo `Carphone` (première trame WZ). L'encodeur ne transmet que le nombre des bits de parité nécessaire par région pour le premier décodeur, à partir duquel le nombre des bits de parité par région nécessaire au second décodeur peut être déduit. Comme on peut le constater au tableau 4.1, les bits de parité envoyés par le premier encodeur sont plus dispersés que les bits de parité transmis par le second décodeur. L'entrelaceur pseudo-aléatoire du turbo code se chargera, en fait, de la dispersion des bits de parité pour le second décodeur.

Encodeur 1												Encodeur 2 (après désentrelacement)										
2	2	2	2	2	3	3	3	8	7	2		0	0	0	0	0	1	1	1	8	8	0
5	5	4	5	4	3	7	4	8	8	7		6	6	5	6	5	1	8	5	8	8	8
5	4	5	4	5	5	5	5	5	8	8		6	5	6	5	6	6	6	6	6	8	8
5	4	3	2	2	3	7	5	2	2	4		6	5	1	0	0	1	8	6	0	0	5

Tableau 4.1: *Distribution des bits de parité dans les 44 régions de la figure 4.5 pour les 2 encodeurs RSCs.*

L'impact de la redirection des bits de parité vers les régions bruitées s'observe, à la figure 4.6 , où l'on suit la convergence du processus de décodage turbo utilisant le poinçonnage adaptatif en comparaison avec celui utilisant un poinçonnage aléatoire, et ce, pour le même nombre total de bits de parité : cette figure montre clairement que le décodage avec poinçonnage adaptatif est plus performant que le décodage avec poinçonnage aléatoire. Pour la première technique de poinçonnage, les bits de parité sont principalement dirigés vers les régions 20 et 21 et vers les autres régions où la procédure d'interpolation exprime des difficultés. Un nombre moindre de bit de parité est réservé pour les régions non bruitées pour faire face aux quelques erreurs restantes et pour maintenir une certaine robustesse le long du treillis durant le processus de décodage log-MAP. En ce qui concerne la technique de poinçonnage aléatoire, où les bits de parité sont distribués aléatoirement le long du treillis, le nombre d'erreurs commence par décroître rapidement durant les deux premières itérations. Mais aussitôt que les erreurs isolées, submergées par un nombre de bit de parité plus ce que nécessaire, sont corrigées, le décodeur ne peut pas continuer à corriger les blocs d'erreurs avec le peu des bits de parité restants et se met à diverger à partir de la troisième itération.

À la figure 4.7, on trace l'évolution du taux de symbole en erreur au cours des itérations du processus de décodage turbo pour la technique de poinçonnage adaptatif et la technique de poinçonnage aléatoire. À partir de cette figure, on constate encore une fois l'intérêt de l'utilisation d'un dispositif de poinçonnage adaptatif pour la convergence du décodeur turbo sans avoir à demander un nombre de bits supplémentaires.

On compare, à la figure 4.8, le nombre de bits de parité nécessaire pour la convergence du turbo décodeur pour les 3 différents techniques de poinçonnage présentées. La technique de poinçonnage adaptatif permet d'obtenir une réduction significative du débit binaire comme on peut le constater au

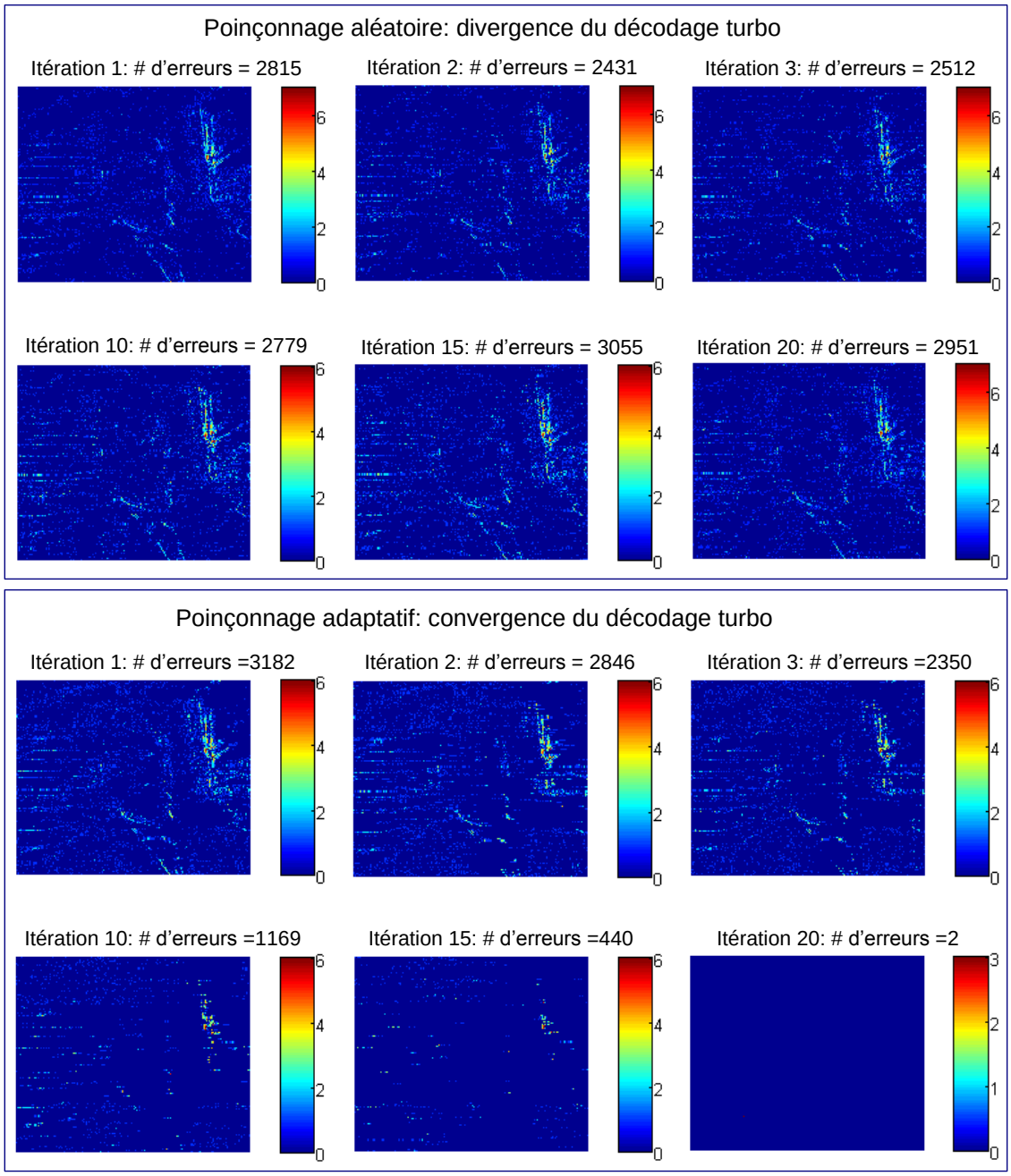

FIGURE 4.6: *Processus de décodage turbo pour la première trame WZ avec la technique de poinçonnage aléatoire et avec la technique de poinçonnage adaptatif pour le même nombre de bits de parité total égal à 27288.*

tableau 4.2. À noter que le débit des trames WZ est de 15 trames par seconde et que les 132 bits transmis par le décodeur, pour informer l'encodeur de la répartition des bits de parité par région, sont pris en considération.

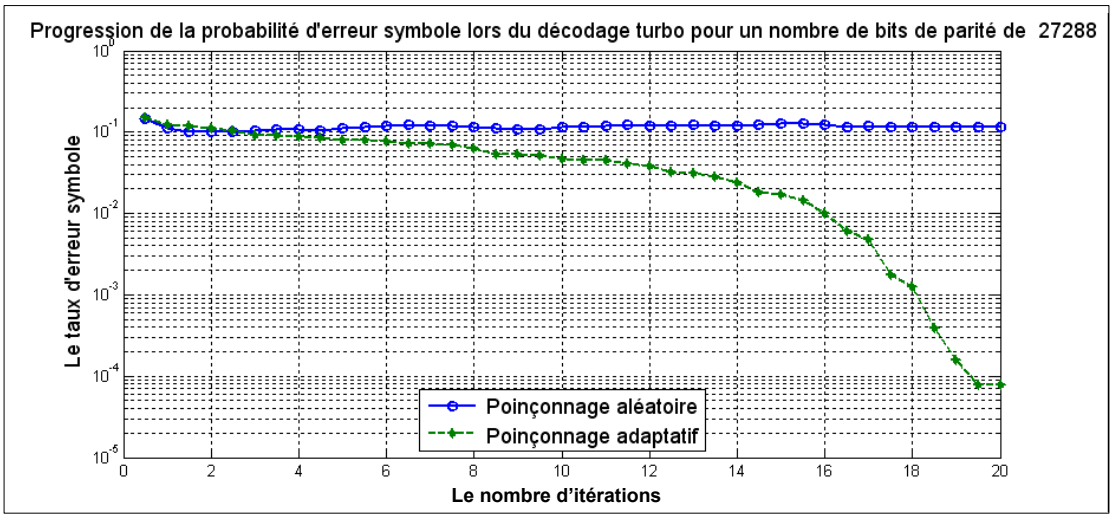

FIGURE 4.7: *Taux d'erreur par symbole lors du décodage turbo pour la première trame WZ avec la technique de poinçonnage aléatoire et avec la technique de poinçonnage adaptatif pour le même nombre de bits de parité total égal à 27288.*

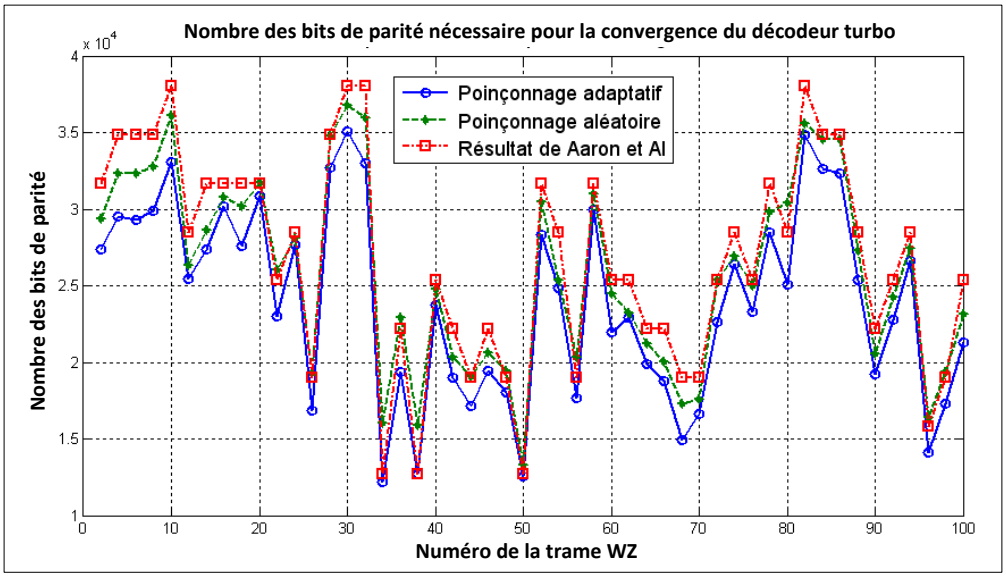

FIGURE 4.8: *Nombre de bits de parité nécessaire pour la convergence du décodage turbo pour les trames WZ de la séquence* `Carphone`.

Poinçonnage périodique Aaron et al. [22]	Poinçonnage aléatoire	Poinçonnage adaptatif
401.07 kbps	388.72 kbps	360.78 kbps

Tableau 4.2: *Débit binaire des bits de parité pour les trames WZ de la séquence* `Carphone`.

4.5 Conclusion

Dans ce chapitre, on a proposé un schéma de codage vidéo distribué permettant de mettre en place un poinçonneur adaptatif. Ce schéma est basé sur un raisonnement intuitif stipulant que diriger les bits de parité vers les endroits de la trame où ils vont être le plus efficace et éviter de gaspiller ces bits dans les régions non bruitées, permet d'assurer une réduction considérable du débit binaire de transmission. Les résultats expérimentaux confirment cette idée et une meilleure compression est obtenue. Cette technique de poinçonnage est réservée à un système de transmission où le décodeur peut prévoir approximativement les régions bruitées du canal et peut transmettre cette information à l'encodeur avec le moindre nombre de bits possible. On retrouve bel et bien ces conditions dans l'architecture de codage vidéo distribué.

Chapitre 5

Codage vidéo distribué dans le domaine des transformées

5.1 Introduction

Le passage du domaine pixel vers le domaine des transformées est une technique efficace déployée par les standards de codage vidéo conventionnel vu la réduction de l'interdépendance entre les différentes composantes transformées et l'amélioration de la compacité des données en concentrant la majeure partie de l'énergie dans un nombre réduit de valeurs [1]. Le passage dans le domaine des transformées a été également adopté dans les architectures de codage vidéo distribuée en utilisant la transformée en cosinus discrète (DCT : discrete cosine transform). Le choix de cette transformée revient à ses atouts du point de vue pratique qui lui ont valu la standardisation dans la préparation du codeur vidéo H.264/AVC. Dans ce chapitre on présente, d'abord, l'architecture du système DVC dans le domaine des pixels avec extraction des plans de bits. Par la suite, on introduit les principales architectures du système DVC dans le domaine des transformées. Chaque architecture se distingue principalement par une conception différente du quantificateur pour les coefficients AC. Finalement on compare entre les techniques de reconstruction standard et optimale avant de s'intéresser à leur implémentation et à l'interprétation des résultats de simulations.

5.2 Codeur/encodeur Wyner-Ziv dans le domaine pixel

La première architecture de codage vidéo distribué, présenté par un groupe de recherche à l'Université de Stanford [22] est dans le domaine pixel avec un codage turbo de rapport 4/5 effectué au niveau des symboles. La procédure de codage et de décodage turbo considère tous les bits de quantification d'un symbole. Cette architecture dans le domaine pixel peut aussi utiliser un code turbo de rapport 1/2 mais en considérant l'extraction de plans de bits permettant de traiter en premier lieu les bits de poids le plus fort et ensuite les bits de poids suivants jusqu'aux bits de plus faibles poids. Pour le décodage d'un plan de bits donné, on se sert des plans de bits précédents déjà décodés. Le schéma bloc du codec

Wyner-Ziv dans le domaine pixel avec extraction des plans de bits est présenté à la figure 5.1

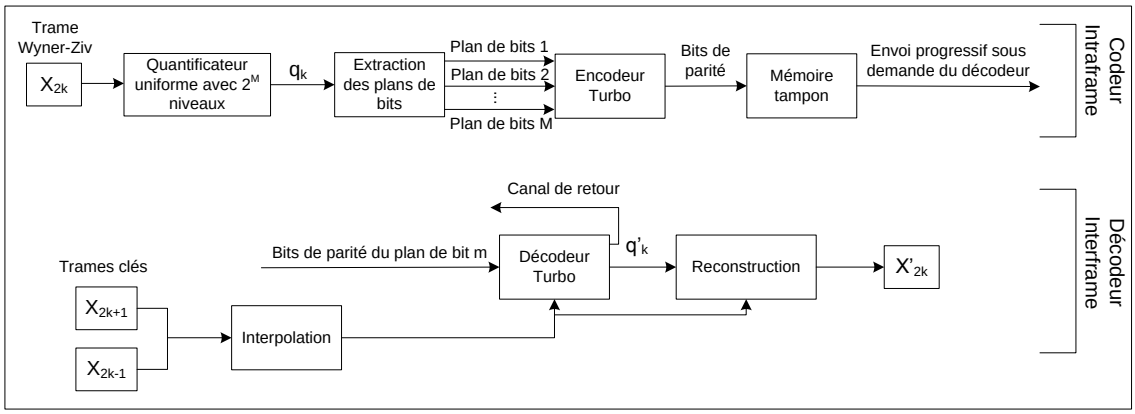

FIGURE 5.1: *Codeur et décodeur Wyner-Ziv dans le domaine pixel.*

5.2.1 Codeur intraframe

Les pixels des trames Wyner-Ziv sont fournis à un quantificateur uniforme avec une plage dynamique $[0,\ldots,255]$ et 2^M niveaux de quantification. Les plans de bits des symboles quantifiés q_k sont formés et envoyés à un encodeur turbo qui génére les bits de parité et élimine les bits systématiques. Chaque plan de bits est traité séparément par le codeur turbo : ceci équivaut à une longueur de code de $L = 144 \times 176 = 25344$ avec 144 lignes de la trame QCIF et 176 colonnes. Les bits de parité sont finalement envoyés graduellement à l'aide d'une matrice de poinçonnage périodique de longueur 32 sous la demande du décodeur. On vérifie bien que la longueur $L = 25344$ du code est un multiple de la longueur de la matrice de poinçonnage 32. Plus la valeur de la longueur de la matrice de poinçonnage est élevée, plus le contrôle du débit est de granularité fine. Cependant, le nombre de sollicitations du canal de retour devient important, ce qui affecte les délais du processus de décodage turbo.

5.2.2 Décodeur interframe

Le décodeur disposant des trames clés adjacentes aux trames WZ (Wyner-Ziv) génère l'information latérale moyennant un bloc d'interpolation. L'information latérale sert d'abord au décodage des symboles de quantification des différents plans de bits. Le décodage d'un plan de bits est considéré effectué avec succès si le taux d'erreur binaire est inférieur à un seuil de 10^{-3}. Dans le cas contraire, une demande d'envoi de bits de parité supplémentaires est transmise à l'encodeur. Les plans de bits sont décodés séparément en commençant par le plan de poids le plus fort jusqu'au plan de poids le plus faible. Le résultat de décodage d'un plan de bit donné sert pour le décodage des plans de bits suivants. Après le décodage de tous les plans de bits, ces derniers sont regroupés pour reformer les symboles de quantifications des trames WZ. Ces symboles ainsi que l'information latérale sont fournis au module de reconstruction pour l'estimation des valeurs des pixels des trames envoyées. L'information latérale intervient alors à deux reprises : lors du décodage turbo et lors de la reconstruction.

5.3 Compression par passage du domaine pixel au domaine transformé

L'architecture DVC dans le domaine pixel considère les données à envoyer comme telles sans exploitation de la nature statistique de l'image qui explicite une forte corrélation spatiale. En effet une image, de façon générale, n'est pas un signal complètement aléatoire où les pixels adjacents peuvent varier de façon quelconque indépendamment les uns des autres. Pour exploiter cette caractéristique de l'image, l'idée consiste à passer dans un domaine transformé approprié.

5.3.1 Transformation de Karhunen-Loève

La transformation KL (Karhunen-Loève), aussi connue sous le nom de la transformation de Hotelling, est la transformation optimale pour exploiter la corrélation des données à transmettre. Pour une image donnée, le calcul de la transformée KL se fait comme suit [46]. On découpe l'image en blocs de $N \times N$ et on considère que chaque bloc constitue un vecteur de longueur N^2. Les vecteurs propres de la matrice d'autocorrélation de ces vecteurs représentent la base de la transformée KL. On remarque que la transformée KL dépend des données : les vecteurs de bases obtenus diffèrent donc d'une image à l'autre comme on le remarque à la figure 5.2 où l'on a tracé les vecteurs de base de la transformée KL pour trois images différentes de la même séquence vidéo `Foreman` et une autre image de la séquence vidéo `Salesman`.

On remarque que la base KL pour la première trame et la deuxième trame de la séquence vidéo `Foreman` sont similaires dans les basses fréquences avec une certaine différence pour des fréquences plus élevées. Il y a une forte similitude, partout, entre deux trames successives sauf dans les régions où il y a un mouvement qui se traduit par un déplacement d'une arête. Le contenu basse fréquence, représenté dans le coin supérieur gauche de la base KLT, reste donc presque le même pour les deux premières trames de la séquence `Foreman` avec un changement dans certaines bandes de hautes fréquences se trouvant dans le coin inférieur droit. Cette transformation dépend alors du contenu fréquentiel des données. Cependant on remarque une certaine similitude de la base de la transformation KL pour différentes trames et pour différentes séquences comme on peut le voir dans les quatre bases représentées à la figure 5.2. Même si cette transformation est optimale pour la décorrélation des données, du point de vue pratique elle présente deux inconvénients majeurs. Le calcul et la mise à jour de la transformée aussitôt que la nature des données initiales change, augmentent la complexité calculatoire et la transmission de la transformée au décodeur réduit le gain de compression réalisé par la décorrélation optimale des données. Cette transformation reste tout de même utile pour avoir une idée sur la limite des performances que l'on peut atteindre.

5.3.2 Transformation en cosinus discrète

Pour remédier aux inconvénients de la transformée KL tout en bénéficiant de la décorrélation spatiale des images à transmettre, la transformée en cosinus discrète est une solution pertinente. En effet, la transformation DCT est indépendante des données initiales. Ainsi, il n'est plus nécessaire ni de

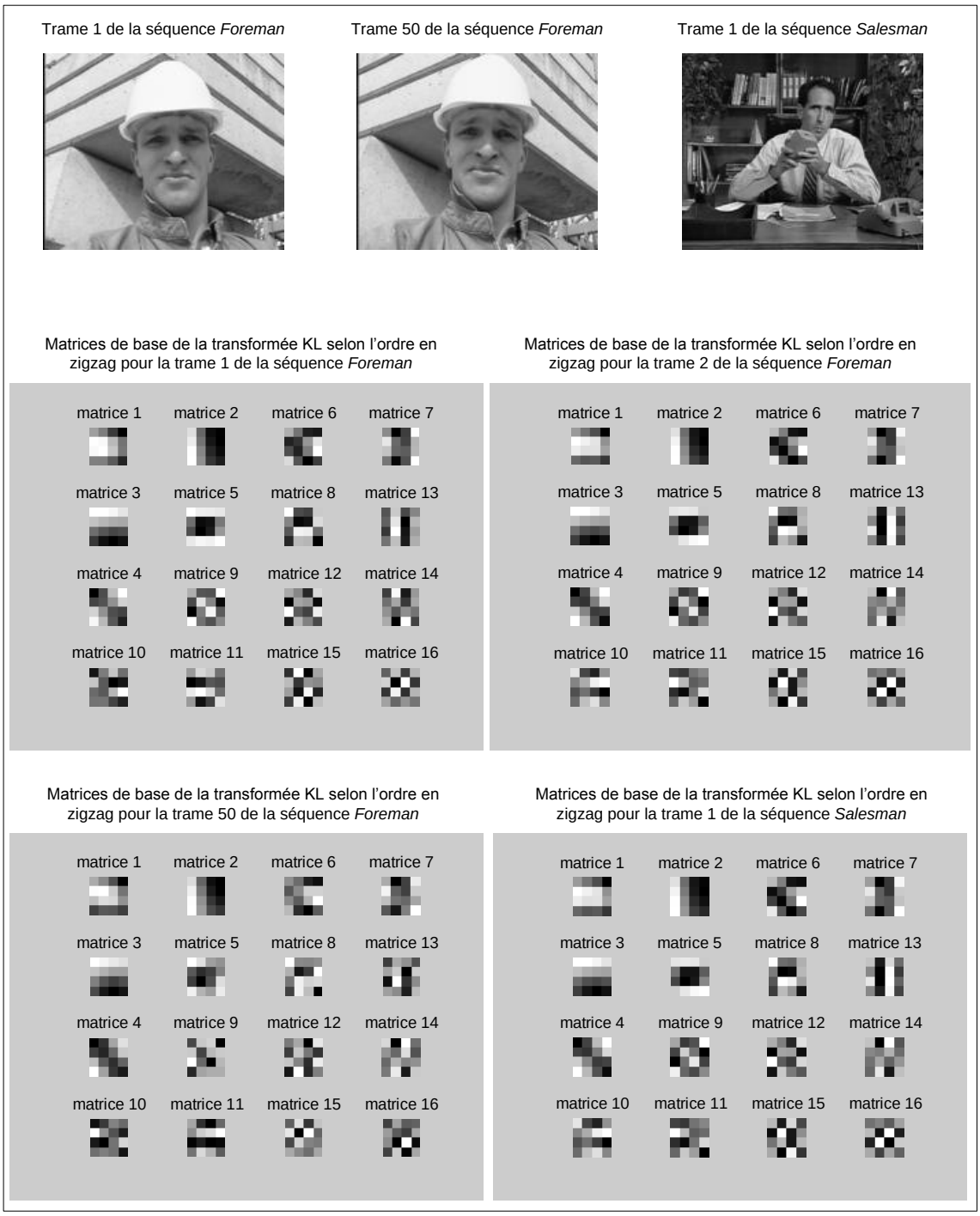

FIGURE 5.2: *Matrices de base de la transformation Karhunen-Loève (KL).*

calculer les vecteurs de base de la DCT ni de les transmettre au décodeur. La transformation en cosinus discrète détient son appellation du fait que les lignes de sa matrice de transformation sont déterminées

à l'aide de fonction cosinus. Ces coefficients sont déterminés comme suit [46] :

$$[C]_{i,j} = \begin{cases} \sqrt{\frac{1}{N}} \cos \frac{(2j+1)i\pi}{2N} & i = 0,\ j = 0,\ 1,\ ...,\ N-1 \\ \sqrt{\frac{2}{N}} \cos \frac{(2j+1)i\pi}{2N} & i = \ 1,\ 2,\ ...,\ N-1,\ j = 0,\ 1,\ ...,\ N-1 \end{cases} \tag{5.1}$$

Les matrices de bases de la transformation DCT sont présentées à la figure 5.3. Ces matrices sont déterminées de façon indépendante des données à transmettre. Chaque bloc 4×4 d'une image dans le domaine pixel peut être représenté par une combinaison linéaire unique des matrices de bases de la DCT. Généralement une image présente une corrélation spatiale importante qui se traduit par la ressemblance entre pixels voisins. Ainsi lors de la décomposition d'un bloc 4×4 en éléments de la base DCT, les composantes de basses fréquences contiennent la majeure partie de l'information et les coefficients qui leur sont alloués sont élevés. En envoyant les coefficients de basses fréquences et en éliminant les coefficients de hautes fréquences, on réalise une compression au prix d'une faible distorsion. Cette distorsion est due à l'élimination du contenu haute fréquence de l'image auquel le système visuel humain (VHS), qui se comporte comme un filtre passe bas, est moins sensible.

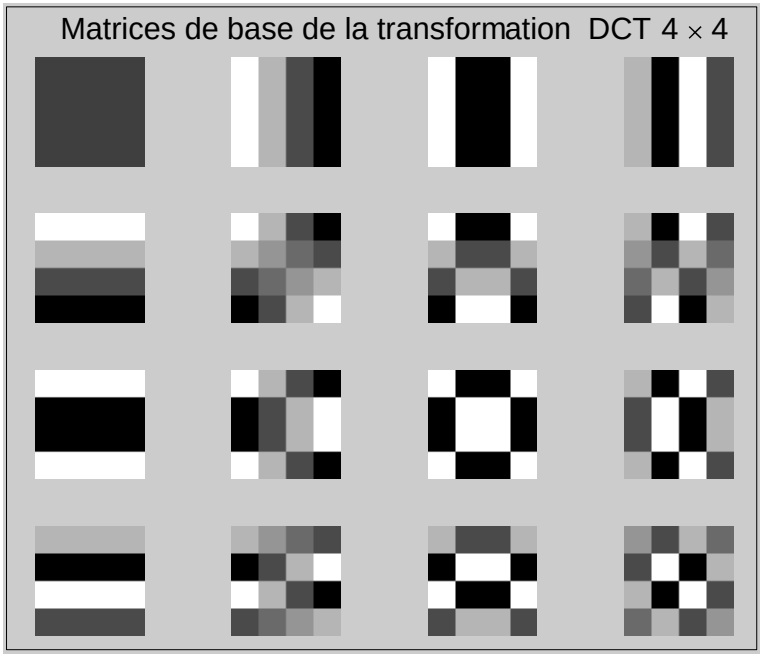

FIGURE 5.3: *Matrices de base de la DCT.*

5.4 Codeur/encodeur Wyner-Ziv dans le domaine transformé

Le passage du domaine pixel au domaine transformé est une solution adoptée par les codeurs vidéo standards pour bénéficier du gain de la décorrélation des données à transmettre. La transformation DCT 4×4 répond bien au compromis performance de compression par rapport au temps de calcul. La transformation DCT 4×4 a été utilisée lors de la standardisation du codeur vidéo AVC/H264. La

transformation DCT 4×4 a été aussi considérée dans l'architecture du codeur vidéo distribué et le schéma bloc de la figure 5.1 a été modifié en rajoutant le module de la DCT et de la DCT inverse. L'architecture du système de codage vidéo distribué dans le domaine transformé est donnée à la figure 5.4.

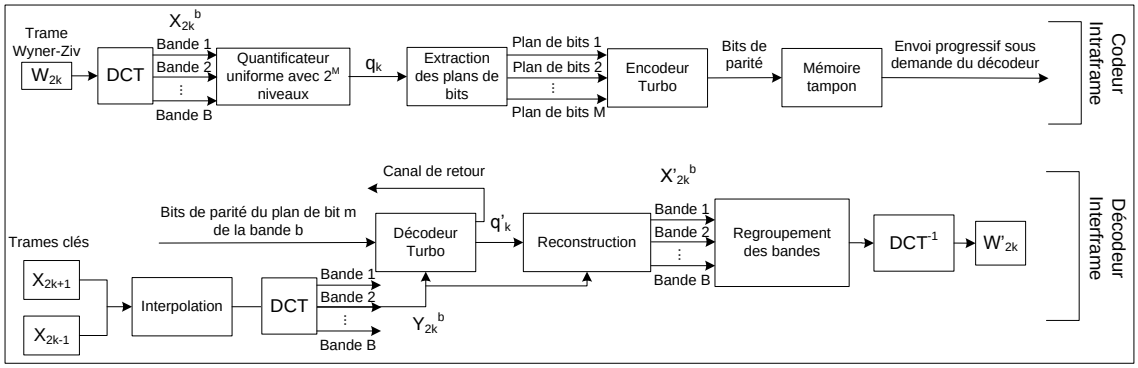

FIGURE 5.4: *Codeur et décodeur Wyner-Ziv dans le domaine transformé.*

5.4.1 Codeur interframe

Au lieu de transmettre les pixels comme tels, on effectue la transformation DCT 4×4 permettant d'avoir les coefficients de chacune des 16 bandes de fréquences. Les bandes sont ensuite traitées indépendamment en effectuant la quantification et l'extraction des plans de bits ainsi que le codage turbo. De ce fait, la taille du code turbo et de son entrelaceur, qui était dans le domaine pixel égale à la taille de la trame $L = 25344$, devient dans le domaine transformée égale au rapport de la taille de la trame et du nombre des bandes de fréquence $L = 25344/16 = 1584$. La période de la matrice de poinçonnage passe de 32 dans le domaine pixels à 48 dans le domaine transformée. On remarque ici que 32 divise 25344 mais ne divise pas 1584. Par contre 48 divise 1584 et offre plus de granularité pour le contrôle du débit.

5.4.2 Décodeur interframe

Pour générer l'information latérale le décodeur procède, tout d'abord, à l'interpolation des trames adjacentes. Ensuite la trame interpolée est projetée dans le domaine transformée (DCT 4×4) pour obtenir les 16 coefficients transformés de chaque bande. Finalement, l'information latérale consiste aux coefficients de la bande b qui est en cours de décodage. La procédure de décodage turbo dans le domaine transformé est alors similaire à celle dans le domaine pixels sauf que les données ne sont plus les pixels mais plutôt les coefficients de chaque bande et l'information latérale n'est plus la trame interpolée mais sa transformée. Après avoir décodé et regroupé les coefficients des bandes transformées, la transformation DCT inverse est effectuée pour regagner le domaine pixel.

5.4.3 Matrices de quantification

Le passage dans le domaine transformé permet de représenter de façon plus efficace les données à transmettre en les concentrant dans un certain nombre de coefficients. En effet, les coefficients des bandes de basses fréquences ont une variance et une plage dynamique plus importante que les coefficients des bandes des hautes fréquences. Pour avoir une idée sur le comportement des coefficients des 16 bandes de fréquences de la transformation DCT 4×4, on représente à la figure 5.5, les histogrammes de ces bandes relativement aux 101 premières trames de la séquence vidéo `Foreman`.

La procédure de quantification doit alors non seulement favoriser l'envoi des bandes les plus représentatives des données mais aussi assurer une faible distorsion pour les coefficients à fortes variances. Pour ce faire, Aaron *et al.* [26] ont défini un ensemble de 7 matrices de quantification, pour différents débits, allouant plus de niveaux de quantification pour les coefficients des bandes des basses fréquences. On reproduit ces matrices dans la figure 5.6 tout en rajoutant une 8^{me} matrice proposée par Brites *et al.* [8] pour les hauts débits.

Les 8 matrices de quantification ou indices de quantification ($Q_i = 1, \ldots, 8 : i = 1$ pour le faible débit et $i = 8$ pour le haut débit) de la figure 5.6 représentent le nombre de niveaux de quantification de chaque coefficients des 16 bandes de la DCT. En utilisant un nombre élevé de niveaux de quantification, on réduit la distorsion au prix d'une augmentation du débit. La valeur 0 indique que le codeur n'envoie pas des bits de parité pour la bande en question. La reconstruction se fait alors en considérant directement la valeur de l'information latérale relative à cette bande.

5.4.4 Modèle de corrélation du canal virtuel

Le codeur turbo envoie seulement les bits de parité générés et les bits systématiques ne sont pas transmis pour augmenter la compression. Le rôle des bits systématiques, lors de la procédure de décodage turbo consiste à calculer les valeurs de vraisemblance de canal. Ce rôle est remplacé par l'information latérale qui est représentée par les coefficients des bandes DCT des trames interpolées dans le cas du domaine transformé. La figure 5.7 représente la modélisation de la corrélation entre les coefficients des bandes de la DCT des trames WZ et les coefficients des bandes DCT des trames interpolées par une distribution Laplacienne de paramètre α et, à titre de comparaison, par une distribution Gaussienne d'écart-type σ. Ce paramètre varie d'une bande à une autre. Dans cette figure, on essaie d'approximer l'histogramme de la différence entre les coefficients des 16 bandes et les coefficients représentant l'information latérale pour les 101 premières images de la séquence vidéo `Foreman`. Dans les simulations effectuées dans [8], l'estimation du paramètre α pour chacune des bandes DCT se fait "hors ligne" pour la séquence vidéo entière moyennant l'équation suivante :

$$\begin{cases} \alpha_k = \frac{\sqrt{2}}{\sigma_k} \\ \sigma_k^2 = \text{ var}\left(b_k^{WZ} - b_k^{SI}\right) \end{cases} \tag{5.2}$$

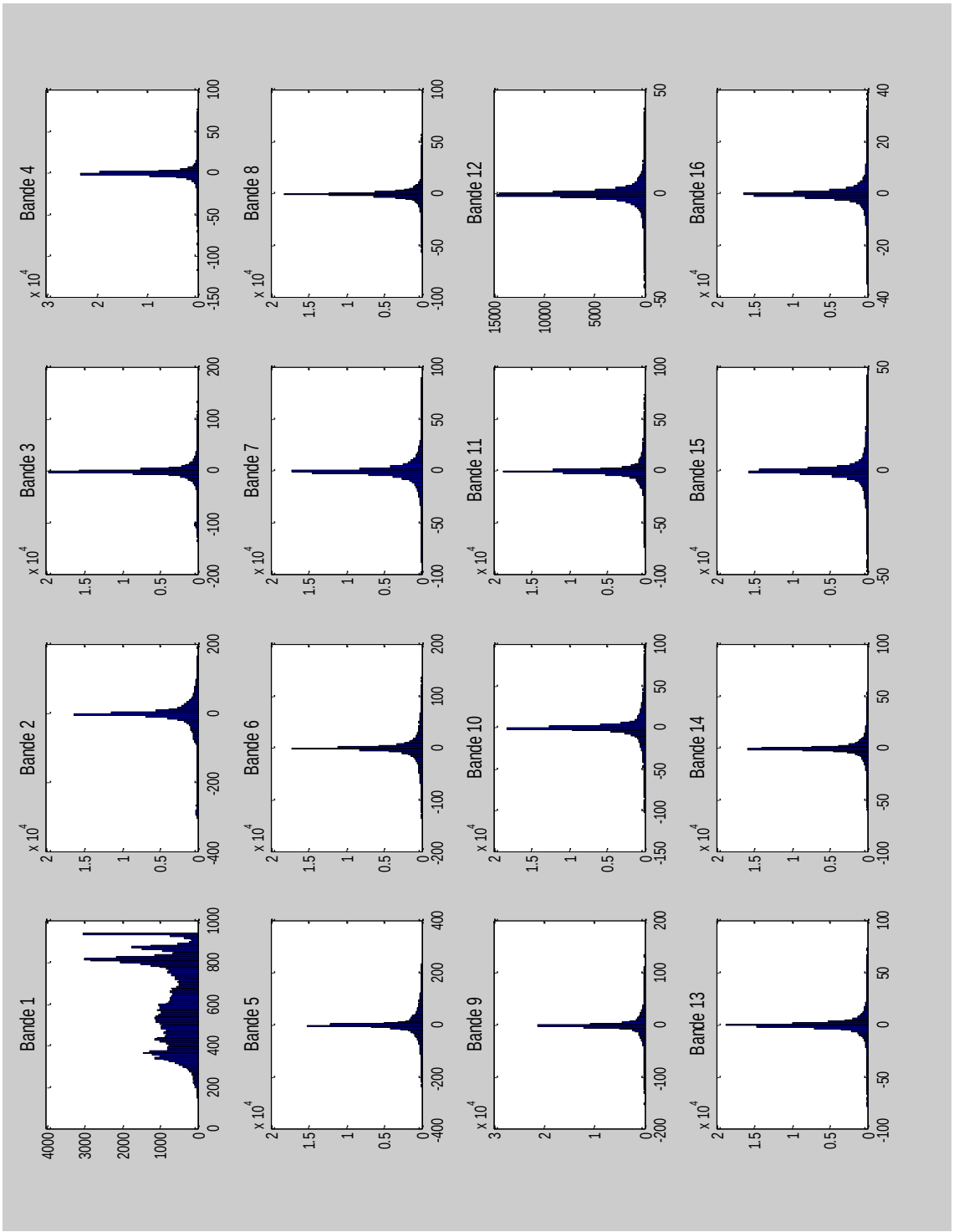

FIGURE 5.5: *Histogrammes des coefficients des 16 bandes de fréquences de la DCT* 4×4 *pour la séquences vidéo* `Foreman`.

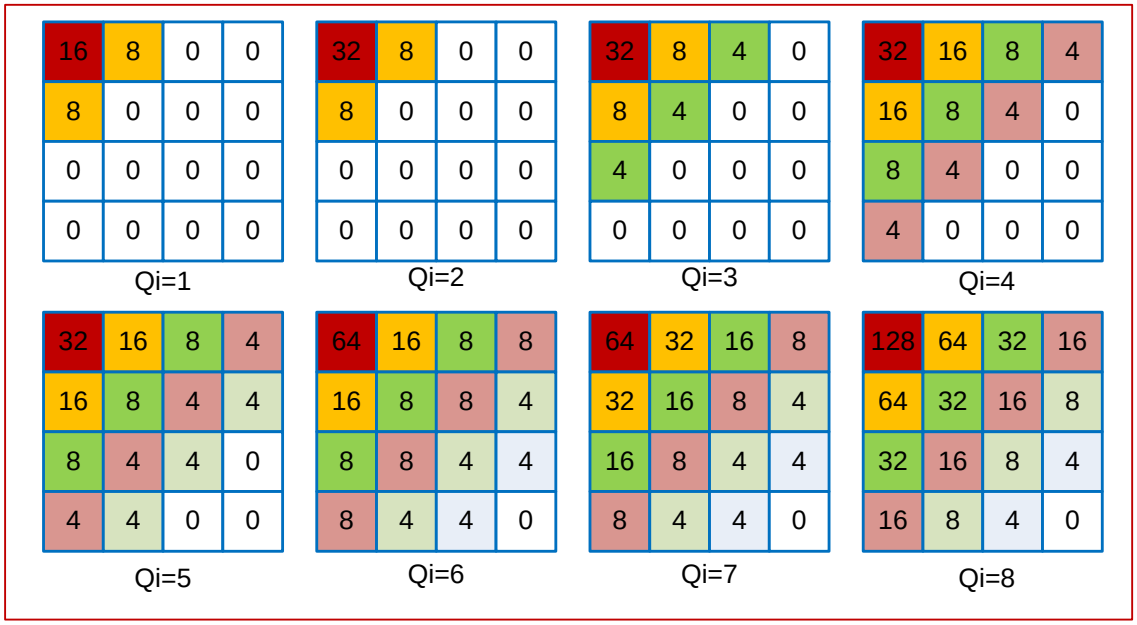

FIGURE 5.6: *Ensemble de matrices de quantification pour différents performances de débits distortion du codeur Wyner-Ziv dans le domaine transformé.*

où b_k^{WZ} désigne le coefficient de la $k^{\text{ème}}$ bande de la DCT de la trame WZ, b_k^{SI} désigne le coefficient de la bande k de la DCT de la trame interpolée, "var" désigne la variance et σ_k l'écart type.

Le modèle de corrélation du canal virtuel d'un système de codage vidéo distribué utilisé dans la littérature se base généralement sur une distribution Laplacienne. En effet l'allure de la distribution Laplacienne a tendance à bien suivre l'allure de l'histogramme de la différence entre les données originales et l'information latérale. De plus l'expression analytique de la fonction de densité de probabilité (pdf) et de la fonction de répartition cumulative (cdf) sont relativement simples. Ceci rend pratique l'utilisation de la loi Laplacienne pour le calcul des valeurs de vraisemblance du canal virtuel lors du processus du décodage turbo. Dans ce qui suit, on propose d'évaluer l'adéquation de la distribution Laplacienne pour modéliser le canal virtuel en la comparant avec une distribution gaussienne par le biais d'un test statistique de chi carré [47] :

$$\chi^2 = \sum_i \frac{(O_i - E_i)^2}{E_i} \tag{5.3}$$

où les O_i représentent les valeurs observées ou empiriques et les E_i les valeurs attendues ou valeurs théoriques. Au final, on peut conclure que la distribution Laplacienne est plus appropriée pour modéliser le canal virtuel que la distribution Gaussienne. Mais cela n'empêche pas la possibilité d'existence d'autres distributions qui réussiront mieux le test du χ^2 que la Laplacienne. On utilise, ici, la distribution Laplacienne car elle permet d'appliquer correctement l'algorithme de décodage turbo et l'utilisation d'une autre distribution plus précise ne va pas forcément augmenter les performances du décodeur turbo.

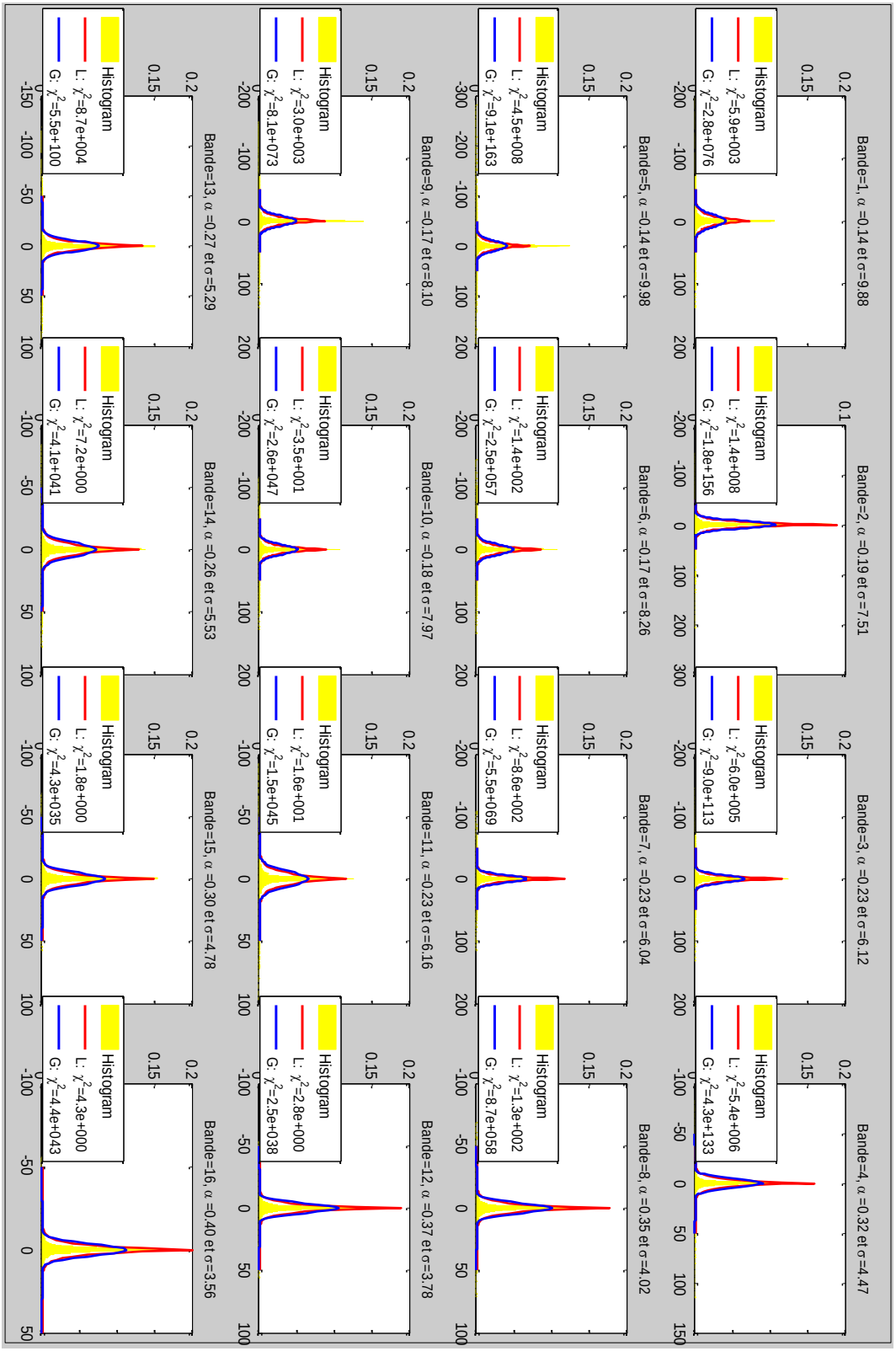

FIGURE 5.7: *Approximation des histogrammes de la différence entre les coefficients des 16 bandes de la DCT des trames WZ et les coefficients de la DCT des trames interpolées pour la séquences vidéo* `Foreman` *par une distribution Laplacienne et une distribution Gaussienne*

5.4.5 Plage dynamique et disposition des intervalles de quantification

Dans le domaine pixel, les valeurs à transmettre sont quasi uniformément distribuées dans l'intervalle $[0,\ldots,255]$. Donc la structure du quantificateur est triviale et consiste à découper l'intervalle $[0,\ldots,255]$ en intervalles égaux. Dans le domaine transformé, on remarque à partir de la figure 5.5 que les coefficients de la bande DC sont quasi uniformément distribués dans l'intervalle $[0,\ldots,1023]$ et que les valeurs des coefficients AC sont principalement concentrés autour de zéro et s'étendent sur une plage dynamique qui change d'une bande à l'autre. Dans ce qui suit on décrit les 3 principaux quantificateurs utilisés dans la littérature avant de proposer un 4^{me} quantificateur qui permet d'obtenir une certaine amélioration du point de vue débit-distorsion.

Quantificateur de Aaron *et al.*

Le système de codage vidéo distribué dans le domaine transformé avec une DCT 4×4 a été au départ proposé par Aaron *et al.* [26]. Les auteurs utilisent un quantificateur scalaire uniforme pour les coefficients de la bande DC et de la bande AC. La plage dynamique a été fixée en traçant les histogrammes de diverses séquences vidéo. On ne spécifie pas cependant les séquences vidéos utilisées ainsi que plusieurs détails quant à l'implémentation des codeur et décodeur WZ dans le domaine transformé [48].

Quantificateur de Brites *et al.*

Brites *et al.* [8] ont défini une autre technique de quantification. Le quantificateur proposé fonctionne comme suit. Pour les coefficients de la bande DC, ils utilisent un quantificateur scalaire uniforme avec une plage dynamique allant de 0 à 1023. Ce choix est fondé dans la mesure où les coefficients de la bande DC sont distribués sur toute la plage comme on peut le voir à la figure 5.5. Pour les coefficients des bandes AC, l'information sur la plage dynamique est envoyée pour chaque trame en prenant le maximum de la valeur absolue de tous les coefficients de la bande en question. Donc la plage dynamique est symétrique autour de 0. Étant donné que les coefficients des bandes AC sont concentrés autour de 0 (voir la figure 5.5), le choix d'un quantificateur avec un intervalle symétrique autour de 0 permet de réduire considérablement le débit requis. En effet, si l'information originale est concentrée autour de 0, alors l'information latérale est aussi concentrée autour de 0 avec une certaine différence modélisée par une Laplacienne. Donc en prenant un quantificateur avec un intervalle centré autour de 0, un grand nombre de valeurs de l'information originale et de l'information latérale vont avoir le même symbole de quantification. Ceci réduit l'erreur entre la quantification de l'information originale et latérale et par conséquent le nombre de bits de parité à envoyer ultérieurement pour corriger l'information latérale. Un quantificateur symétrique autour de 0 pour les coefficients de la bande AC permet alors, de réduire considérablement le débit au prix d'une faible distorsion limitée par l'intervalle de quantification autour de 0. Un exemple de ce quantificateur pour les coefficients de la bande DC et les coefficients des bandes AC est illustré à la figure 5.8 pour un nombre de bits de quantification égal à 3.

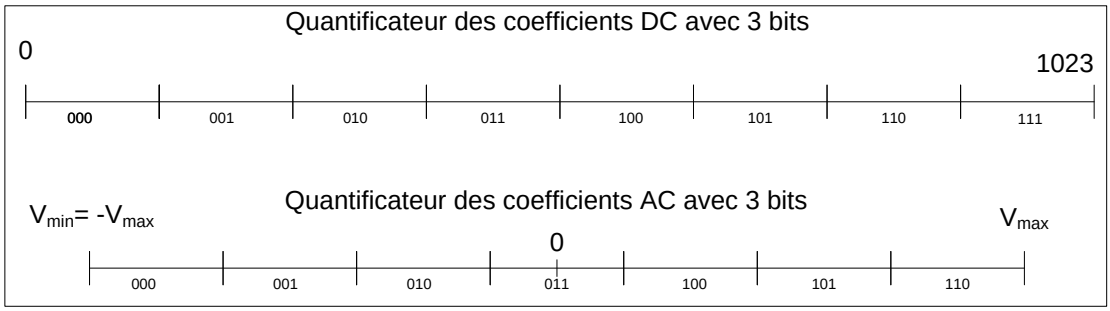

FIGURE 5.8: *Quantificateur de Brites et al. [8] pour les coefficients DC et AC.*

Quantificateur de Discover

Le quantificateur proposé par les auteurs de [9] reprend les mêmes caractéristiques que le quantificateur de Brites avec la seule différence que l'intervalle autour de zéro est dédoublé comme le montre la figure 5.9. Ceci permet de réduire encore le débit de transmission mais en augmentant légèrement la distorsion. Les performances globales du quantificateur de Discover [9] dépassent, cependant, les performances du quantificateur de Brites.

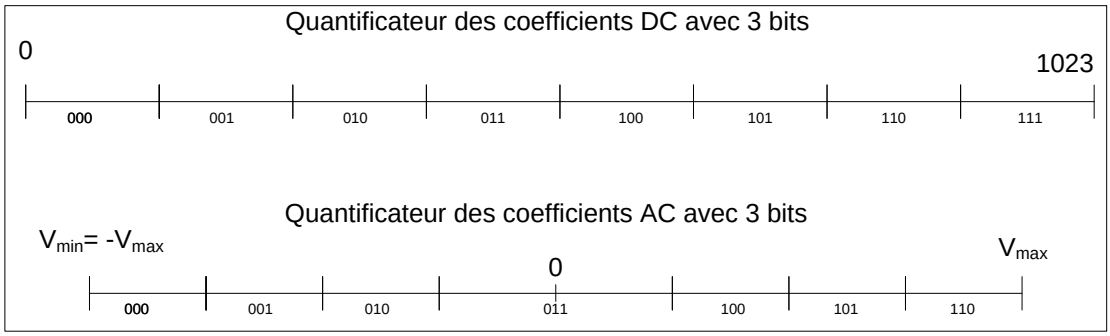

FIGURE 5.9: *Quantificateur de Discover [9] pour les coefficients DC et AC.*

Quantificateur proposé

Dans le cas d'un système de communication habituel, l'algorithme de Lloyd-Max permet de construire un quantifieur scalaire optimal. Cet algorithme prend en considération la distribution des données à envoyer et place plus de symboles de quantification dans les régions de densité de probabilité élevée et moins de symboles dans les régions de densité de probabilité faible. Dans le cas d'un système de communication distribué avec information latérale au décodeur, l'algorithme de Lloyd-Max permet d'optimiser les performances du point de vue distorsion mais le débit augmente de façon démesurée.

Le quantificateur de Discover fonctionne, en quelque sorte, suivant la logique inverse du quantificateur de Lloyd-Max, c'est-à-dire de garder le moins de symboles de quantification autour de 0, là où

la concentration des données originales est élevée en doublant l'intervalle de quantification autour de 0. Or les questions suivantes se posent : *À quel point peut-on augmenter l'intervalle de quantification autour de 0 afin de réduire le débit sans toutefois détériorer les performances du point de vue de la distorsion ? Et, de façon plus générale, qu'elle est la disposition optimale des intervalles de quantification ?*

La détermination systématique d'un quantificateur optimal dans le cas distribué reste un problème assez complexe dans la mesure où l'on doit prendre en considération non seulement la distribution des données originales mais aussi la corrélation qui relie ces dernières avec l'information latérale. On a essayé par des simulations, diverses dispositions des intervalles de quantification et quoi que l'on soit parvenu, dans certains cas, à obtenir une amélioration des performances, la démarche reste empirique et les résultats non concluants.

L'idée du quantificateur proposé consiste à faire un choix adaptatif entre le quantificateur de Brites et le quantificateur de Discover concernant les coefficients des bandes AC. Pour les coefficients de la bande DC, le même quantificateur scalaire uniforme est toujours considéré. La largeur de l'intervalle du quantificateur DC W_{DC} est, par contre, utilisée pour la détermination de la largeur de l'intervalle autour de 0 des coefficients AC en utilisant l'équation empirique suivante :

$$W_{AC}^{0} = \begin{cases} \frac{V_{\max}-V_{\min}}{2^{nbBits}-1}, & \text{si} \quad 3\times W_{DC} < \frac{V_{\max}-V_{\min}}{2^{nbBits}-1} \\ 3\times W_{DC}, & \text{si} \quad \frac{V_{\max}-V_{\min}}{2^{nbBits}-1} < 3\times W_{DC} < 3\frac{V_{\max}-V_{\min}}{2^{nbBits}} \\ 3\frac{V_{\max}-V_{\min}}{2^{nbBits}}, & \text{si} \quad 3\times W_{DC} > 3\frac{V_{\max}-V_{\min}}{2^{nbBits}}. \end{cases} \tag{5.4}$$

Ici, $nbBits$ désigne le nombre de bits alloués pour la quantification du coefficient de la bande AC en question.

Le nombre de bits de quantification des coefficients des 16 bandes est présenté par les différentes matrices de la figure 5.6. W_{DC} représente la plage de valeurs du quantificateur scalaire uniforme des coefficients de la bande DC. $V_{\max} = -V_{\min}$ définit la plage dynamique des coefficients d'une bande donnée. Les différentes valeurs de $V_{\max}$ correspondant à chacune des bandes de fréquences sont envoyées du codeur à l'encodeur pour chacune des trames de la séquence vidéo. W_{AC}^{0} est la largeur de l'intervalle de quantification autour de 0 et le reste de la plage dynamique est partagé identiquement entre le restant des symboles de quantification.

Ce quantificateur permet de garder une certaine concordance entre la précision du quantificateur des coefficients DC et de la précision des quantificateurs des coefficients AC. On évite ainsi d'avoir d'un côté un quantificateur DC à faible largeur d'intervalle de quantification (correspondant à un faible niveau de distorsion mais un débit élevé) et un quantificateur d'un certain coefficient AC avec un intervalle autour de 0 beaucoup plus grand. Cette démarche dans la spécification du quantificateur proposé est basée principalement sur des simulations au cours desquelles diverses autres dispositions des intervalles de quantification ont été essayées sans toutefois donner un gain de performance consi-

dérable. La formulation théorique du problème de la détermination d'un quantificateur optimal dans le cas des systèmes DVC diffère de celle dérivée pour les cas de transmission habituels qui trouvent leur optimalité dans le quantificateur de Max-Lloyd.

5.5 Simulations et discussion

Après avoir étudié les principes d'un codeur Wyner-Ziv dans le domaine transformé, on présente dans cette partie les détails et les résultats des simulations effectuées. Ces simulations ont pour but de comparer entre les performances d'un codeur/encodeur DVC dans le domaine pixel et dans le domaine transformé. Dans le domaine transformé, on présente les résultats des divers quantificateurs utilisés pour les coefficients des bandes AC en utilisant la méthode de reconstruction standard et la méthode de reconstruction optimale [32]. En plus de la mesure de distorsion habituelle, le PSNR, qui traduit la qualité objective des trames WZ, on introduit la mesure de distorsion SSIM (Structural SIMilarities) présentée dans [10] : celle-ci permet de juger de la qualité subjective des trames WZ.

5.5.1 Matériels et données d'expérimentation

Les séquences vidéo utilisées dans les simulations sont sous le format QCIF (*Quarter Common Intermediate Format*) et ont une résolution de 176×144 et une fréquence d'affichage d'images correspondant à 30 trames par secondes. Ces séquences peuvent être téléchargées à partir de divers sites sous le format YUV [49, 50, 51, 52, 53]. Cependant, on a remarqué que la même séquence vidéo obtenue de deux sites différents peut présenter une certaine différence. Pour cette raison, on précise que les simulations effectuées dans ce chapitre utilisent des séquences vidéo obtenues du site Web du centre de l'Université de Stanford pour l'ingénierie des systèmes et des images [53]. Étant donné que la fréquence d'affichage des images des séquences vidéo utilisées est de 30 trames par seconde, la fréquence d'affichage des trames WZ correspond à 15 trames par seconde. Dans les courbes de performances présentées dans cette section, on ne considère que le débit relatif à la transmission des trames WZ (bits de parité nécessaires pour raffiner l'information latérale). Les trames clés sont supposées reconstruites parfaitement par le décodeur pour la génération de l'information latérale par interpolation des deux trames adjacentes aux trames WZ. Dans le tableau 5.1, on présente les caractéristiques principales des séquences vidéo utilisées dans les différentes simulations. Toutes ces séquences sont extraites de [53] et constituent un contenu assez varié pour suivre le comportement des divers algorithmes d'une séquence à une autre.

5.5.2 Reconstruction standard et reconstruction optimale

La reconstruction standard utilisée par Aaron *et al.* [26] et par Brites *et al.* [8] se résume à limiter la distorsion de reconstruction à la plage du quantificateur. Cette reconstruction comporte 3 cas repré-

Nom de la séquence	Foreman	Carphone	Salesman	Mother and daugther
Échantillon				
# total de trames	400	382	449	961
# de trames utilisées	101	101	101	101
# de trames WZ	50	50	50	50
# de trames clés	51	51	51	51
Résolution spatiale	QCIF (176×144)	QCIF (176×144)	QCIF (176×144)	QCIF (176×144)
Résolution temporelle	30 trames par seconde	30 trames par seconde	30 trames par seconde	30 trames par seconde

Tableau 5.1: *Caractéristiques des séquences vidéo utilisées dans les simulations .*

sentés à la figure 5.10 et dans l'équation (5.5).

$$\hat{x} = \begin{cases} z_i, & \text{si} \quad y < z_i & \text{(cas 3)} \\ y, & \text{si} \quad y \in [z_i,\, z_{i+1}[& \text{(cas 1)} \\ z_{i+1}, & \text{si} \quad y \geq z_{i+1} & \text{(cas 2)} \end{cases} \tag{5.5}$$

où $\hat{x}$ désigne la valeur estimée, i désigne l'indice de quantification de l'intervalle délimité par les bornes z_i et z_{i+1} de la valeur envoyée x et y désigne l'information latérale.

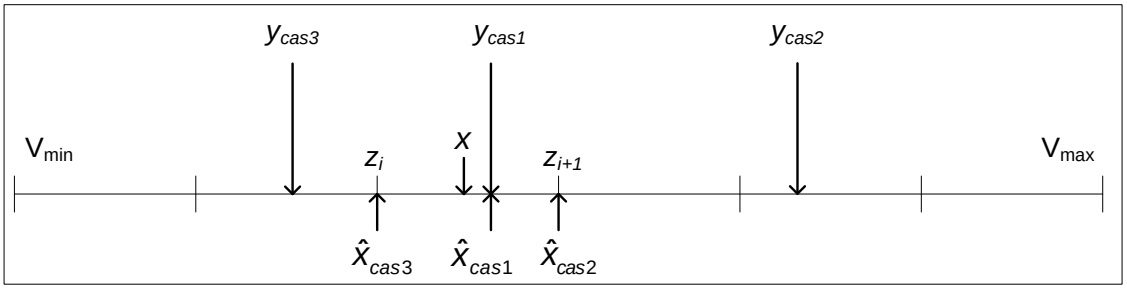

FIGURE 5.10: *Reconstruction standard.*

La reconstruction standard est une reconstruction sous optimale. La méthode de la reconstruction optimale a été présentée dans [32] : on détaille ici la démarche analytique effectuée pour la définir. La valeur de $\hat{x}$ doit être évaluée comme la valeur moyenne de la variable aléatoire x (inconnue au niveau du décodeur) étant donnée la connaissance de l'indice de quantification i, qui constitue le résultat du décodage turbo, et la connaissance de l'information latérale y. Ainsi on a $\hat{x}_{opt} = E\{x | x \in [z_i,\, z_{i+1}[\,, y\}$.

En se basant sur la figure 5.11, la valeur de $\hat{x}_{opt}$ telle que définie, correspond graphiquement à la position suivant laquelle les deux surfaces hachurées délimitées par la Laplacienne et séparées par $\hat{x}_{opt}$, sont égales. De façon intuitive, on peut s'apercevoir de l'efficacité de cette méthode en prenant le cas où l'information latérale se trouve éloignée de l'intervalle défini par l'indice de quantification. Dans ce cas précis, l'information latérale est très bruitée et on ne peut pas se baser sur cette valeur lors de la décision de $\hat{x}$. Seule la valeur de l'indice de quantification contribuera efficacement à la décision : le choix est porté à être le centre de l'intervalle de quantification qui lui est associé. Graphiquement on peut constater que lorsque l'information latérale est bruitée, les deux zones hachurées sont égales pour une valeur de $\hat{x}$ tendant vers le centre. La reconstruction standard aurait plutôt évalué $\hat{x}$ à la borne la plus rapprochée de l'intervalle de quantification, c'est-à-dire z_i ou z_{i+1}.

Afin d'être utilisée dans l'implémentation, la valeur de $\hat{x}_{opt}$ doit être évaluée analytiquement sous une forme simple. Pour déterminer $\hat{x}_{opt}$ on effectue d'abord le calcul dans le domaine discret :

$$\begin{aligned}
\hat{x}_{opt} &= E\{x | x \in [z_i,\, z_{i+1}[\,, y\} \\
&= \sum_x x \cdot \Pr\{X = x | X \in [z_i,\, z_{i+1}[\,, y\} \\
&= \sum_x \frac{x \cdot \Pr\{X = x, X \in [z_i,\, z_{i+1}[\,| y\}}{\Pr\{X \in [z_i,\, z_{i+1}[\,| y\}} \\
&= \frac{\sum_x x \cdot \Pr\{X = x, X \in [z_i,\, z_{i+1}[\,| y\}}{P\{X \in [z_i,\, z_{i+1}[\,| y\}} \\
&= \frac{\sum\limits_{x \in [z_i,\, z_{i+1}[} x \cdot \Pr\{X = x | y\}}{P\{X \in [z_i,\, z_{i+1}[\,| y\}}
\end{aligned} \tag{5.6}$$

Par la suite, on considère une distibution continue de X à l'aide de sa fonction de densité de probabilité (pdf) : $\Pr(X = x) = f(x)\,dx$. On obtient alors :

$$\begin{aligned}
\hat{x}_{opt} &= E\{x | x \in [z_i,\, z_{i+1}[\,, y\} \\
&= \frac{\sum\limits_{x \in [z_i,\, z_{i+1}[} x \cdot \Pr\{X = x | y\}}{P\{X \in [z_i,\, z_{i+1}[\,| y\}} \\
&= \frac{\int_{z_i}^{z_{i+1}} x \cdot f_{XY}(x|y) dx}{\int_{z_i}^{z_{i+1}} f_{XY}(x|y) dx}
\end{aligned} \tag{5.7}$$

où la fonction de densité de probabilité conditionnelle $f_{XY}(x|y) = \frac{\alpha}{2} e^{-\alpha|x-y|}$ représente le modèle de corrélation Laplacienne. Ainsi, on peut déterminer l'expression analytique du numérateur $N = \frac{\alpha}{2} \int_{z_i}^{z_{i+1}} x e^{-\alpha|x-y|} dx$ et du dénominateur $D = \frac{\alpha}{2} \int_{z_i}^{z_{i+1}} e^{-\alpha|x-y|} dx$ de l'expression (5.7).

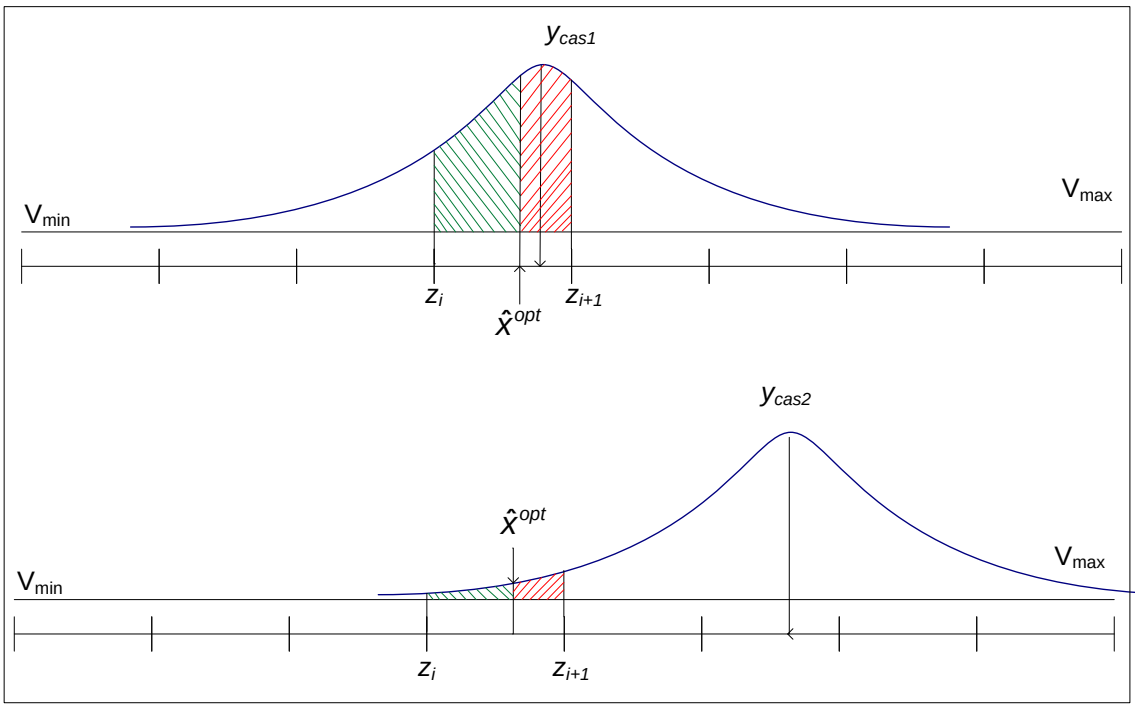

FIGURE 5.11: *Principe de la reconstruction optimale : (a) l'information latérale est dans l'intervalle de quantification* $[z_i,\ z_{i+1}[$ *; (b) l'information latérale est à l'extérieur de l'intervalle de quantification.*

Considérons le cas où $y \geq z_{i+1}$ pour le calcul de N (i.e., cas 2).

$$
\begin{aligned}
N(y \geq z_{i+1}) &= \frac{\alpha}{2} \int_{z_i}^{z_{i+1}} x e^{-\alpha|x-y|} dx \\
&= \frac{\alpha}{2} \int_{z_i}^{z_{i+1}} x e^{\alpha(x-y)} dx \\
&= \frac{\alpha}{2} e^{-\alpha y} \int_{z_i}^{z_{i+1}} x e^{\alpha x} dx \\
&= \frac{\alpha}{2} e^{-\alpha y} \left\{ \left(\frac{x e^{\alpha x}}{\alpha} \right) \Big|_{z_i}^{z_{i+1}} - \int_{z_i}^{z_{i+1}} \frac{e^{\alpha x}}{\alpha} dx \right\} \\
&= \frac{\alpha}{2} e^{-\alpha y} \left\{ \frac{z_{i+1} e^{\alpha z_{i+1}} - z_i e^{\alpha z_i}}{\alpha} - \frac{e^{\alpha z_{i+1}} - e^{\alpha z_i}}{\alpha^2} \right\} \\
&= \frac{e^{-\alpha y}}{2} \left\{ \frac{(1-\alpha z_i) e^{\alpha z_i}}{\alpha} - \frac{(1-\alpha z_{i+1}) e^{\alpha z_{i+1}}}{\alpha} \right\} \\
&= \frac{(1-\alpha z_i) e^{-\alpha(y-z_i)}}{2\alpha} - \frac{(1-\alpha z_{i+1}) e^{-\alpha(y-z_{i+1})}}{2\alpha}
\end{aligned}
\tag{5.8}
$$

D'une façon similaire, on obtient pour $y < z_i$ (cas 3) :

$$N(y < z_i) = \frac{(1+\alpha z_i)\, e^{\alpha(y-z_i)}}{2\alpha} - \frac{(1+\alpha z_{i+1})\, e^{\alpha(y-z_{i+1})}}{2\alpha} \tag{5.9}$$

Enfin, lorsque $y \in [z_i,\ z_{i+1}[$, (i.e., cas 1), N peut être décomposé de la façon suivante :

$$\begin{aligned} N(y \in [z_i,\ z_{i+1}[) &= \frac{\alpha}{2} \int_{z_i}^{z_{i+1}} x e^{-\alpha|x-y|} dx \\ &= \frac{\alpha}{2} \int_{z_i}^{y} x e^{-\alpha(y-x)} dx + \frac{\alpha}{2} \int_{y}^{z_{i+1}} x e^{-\alpha(x-y)} dx \end{aligned} \tag{5.10}$$

Le premier terme à la droite de l'équation (5.10) peut être calculé à partir de l'équation (5.8) en remplaçant z_{i+1} par y et le second terme à partir de l'équation (5.9) en remplaçant z_i par y. Ce qui donne le développement suivant :

$$\begin{aligned} N(y \in [z_i,\ z_{i+1}[) &= \frac{\alpha}{2} \int_{z_i}^{y} x e^{-\alpha(y-x)} dx + \frac{\alpha}{2} \int_{y}^{z_{i+1}} x e^{-\alpha(x-y)} dx \\ &= \frac{(1-\alpha z_i)\, e^{-\alpha(y-z_i)}}{2\alpha} - \frac{(1-\alpha y)}{2\alpha} + \frac{(1+\alpha y)}{2\alpha} - \frac{(1+\alpha z_{i+1})\, e^{\alpha(y-z_{i+1})}}{2\alpha} \\ &= \frac{(1-\alpha z_i+\alpha y)\, e^{-\alpha(y-z_i)}}{2\alpha} - \frac{y e^{-\alpha(y-z_i)}}{2} + y \\ &\quad - \frac{(1+\alpha z_{i+1}-\alpha y)\, e^{\alpha(y-z_{i+1})}}{2\alpha} - \frac{y e^{\alpha(y-z_{i+1})}}{2} \\ &= \frac{(1+\alpha\gamma)\, e^{-\alpha\gamma}}{2\alpha} - \frac{y e^{-\alpha\gamma}}{2} + y - \frac{(1+\alpha\delta)\, e^{-\alpha\delta}}{2\alpha} - \frac{y e^{-\alpha\delta}}{2} \\ &= \frac{(1+\alpha\gamma)\, e^{-\alpha\gamma}}{2\alpha} - \frac{(1+\alpha\delta)\, e^{-\alpha\delta}}{2\alpha} + \left(1 - \frac{e^{-\alpha\gamma}}{2} - \frac{e^{-\alpha\delta}}{2}\right) y \end{aligned} \tag{5.11}$$

avec $\gamma = y - z_i$ et $\delta = z_{i+1} - y$. Quant au terme du dénominateur D dans le cas où $y \geq z_{i+1}$, le calcul est effectué comme suit :

$$\begin{aligned} D(y \geq z_{i+1}) &= \frac{\alpha}{2} \int_{z_i}^{z_{i+1}} e^{-\alpha|x-y|} dx \\ &= \frac{\alpha}{2} \int_{z_i}^{z_{i+1}} e^{\alpha(x-y)} dx \\ &= \frac{\alpha}{2} e^{-\alpha y} \int_{z_i}^{z_{i+1}} e^{\alpha x} dx \\ &= \frac{\alpha}{2} e^{-\alpha y} \left(\frac{e^{\alpha x}}{\alpha}\right)\Big|_{z_i}^{z_{i+1}} \\ &= \frac{e^{\alpha(z_{i+1}-y)} - e^{-\alpha(y-z_i)}}{2} \end{aligned} \tag{5.12}$$

De manière équivalente, on obtient :

$$D(y < z_i) = -\frac{e^{-\alpha(z_{i+1}-y)} - e^{\alpha(y-z_i)}}{2} \tag{5.13}$$

La décomposition effectuée dans (5.10) pour le calcul de $N\,(y \in [z_i,\ z_{i+1}[)$ est également utilisée dans le calcul de $D\,(y \in [z_i,\ z_{i+1}[)$:

$$D\,(y \in [z_i,\ z_{i+1}[) = \frac{\alpha}{2} \int_{z_i}^{z_{i+1}} e^{-\alpha|x-y|} dx = \frac{\alpha}{2} \int_{z_i}^{y} e^{-\alpha(y-x)} dx + \frac{\alpha}{2} \int_{y}^{z_{i+1}} e^{-\alpha(x-y)} dx \tag{5.14}$$

Le premier terme à la droite de l'équation (5.14) peut être calculé à partir de (5.12) en remplaçant z_{i+1} par y et le second terme à partir de (5.13) en remplaçant z_i par y. Ce qui donne le développement suivant :

$$\begin{aligned} D\,(y \in [z_i,\ z_{i+1}[) &= \frac{\alpha}{2} \int_{z_i}^{y} e^{-\alpha(y-x)} dx + \frac{\alpha}{2} \int_{y}^{z_{i+1}} e^{-\alpha(x-y)} dx \\ &= \frac{1 - e^{-\alpha(y-z_i)}}{2} - \frac{e^{-\alpha(z_{i+1}-y)} - 1}{2} \\ &= 1 - \frac{e^{-\alpha(y-z_i)}}{2} - \frac{e^{-\alpha(z_{i+1}-y)}}{2} \\ &= 1 - \frac{e^{-\alpha\gamma}}{2} - \frac{e^{-\alpha\delta}}{2} \end{aligned} \tag{5.15}$$

où encore une fois on $\gamma = y - z_i$ et $\delta = z_{i+1} - y$.

Récapitulons ici les expressions des numérateur et dénominateur dérivées plus haut :

$$\begin{aligned} N\,(y \geq z_{i+1}) &= \frac{(1-\alpha z_i)\, e^{-\alpha(y-z_i)}}{2\alpha} - \frac{(1-\alpha z_{i+1})\, e^{-\alpha(y-z_{i+1})}}{2\alpha} \\ N\,(y < z_i) &= \frac{(1+\alpha z_i)\, e^{\alpha(y-z_i)}}{2\alpha} - \frac{(1+\alpha z_{i+1})\, e^{\alpha(y-z_{i+1})}}{2\alpha} \\ N\,(y \in [z_i,\ z_{i+1}[) &= \frac{(1+\alpha\gamma)\, e^{-\alpha\gamma}}{2\alpha} - \frac{(1+\alpha\delta)\, e^{-\alpha\delta}}{2\alpha} + \left(1 - \frac{e^{-\alpha\gamma}}{2} - \frac{e^{-\alpha\delta}}{2}\right) y \\ D\,(y \geq z_{i+1}) &= \frac{e^{\alpha(z_{i+1}-y)} - e^{-\alpha(y-z_i)}}{2} \\ D\,(y < z_i) &= -\frac{e^{-\alpha(z_{i+1}-y)} - e^{\alpha(y-z_i)}}{2} \\ D\,(y \in [z_i,\ z_{i+1}[) &= 1 - \frac{e^{-\alpha\gamma}}{2} - \frac{e^{-\alpha\delta}}{2} \end{aligned} \tag{5.16}$$

Le calcul de $\hat{x}_{opt}\,(y < z_i)$ (cas 3) suit alors le développement suivant :

$$
\begin{aligned}
\hat{x}_{opt}\left(y<z_i\right) &= \frac{\frac{(1+\alpha z_i)e^{\alpha(y-z_i)}}{2\alpha}-\frac{(1+\alpha z_{i+1})e^{\alpha\left(y-z_{i+1}\right)}}{2\alpha}}{-\frac{e^{-\alpha\left(z_{i+1}-y\right)}-e^{\alpha(y-z_i)}}{2}} \\
&= \frac{e^{\alpha(y-z_i)}-e^{-\alpha(z_{i+1}-y)}}{\alpha\left(e^{\alpha(y-z_i)}-e^{-\alpha(z_{i+1}-y)}\right)}+\frac{\alpha z_i e^{\alpha(y-z_i)}-\alpha z_{i+1}e^{\alpha(y-z_{i+1})}}{\alpha\left(e^{\alpha(y-z_i)}-e^{-\alpha(z_{i+1}-y)}\right)} \\
&= \frac{1}{\alpha}+\frac{z_i e^{-\alpha z_i}-z_i e^{-\alpha z_{i+1}}-z_{i+1}e^{-\alpha z_{i+1}}+z_i e^{-\alpha z_{i+1}}}{\left(e^{-\alpha z_i}-e^{-\alpha z_{i+1}}\right)} \\
&= \frac{1}{\alpha}+\frac{z_i e^{-\alpha z_i}-z_i e^{-\alpha z_{i+1}}}{\left(e^{-\alpha z_i}-e^{-\alpha z_{i+1}}\right)}-\frac{z_{i+1}e^{-\alpha z_{i+1}}-z_i e^{-\alpha z_{i+1}}}{\left(e^{-\alpha z_i}-e^{-\alpha z_{i+1}}\right)} \\
&= \frac{1}{\alpha}+z_i+\frac{z_{i+1}-z_i}{\left(1-e^{\alpha(z_{i+1}-z_i)}\right)} \qquad (5.17)
\end{aligned}
$$

Suite à un développement similaire on obtient, pour $y \geq z_{i+1}$ (cas 2) :

$$
\hat{x}_{opt}\left(y \geq z_{i+1}\right) = \frac{1}{\alpha}+z_{i+1}-\frac{z_{i+1}-z_i}{\left(1-e^{\alpha(z_{i+1}-z_i)}\right)} \qquad (5.18)
$$

Finalement pour $y \in [z_i,\, z_{i+1}[$ (cas 1) on a :

$$
\begin{aligned}
\hat{x}_{opt}\left(y \in [z_i,\, z_{i+1}[\right) &= \frac{\frac{(1+\alpha\gamma)e^{-\alpha\gamma}}{2\alpha}-\frac{(1+\alpha\delta)e^{-\alpha\delta}}{2\alpha}+\left(1-\frac{e^{-\alpha\gamma}}{2}-\frac{e^{-\alpha\delta}}{2}\right)y}{1-\frac{e^{-\alpha\gamma}}{2}-\frac{e^{-\alpha\delta}}{2}} \\
&= y+\frac{\left(\frac{1}{\alpha}+\gamma\right)e^{-\alpha\gamma}-\left(\frac{1}{\alpha}+\delta\right)e^{-\alpha\delta}}{2-e^{-\alpha\gamma}-e^{-\alpha\delta}} \qquad (5.19)
\end{aligned}
$$

La reconstruction optimale est implémentée en utilisant l'équation suivante qui permet d'éviter le calcul des rapports d'intégrales :

$$
\hat{x}_{opt} = \begin{cases}
z_i+\frac{1}{\alpha}+\frac{z_{i+1}-z_i}{\left(1-e^{\alpha(z_{i+1}-z_i)}\right)}, & \text{pour } y<z_i \\
y+\frac{\left(\frac{1}{\alpha}+\gamma\right)e^{-\alpha\gamma}-\left(\frac{1}{\alpha}+\delta\right)e^{-\alpha\delta}}{2-e^{-\alpha\gamma}-e^{-\alpha\delta}} & \text{pour } y \in [z_i,\, z_{i+1}[\\
z_{i+1}+\frac{1}{\alpha}-\frac{z_{i+1}-z_i}{\left(1-e^{\alpha(z_{i+1}-z_i)}\right)}, & \text{pour } y \geq z_{i+1}
\end{cases} \qquad (5.20)
$$

avec $\gamma = y-z_i$, $\delta = z_{i+1}-y$ et α le paramètre de la distribution Laplacienne modélisant le canal virtuel.

5.5.3 Mesure de qualité d'image objective et subjective

MSE et PSNR

Pour évaluer les performances de l'encodeur/décodeur DVC, une mesure de la qualité de la séquence vidéo reconstruite doit être utilisée. Dans un système de communication, l'erreur quadratique moyenne (*MSE*) est une mesure objective très utilisée. Dans le cas d'une image le PSNR ((*Peak signal-to-noise ratio*) serait plus appropriée dans la mesure où elle prend en considération la profondeur des

pixels et elle est exprimée suivant une échelle logarithmique. Le PSNR peut constituer alors une approximation de la perception du système visuel humain mais il reste toujours une mesure objective et liée directement à l'erreur quadratique moyenne.

SSIM

Dans une tentative de fournir une mesure subjective prenant en considération l'appréciation humaine de l'image reconstruite, le SSIM (Structural SIMilarity) a été proposé dans [10]. Cette mesure permet de contrecarrer l'inconsistance du MSE et du PSNR avec la perception visuelle humaine. En effet, au lieu de mesurer la qualité de chaque pixel indépendamment de son voisinage, le SSIM s'intéresse à la similarité des structures en considérant plusieurs fenêtres de taille $N \times N$. La valeur du SSIM calculée pour une fenêtre de la trame originale x et une fenêtre de la trame reconstruite y est donnée par :

$$SSIM\,(x,y) = \frac{(2\mu_x\mu_y + c_1)\,(2\sigma_{xy} + c_2)}{\left(\mu_x^2 + \mu_y^2 + c_1\right)\left(\sigma_x^2 + \sigma_y^2 + c_2\right)} \tag{5.21}$$

où μ_x et σ_x^2 sont la moyenne et la variance de la fenêtre x, μ_y et σ_y^2 sont la moyenne et la variance de la fenêtre y et σ_{xy} est la covariance de x et y. $c_1 = (k_1 L)^2$, $c_2 = (k_2 L)^2$ sont deux variables pour stabiliser numériquement le processus de la division lorsque le dénominateur devient très petit : L est la valeur maximale des pixels, soit 255 pour des images à 8 bits. Finalement on prend, par défaut, $k_1 = 0.01$ et $k_2 = 0.03$ avec une taille de fenêtre de $N \times N = 8 \times 8$.

Pour mettre en évidence la pertinence du SSIM par rapport au MSE (ou encore le PSNR) les auteurs de [10] présentent une série d'images ayant une valeur de MSE similaire mais de qualité très différente (voir figure 5.12). Le SSIM donne une meilleure indication subjective concernant les différentes images.

5.5.4 Gain de performance par exploitation de la corrélation au niveau des trames

Tel que discuté précédemment, la transformation sous-optimale (mais qui s'approche des performances de la transformation optimale de Karhunen-Loève) DCT 4×4 permet d'exploiter la corrélation spatiale au niveau de la trame. Ainsi les données à envoyer sont décorrélées avec une transformation permettant de supprimer la redondance dans les données pour assurer une meilleure compression. On compare, à la figure 5.13, le gain de performance obtenu suite au passage du domaine pixel au domaine transformé pour les séquences vidéos présentées précédemment : `Foreman`, `Carphone`, `Salesman` et `Mother and daughter`. L'exploitation de la corrélation au niveau des trames permet d'obtenir un gain de performance allant de 1 à 2 dB. Toutes les séquences utilisées sont téléchargées de [53]. L'implémentation de l'architecture du codeur DVC dans le domaine transformé considère les deux types de quantificateurs les plus utilisés dans la littérature et qu'on a présenté précédemment : le quantificateur de Brites *et al.* [8] et le quantificateur de Discover [9]. Pour chaque courbe du domaine transformé on considère les 8 matrices de quantifications de la figure 5.6. Quant au domaine pixel, chaque courbe comporte quatre points correspondant à un nombre de niveaux de quantification de 2, 4, 8 et 16.

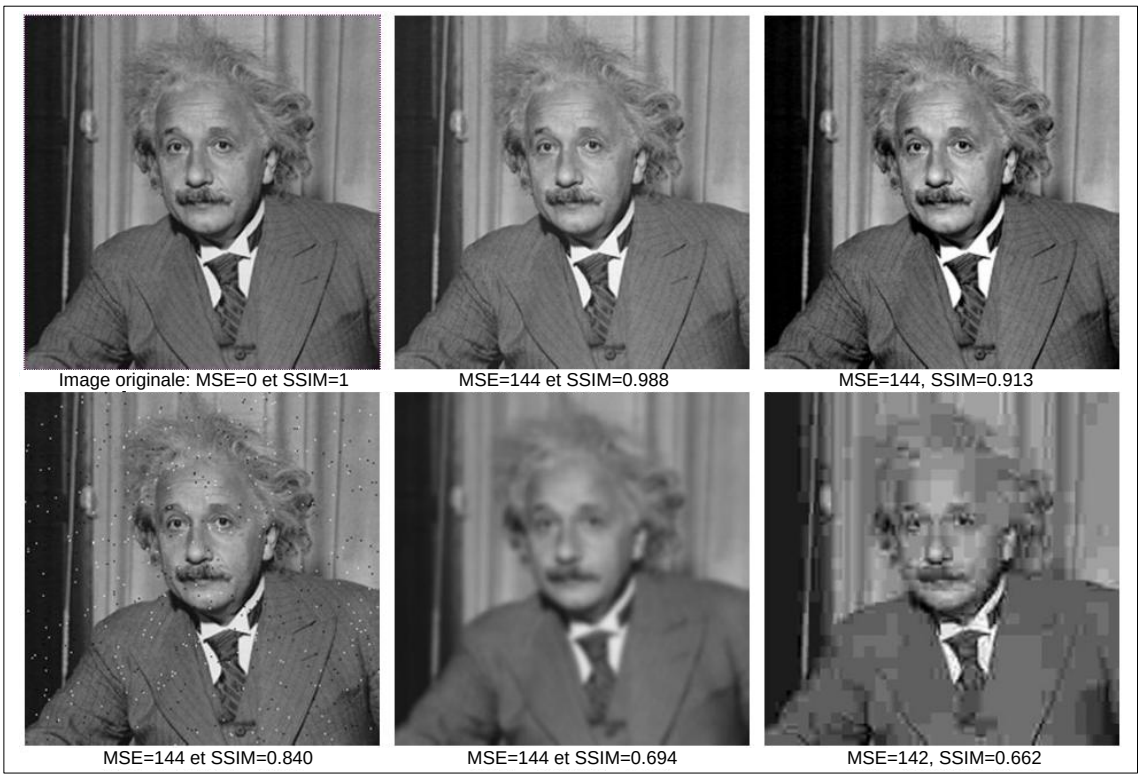

FIGURE 5.12: *Comparaison entre la pertinence du MSE et du SSIM pour l'évaluation de la qualité subjective de l'image [10].*

5.5.5 Gain de performance obtenu par la reconstruction optimale

La reconstruction optimale, comme son nom l'indique, développée et étudiée précédemment permet d'optimiser le bloc reconstruction de l'architecture DVC. À la figure 5.14, cette méthode de reconstruction est comparée à la reconstruction standard dans le domaine transformée, pour les mêmes 4 séquences vidéos. On considére la mesure de qualité d'image objective habituelle, le PSNR et la mesure subjective présentée ci-dessus, le SSIM.

On remarque bien un gain de performance qui peut dépasser 1 dB du point de vue PSNR et qui peut atteindre 5×10^{-4} du point de vue SSIM. Ce gain varie d'une séquence à une autre et il est plus apparent dans les séquences où l'information latérale est bruitée. En effet, lorsque cette dernière est peu bruitée, les symboles de quantification de l'information originale et de l'information latérale sont dans la plupart des cas les mêmes. La reconstruction standard prend alors une décision égale à l'information latérale, ce qui est une décision pertinente. En effet, on remarque à partir de la figure 5.14, que les performances de système DVC avec la séquence `Salesman` ne varient pas grandement en utilisant la reconstruction optimale vu que l'information latérale est de bonne qualité. Quant à la séquence `Foreman`, l'information latérale est bruitée et l'apport de la reconstruction optimale est plus prononcé.

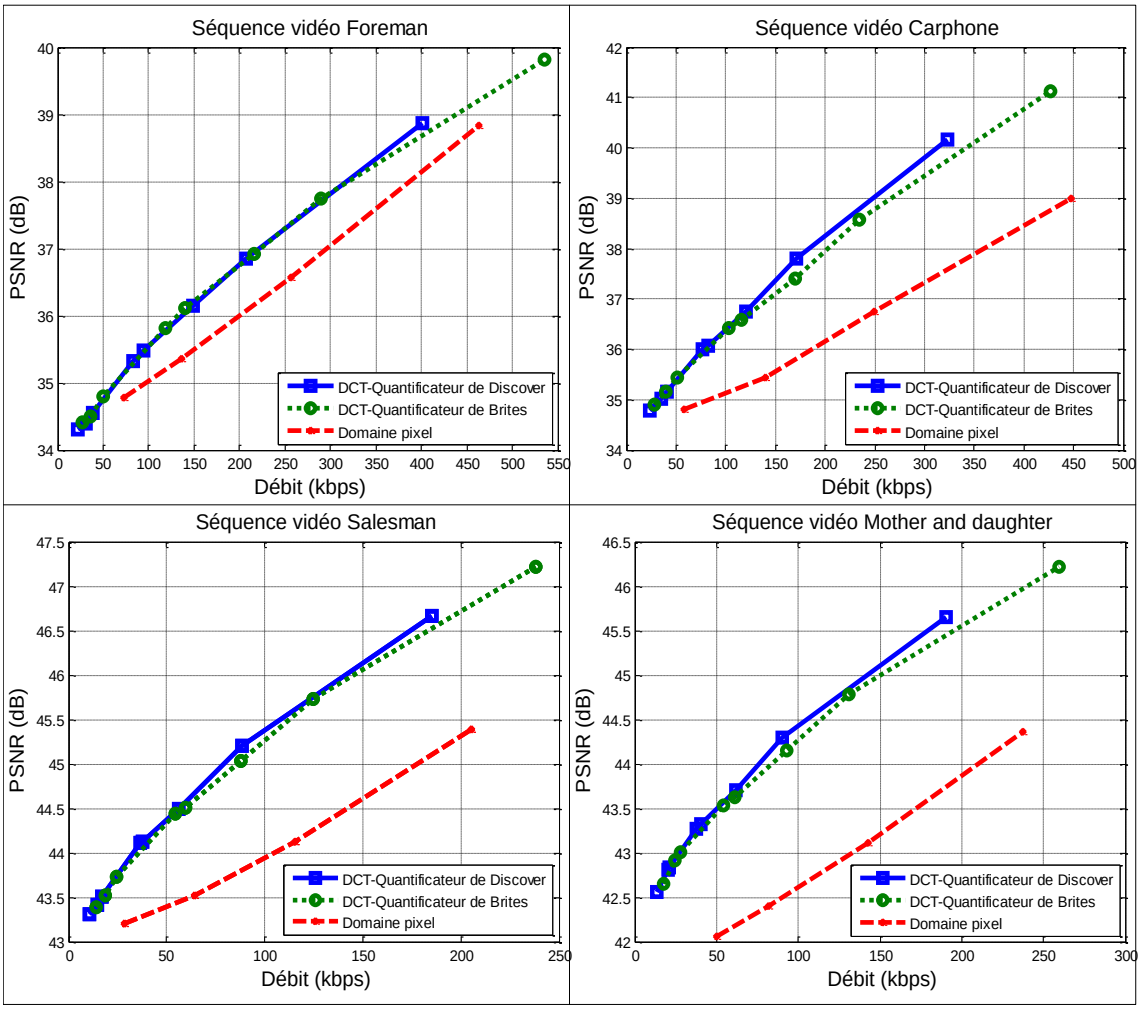

FIGURE 5.13: *Comparaison entre les performances du système DVC dans le domaine pixel et dans le domaine transformée.*

Cependant on remarque, contre toutes attentes, que pour la séquence `Mother and daugther`, la reconstruction standard donne de meilleures performances que la reconstruction optimale sauf pour la matrice de plus haut débit. Ceci s'explique par le fait que l'information latérale est de très bonne qualité et que la reconstruction standard prend la valeur de l'information latérale telle quelle si elle se trouve dans l'intervalle du symbole de quantification. Cependant la reconstruction optimale, prend une valeur après moyennage autour de l'information latérale. Il est important de noter que la reconstruction optimale, ne l'est que du point de vue étude *statistique*. Bien que cela se produise rarement, il peut y avoir des cas où la reconstruction standard dépasse en performance la reconstruction (statistiquement) optimale.

Concernant les deux mesures de qualité d'image, on remarque bien une concordance entre la qualité objective et subjective. C'est-à-dire que si, du point de vue du PSNR, les performances d'une courbe est supérieure aux performances d'une autre courbe, le SSIM ne contredira pas ce fait. La figure

5.12, présentée ci-dessus et qui permettait de mettre en évidence la pertinence du SSIM par rapport au PSNR (et le MSE) en affichant une série d'images de qualités différentes, est en réalité un choix bien précis de répartition de l'erreur de telle sorte qu'il soit visuellement gênant. Dans l'architecture distribuée de transmission vidéo, l'erreur du canal affecte aléatoirement l'image et le fait de tomber dans le cas d'impertinence du PSNR comme dans la figure 5.12 est improbable. Donc le choix de la considération du SSIM comme mesure de qualité d'image dans les simulations effectuées vient consolider les résultats obtenu par le PSNR et non pas les mettre à l'épreuve.

5.5.6 Gain de performance obtenu par l'utilisation d'un quantificateur adaptatif

L'idée du quantificateur proposé et présenté ci-dessus provient de la constatation que le quantificateur de Discover ayant un intervalle double autour de 0 pour les coefficients AC donne généralement de meilleures performances que le quantificateur scalaire uniforme utilisée par les auteurs de [8]. En effet ceci suggère que le changement de la disposition des intervalles de quantification, de façon autre qu'uniforme, pourrait induire des améliorations. La formulation théorique d'un quantificateur optimal pour les coefficients AC de l'architecture DVC dans le domaine transformée n'est pas vraiment un problème bien défini comme c'est le cas pour le quantificateur de Max-Lloyd. Ce dernier ne considère pas la présence d'une information latérale corrélée avec l'information originale et le compromis débit-distorsion inhérent. Le design du quantificateur de Max-Lloyd se base uniquement sur la distribution des données originales et vise simplement à diminuer la valeur moyenne de distorsion sans une seconde contrainte de minimisation du débit.

Connaissant la complexité de la spécification d'un quantificateur optimal pour les coefficients AC, plusieurs simulations ont été réalisées dans le but de comparer avec les quantificateurs utilisés dans la littérature. Suite à cette approche empirique, on a retenu finalement le quantificateur décrit précédemment et on présente à la figure 5.15 une comparaison entre le quantificateur proposé et le quantificateur de Discover en considérant le PSNR et le SSIM comme mesures de distorsion des trames WZ reconstruites. On note un gain de performance allant jusqu'à 0.75 dB pour la séquence `Foreman` et jusqu'à 0.5 dB pour la séquence `Carphone`. Quant aux séquences `Salesman` et `Mother and daugther`, le quantificateur proposé et le quantificateur de Discover donnent des performances similaires. Le gain du quantificateur proposé se manifeste le plus lorsque l'information latérale est bruitée. L'objectif de ce quantificateur est de corriger les erreurs qui ont le plus d'impact dans la réduction de la distorsion et d'éviter de corriger les erreurs qui n'ont que peu d'impact sur la distorsion évitant ainsi d'augmenter le débit.

5.6 Conclusion

Dans ce chapitre, on s'est intéressé à l'étude de l'architecture distribuée d'un encodeur et d'un décodeur vidéo dans le domaine pixel et dans le domaine transformé avec la DCT 4×4. Cette transformation simple et pratique permet de s'approcher des performances de la transformation optimale de

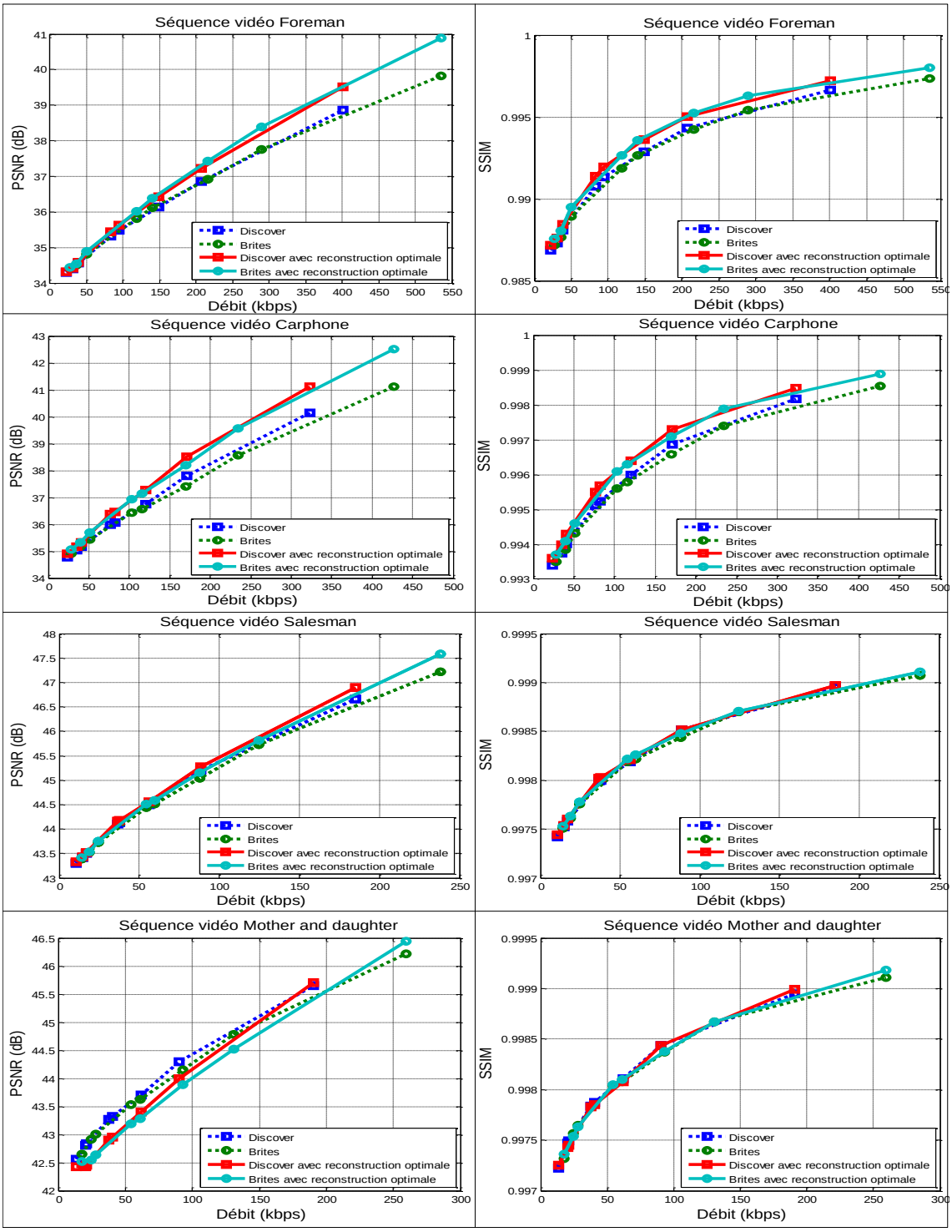

FIGURE 5.14: *Comparaison entre les performances du système DVC dans le domaine transformée avec une reconstruction standard et une reconstruction optimale.*

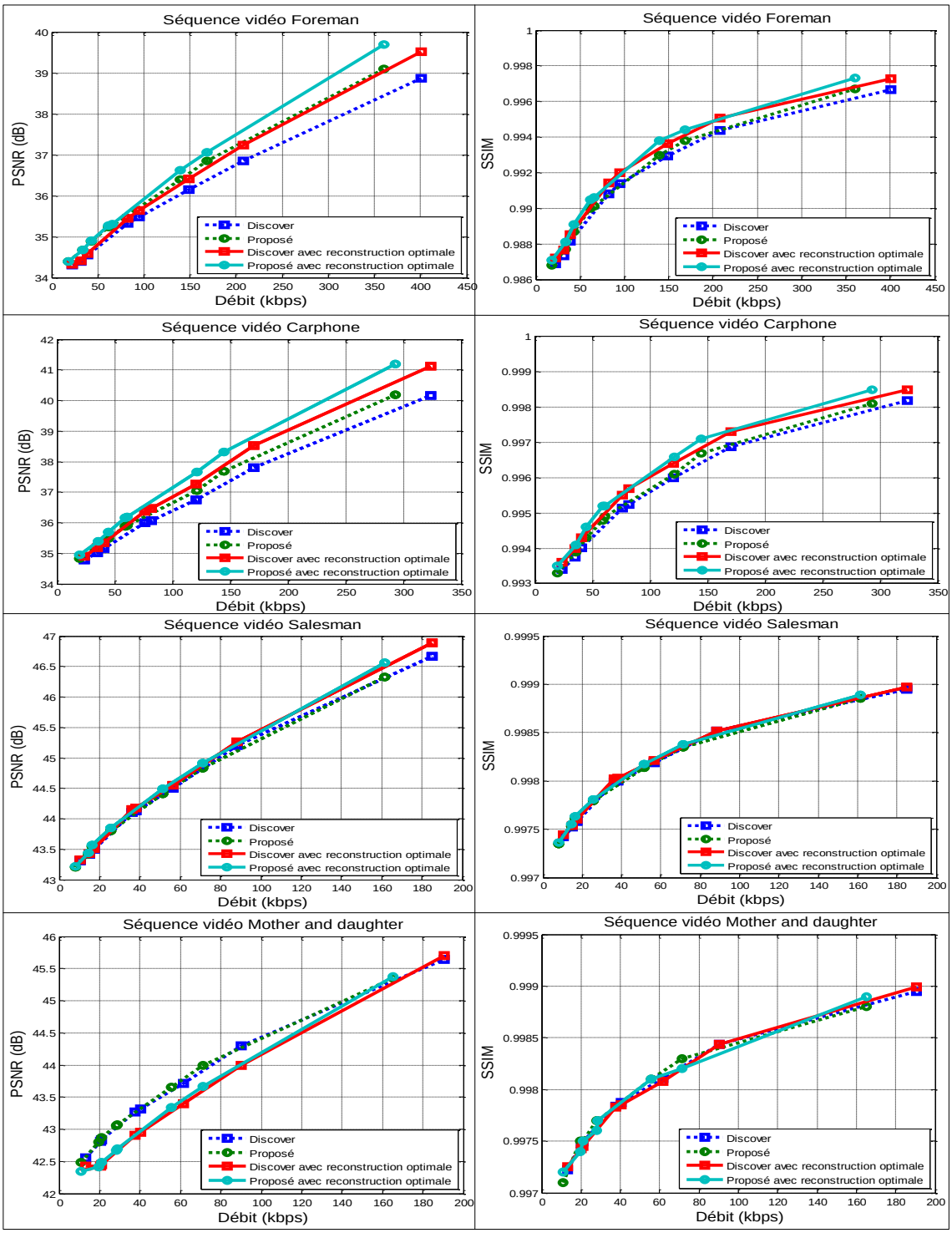

FIGURE 5.15: *Comparaison entre le quantificateur proposé et le quantificateur de DISCOVER dans le domaine transformée du système DVC avec une reconstruction standard et une reconstruction optimale*

Karhunen-Loève pour la décorrélation des données originales. La comparaison des résultats de simulation du domaine pixel et du domaine transformé montre bien le gain considérable de performance qu'on peut atteindre en exploitant la corrélation au niveau de l'image.

La transformation DCT 4×4 génère des coefficients de la bande DC distribués sur la plage dynamique de 0 à 1023 et des coefficients de 15 bandes AC qui sont concentrés autour de 0 et avec une plage dynamique qui varie d'une bande à l'autre. Diverses techniques de quantification pour le domaine transformé de l'architecture DVC, se trouvant dans la littérature ont été étudiées et implémentées. D'autres techniques de quantification ont été essayées dans le but de déterminer comment le design du quantificateur des coefficients AC devrait être conçu et si un gain de performance pouvait être atteint. Ces simulations ont conduit à la spécification d'un quantificateur permettant d'avoir une certaine amélioration par rapport à ceux de la littérature.

Dans les simulations présentées, on a également utilisé une autre mesure de la qualité d'images, le SSIM afin d'évaluer la qualité subjective au-delà de la qualité objective donnée par le PSNR. D'après les simulations effectuées, ces deux mesures concordent bien et permettent de se compléter dans l'évaluation de la qualité de la séquence vidéo reconstruite.

Le bloc de reconstruction a aussi été étudié en présentant la formulation de la technique de reconstruction statistiquement optimale qui se distingue de la reconstruction standard par le fait qu'elle prend en considération la corrélation des données originales avec l'information latérale. Les simulations présentent des gains de performance dépassant 1 dB du point de vue PSNR par l'utilisation de la reconstruction optimale par rapport à la reconstruction standard. Les divers blocs d'un codec vidéo Wyner-Ziv dans le domaine transformé ont été étudiés et implémentés. Le bloc d'interpolation permettant d'améliorer la qualité de l'information latérale sera étudié et traité dans le prochain chapitre. L'information latérale est l'élément clé dans l'architecture du codage vidéo distribué et son amélioration permet non seulement d'augmenter la compression en réduisant le nombre de bits de parité nécessaires à la convergence du processus de décodage turbo mais aussi de réduire la distorsion de la vidéo reconstruite.

Chapitre 6

Interpolation bidirectionnelle dans le codage vidéo distribué

6.1 Introduction

Le codage vidéo distribué est un axe de recherche qui regroupe à la fois les techniques des codes correcteurs d'erreurs et les techniques de compression vidéo. Dans les chapitres précédents les divers blocs de l'architecture de codage vidéo distribué ont été étudiés et implémentés en faisant intervenir principalement des outils et des connaissances de la théorie de l'information et de codage canal distribué. Le bloc d'interpolation pour la génération de l'information latérale à partir des trames adjacentes se limite dans les simulations effectuées précédemment à une simple opération de moyennage. Pour améliorer considérablement les performances du système DVC, l'amélioration du bloc d'interpolation est une approche efficace et indispensable faisant intervenir des techniques et des connaissances sur le codage et la compression vidéo.

Dans l'architecture de codage vidéo distribué, l'information latérale est générée en exploitant les trames avoisinantes déjà décodées pour approximer l'information originale. Ensuite, un codeur WZ permet de corriger les erreurs de cette approximation par la transmission des bits de parité générés par un codeur turbo. Lorsque l'interpolation des trames disponibles au niveau du décodeur pour la génération de l'information latérale fournit un résultat de bonne qualité, un faible nombre de bits de parité sera nécessaire pour assurer la convergence de l'opération de décodage turbo entrainant ainsi une réduction du débit WZ. De plus, le bloc de reconstruction fait appel à la même information latérale que le décodeur turbo et si celle-ci est de bonne qualité, la distorsion des trames reconstruites sera faible. L'information latérale intervient, alors, en deux temps dans un système DVC et sa qualité influe directement à la réduction du débit et de la distorsion. Pour cette raison, la communauté de codage vidéo distribué a prêté une attention particulière au module d'interpolation qui n'a pas cessé d'évoluer depuis l'émergence de ce nouveau paradigme. Ce module adapte les techniques d'estimation et de compensation de mouvement des standards de codage vidéo conventionnels (i.e. MPEG 4, H.26x)

pour les exploiter dans le contexte du codage vidéo distribué. La particularité de ce contexte réside dans le fait que les vecteurs de mouvement sont calculés sans connaitre la trame originale mais plutôt à partir des deux trames avoisinantes.

Dans ce chapitre, on commence par détailler et implémenter l'un des algorithmes d'interpolation bidirectionnelle du système DVC les plus utilisés dans la littérature [13]. Les divers blocs de cet interpolateur seront étudiés pour présenter une meilleure compréhension de son fonctionnement et pour préparer la phase d'implémentation. Par la suite, on présente la méthode de calcul de flux optique de Lucas-Kanade multi-résolution (avec pyramide) qui permet de déterminer le flux de mouvement entre deux images données. Après quoi, on implémente cette technique d'interpolation en l'adaptant au contexte distribué de la compression vidéo. Les performances de ces deux méthodes et d'autres méthodes moins performantes sont ensuite comparées pour une panoplie de séquences pour couvrir les diverses natures de mouvement qu'une vidéo peut présenter. On incorpore, au final, le bloc d'interpolation utilisant les techniques bidirectionnelles dans l'architecture du codage vidéo distribué. Ainsi on pourra constater l'apport d'une estimation précise de l'information latérale et sur le débit et sur la distorsion.

6.2 Estimation et compensation bidirectionnelle de mouvement avec lissage spatial

On présente dans cette section, l'algorithme d'interpolation présenté par Ascenso *et al.* dans [13] : celui-ci comporte les différentes étapes décrites à la figure 6.1. Cette figure récapitule 3 techniques d'estimation de mouvement : l'estimation de mouvement en avant, l'estimation de mouvement bidirectionnelle et l'estimation de mouvement bidirectionnelle avec lissage spatial. Chacune de ces trois techniques va être étudiée dans une section à part. Mais on présente, tout d'abord, le fonctionnement des blocs qui se répètent dans les divers algorithmes : le bloc de filtrage passe bas et le bloc de compensation de mouvement bidirectionnelle.

6.2.1 Filtrage passe bas

Le but du bloc d'interpolation est la détermination de la vraie nature du mouvement entre les trames $i-1$ et $i+1$ afin de déterminer une prédiction adéquate de la trame i. Les deux trames avoisinantes sont alors filtrées par un filtre passe bas permettant ainsi de réduire le bruit et d'assurer une fiabilité de l'estimation des vecteurs de mouvement. Le filtre utilisé est un filtre gaussien ayant les coefficients suivants :

$$\mathrm{LPF} = \begin{pmatrix} 0.0751 & 0.1238 & 0.0751 \\ 0.1238 & 0.2042 & 0.1238 \\ 0.0751 & 0.1238 & 0.0751 \end{pmatrix} \tag{6.1}$$

Le résultat du filtrage, comme on le voit à la figure 6.2, introduit un certain degré de flou en atténuant l'impact des arêtes. Ainsi le bloc d'estimation de mouvement parvient à détecter le mouvement réel

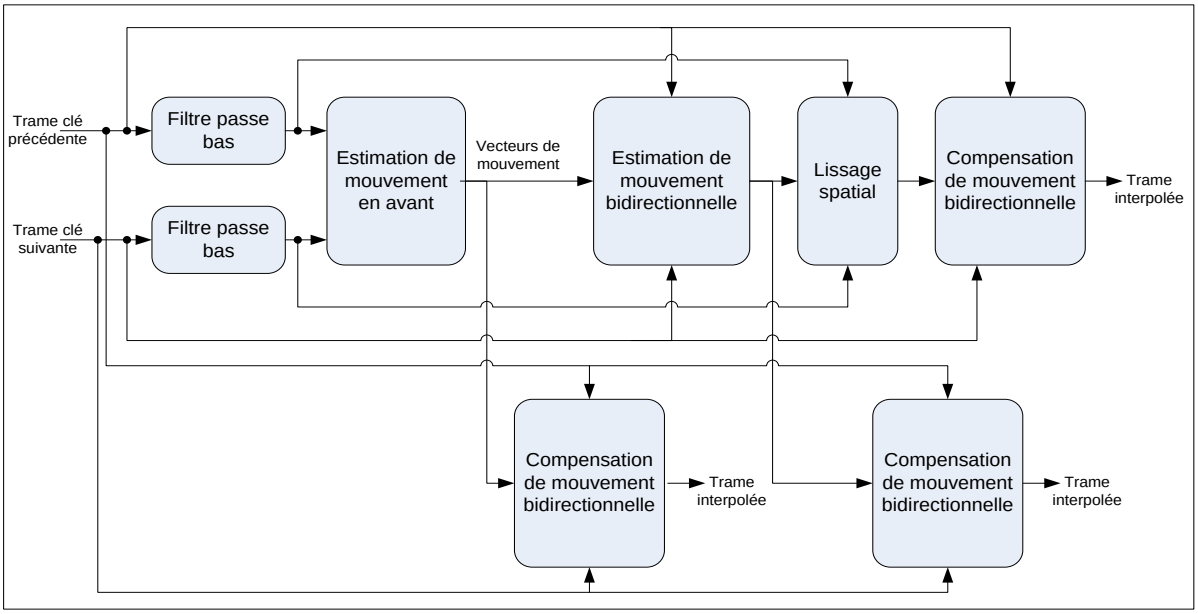

FIGURE 6.1: *Composantes du bloc d'interpolation bidirectionnelle utilisé dans un système DVC.*

et les vecteurs de mouvements obtenus seront moins bruités.

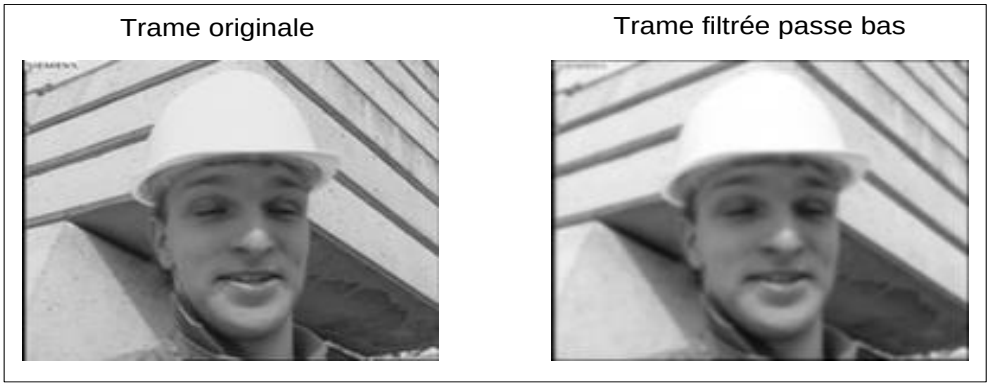

FIGURE 6.2: *Filtrage passe bas d'une trame.*

6.2.2 Compensation de mouvement bidirectionnelle

On retrouve l'opération de compensation de mouvement à la fin de tous les algorithmes d'estimation de mouvement comme on le voit à la figure 6.1. Ce bloc prend les vecteurs de mouvement bidirectionnels obtenus pour reconstituer la trame interpolée à partir des deux trames adjacentes. Cette opération suppose que le mouvement entre la trame précédente et la trame interpolée est égale au mouvement entre cette dernière et la trame suivante. Chaque bloc de la trame interpolée prend alors la moyenne

entre le bloc de la trame précédente obtenu par le vecteur de mouvement vers l'arrière et le bloc de la trame suivante obtenu par le vecteur de mouvement vers l'avant.

6.3 Description de l'algorithme d'estimation de mouvement vers l'avant

On détaille maintenant les diverses techniques d'interpolation de la figure 6.1. Ces techniques se complètent et les vecteurs de mouvement obtenus par un bloc donné servent comme entrées au bloc ultérieur. La première technique d'interpolation de mouvement suggérée par la figure 6.1 consiste en l'estimation de mouvement vers l'avant. Le fonctionnement de cet algorithme peut être subdivisé en deux étapes :

6.3.1 Étape 1 : Estimation de mouvement basée sur des blocs rigides

Dans cette étape schématisée à la figure 6.3, on considère un découpage en blocs de taille 8×8 pixels de la trame suivante $i+1$. Pour chaque bloc de la trame $i+1$, on recherche le bloc de la trame $i-1$ qui lui ressemble le plus dans une zone de recherche de ± 8 pixels. La ressemblance des blocs peut être évaluée à l'aide de l'erreur quadratique moyenne (MSE) ou la différence moyenne absolue (MAD). Le pas de recherche est de 2 pixels. L'augmentation de la valeur de ce pas permet de réduire la complexité de cet algorithme. Cette étape permet de déterminer les vecteurs de mouvement entre la trame $i-1$ et la trame $i+1$ et le bloc de la trame interpolée est supposé à mi-chemin entre les deux.

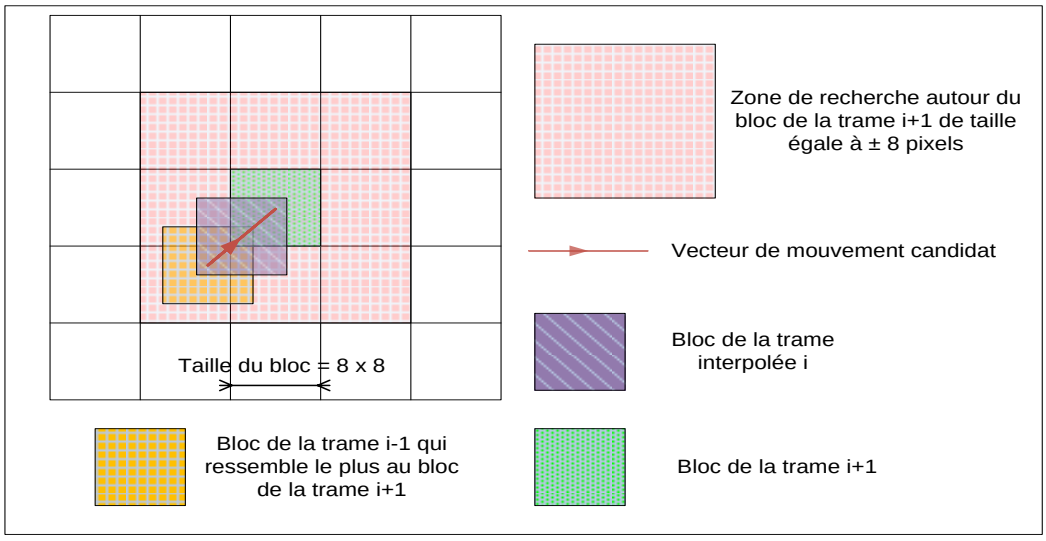

FIGURE 6.3: *Première étape : estimation du mouvement entre la trame i-1 et la trame i+1.*

Cette étape permet de déterminer les vecteurs de mouvement entre la trame $i-1$ et la trame $i+1$ dans le but de déterminer le bloc de la trame interpolée supposée à mi-chemin entre les deux. Cependant, les vecteurs de mouvements trouvés n'interceptent pas les blocs de la trame interpolée au niveau de

leurs centres. Ceci peut entrainer l'apparition de zones non couvertes et de zones de chevauchement comme le montre la figure 6.4.

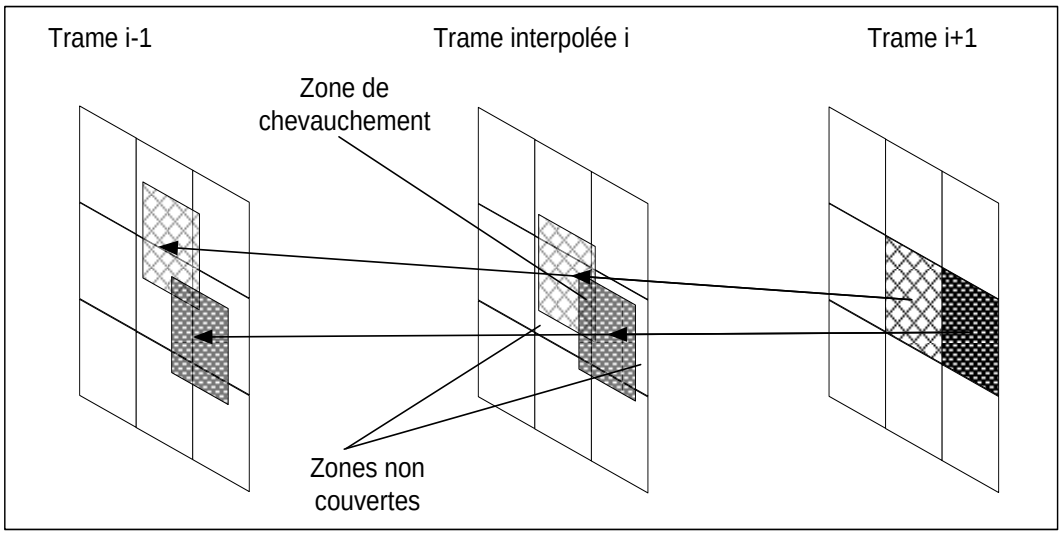

FIGURE 6.4: *Apparition de zones non couvertes et de zones de chevauchement suite à l'algorithme d'estimation de mouvement basée sur des blocs rigides.*

6.3.2 Étape 2 : translation du vecteur de mouvement

Pour remédier au problème rencontré dans la première étape, l'idée consiste à considérer que le vecteur de mouvement déterminé par une estimation basée sur des blocs rigides est un vecteur candidat pour estimer le mouvement au niveau de la trame interpolée. Ce vecteur de mouvement candidat est alors translaté pour coïncider avec le centre du bloc de la trame interpolée évitant, ainsi les chevauchements et les zones non couvertes dans celle-ci comme on le voit à la figure 6.5. A ce niveau on peut procéder à une compensation de mouvement bidirectionnelle pour déterminer la trame interpolée à partir des vecteurs de mouvements obtenus. Ces vecteurs de mouvement peuvent être encore améliorés par une estimation de mouvement bidirectionnelle.

6.3.3 Implémentation de l'estimation de mouvement avant

Dans l'implémentation des divers algorithmes d'estimation de mouvement que l'on effectuera dans cette section, on considère la trame WZ numéro 7 de la séquence `Foreman` avec le sigle *Siemens*, téléchargée à partir de la référence [49], pour afficher les vecteurs de mouvements obtenus et le résultat de l'interpolation. Concernant cet algorithme, on présente le résultat des diverses étapes de simulations à la figure 6.6. En premier lieu, on retrouve les vecteurs de mouvement obtenus suite à la première étape et qui représente le mouvement entre la trame numéro 13 et la trame numéro 15 entourant la trame numéro 14 qui correspond à la trame WZ numéro 7. Ces vecteurs de mouvements, tel discuté précédemment, n'interceptent pas la trame interpolée au centre de ses blocs. Pour cette raison, ils sont,

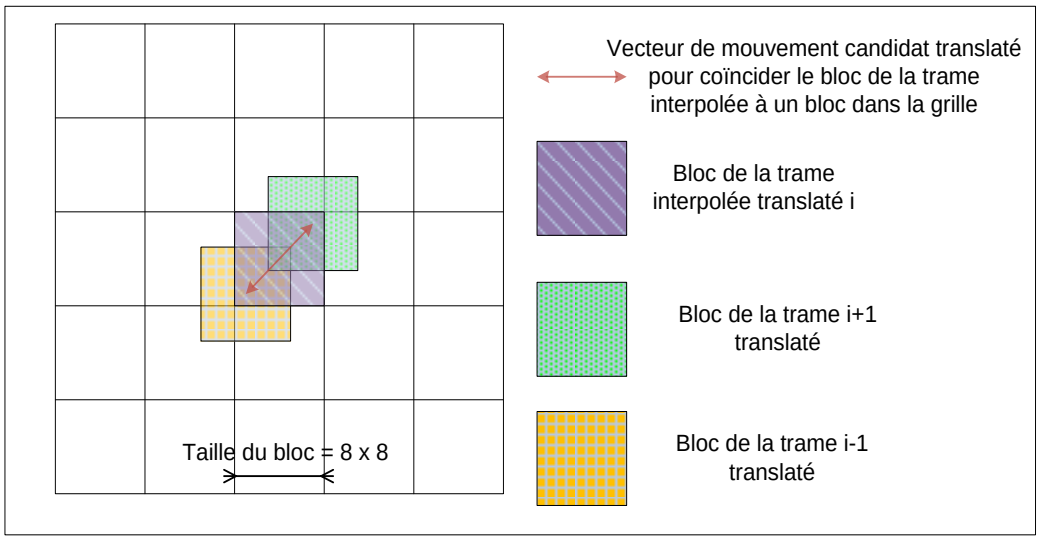

FIGURE 6.5: *Deuxième étape : translation du vecteur de mouvement obtenu à la première étape ainsi que les 3 blocs pour coïncider le bloc d'interpolation au centre d'un bloc de la grille.*

en second lieu, translatés au niveau de la seconde étape pour pouvoir effectuer une compensation de mouvement sans chevauchement ni zones non couvertes. Le résultat final de l'interpolation par l'algorithme d'estimation de mouvement vers l'avant pour la trame WZ numéro 7 de la séquence `Foreman` conduit à un PSNR de 28.36 dB.

6.4 Estimation de mouvement bidirectionnelle

L'algorithme décrit dans la section précédente peut être amélioré en plaçant le bloc d'estimation de mouvement bidirectionnelle comme le montre la figure 6.1. Ce bloc permet d'améliorer la qualité des vecteurs de mouvement bidirectionnels en effectuant une translation dans une zone de recherche correspondant à $\pm$ 2 pixels horizontalement et verticalement. Lors de cette recherche, le vecteur de mouvement est maintenu, interceptant le bloc de la trame d'interpolation à son centre. Ainsi, on peut déterminer un bloc de la trame précédente et un bloc de la trame suivante avec un maximum de corrélation tout en évitant les zones de chevauchement et les zones non couvertes. On schématise le fonctionnement de cet algorithme à la figure 6.7.

6.4.1 Implémentation de l'estimation de mouvement bidirectionnelle

L'utilisation du bloc d'estimation de mouvement bidirectionnelle, permet d'améliorer la qualité des vecteurs de mouvement comme le montre la figure 6.8. On remarque que dans les régions où il y a le mouvement du visage, l'estimation bidirectionnelle donne des vecteurs de mouvements différents de ceux obtenus par l'algorithme de la section précédente. Ainsi, le résultat final d'interpolation bidirectionnelle permet d'obtenir un PSNR de 30.44 dB soit une amélioration de 2 dB par rapport à

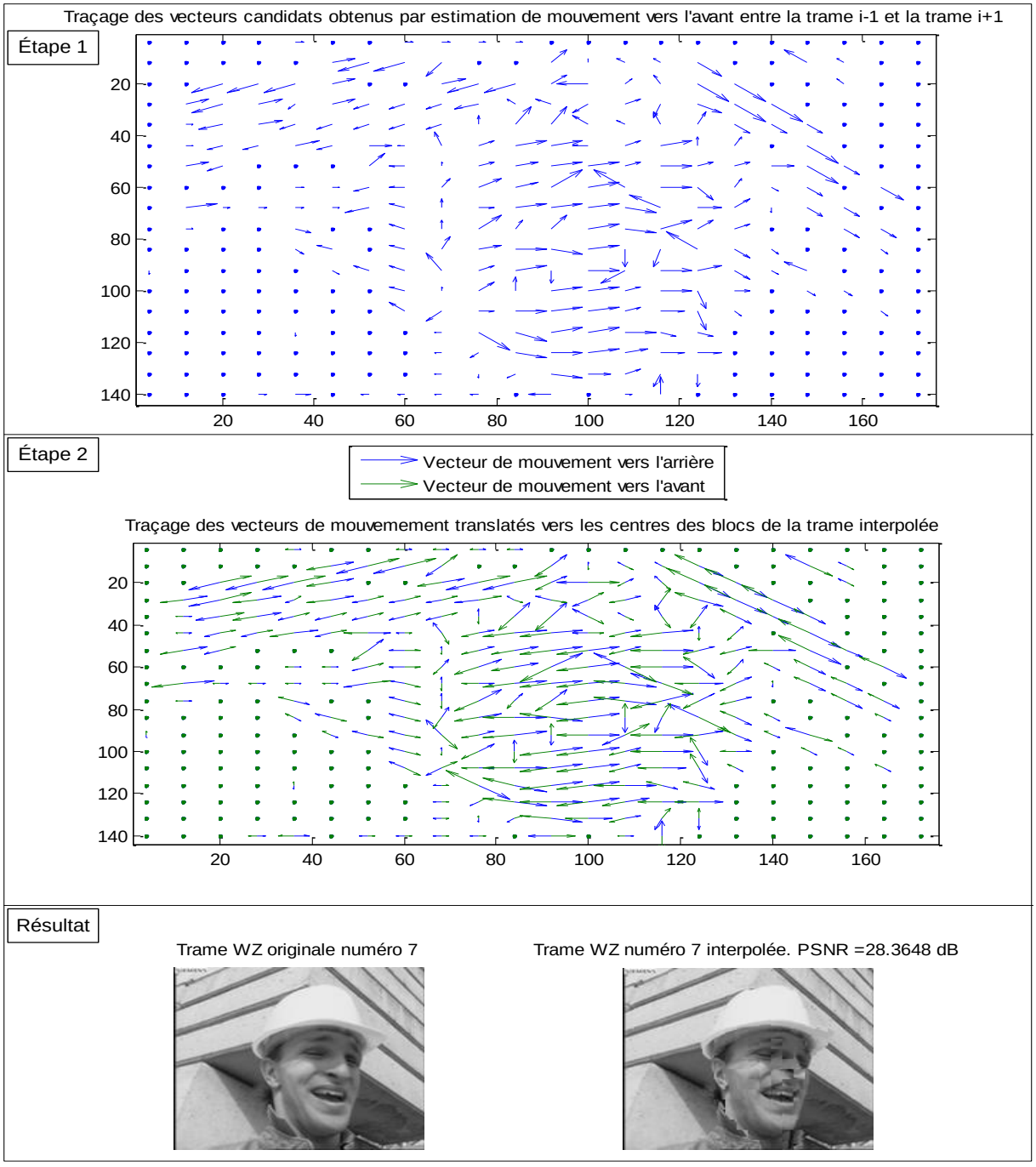

FIGURE 6.6: *Algorithme d'interpolation basé sur l'estimation de mouvement vers l'avant (PSNR = 28.36 dB). Les vecteurs de mouvement de l'étape 1 ont le même module que les vecteurs de mouvement bidirectionnels de l'étape 2. L'affichage ne prend pas en considération la valeur du module des vecteurs de façon absolue mais plutôt de façon relative au niveau de la même figure.*

précédemment.

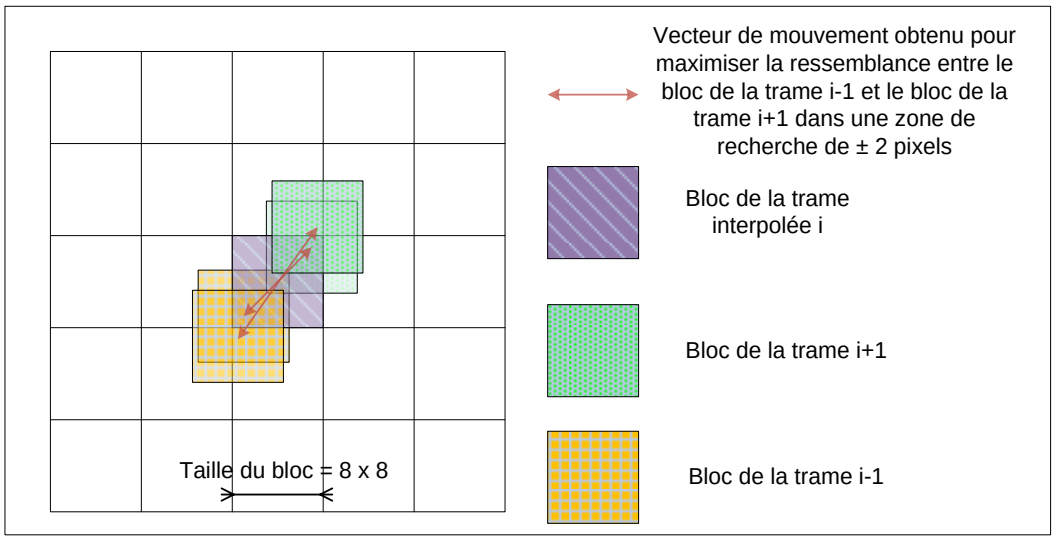

FIGURE 6.7: *Estimation de mouvement bidirectionnelle partant d'une translation de ± 2 pixels horizontalement et verticalement du vecteur de mouvement obtenu avec l'algorithme d'estimation de mouvement vers l'avant.*

6.5 Lissage spatial de mouvement

On remarque à la figure 6.8, que les vecteurs obtenus par estimation de mouvement bidirectionnelle présentent parfois une certaine incohérence spatiale. En effet, dans certaines régions, on remarque que certains vecteurs ne concordent pas avec la nature du mouvement décrit par l'ensemble des vecteurs voisins. Pour remédier à ce problème, on utilise un algorithme de lissage spatial de mouvement. Celui-ci est basé sur les filtres médians de vecteurs pondérés tel que présenté dans [54]. On présente à la figure 6.9 une schématisation du lissage spatial permettant de redresser un certain nombre de vecteurs de mouvement erronés. Ce filtre permet de prendre en considération des vecteurs de mouvement des blocs voisins dans le choix du vecteur relatif à un bloc donné. La formulation de ce filtre est comme suit :

$$x_{wvmf} = \arg\left(\min_{x_i \in V} \sum_{j=1}^{N} w_j \left\|x_i - x_j\right\|_2\right) \tag{6.2}$$

avec x_{wvmf} désigne le vecteur résultat du filtrage proposé, V désigne l'ensemble des vecteurs de mouvements du bloc en question et des blocs voisins. Ces vecteurs de mouvement sont donnés par l'algorithme d'estimation du mouvement bidirectionnelle présenté à la section précédente. N désigne le nombre d'éléments de l'ensemble V et correspond à 9 (le bloc en question et ses 8 blocs voisins). $\|.\|_2$ désigne la norme euclidienne. Le résultat du lissage spatial est pondéré par des coefficients w_j déterminés en calculant l'erreur quadratique moyenne (MSE) entre chacun des blocs voisins dans la trame précédente et dans la trame suivante :

$$w_j = \frac{MSE\left(x_c, B\right)}{MSE\left(x_j, B\right)} \tag{6.3}$$

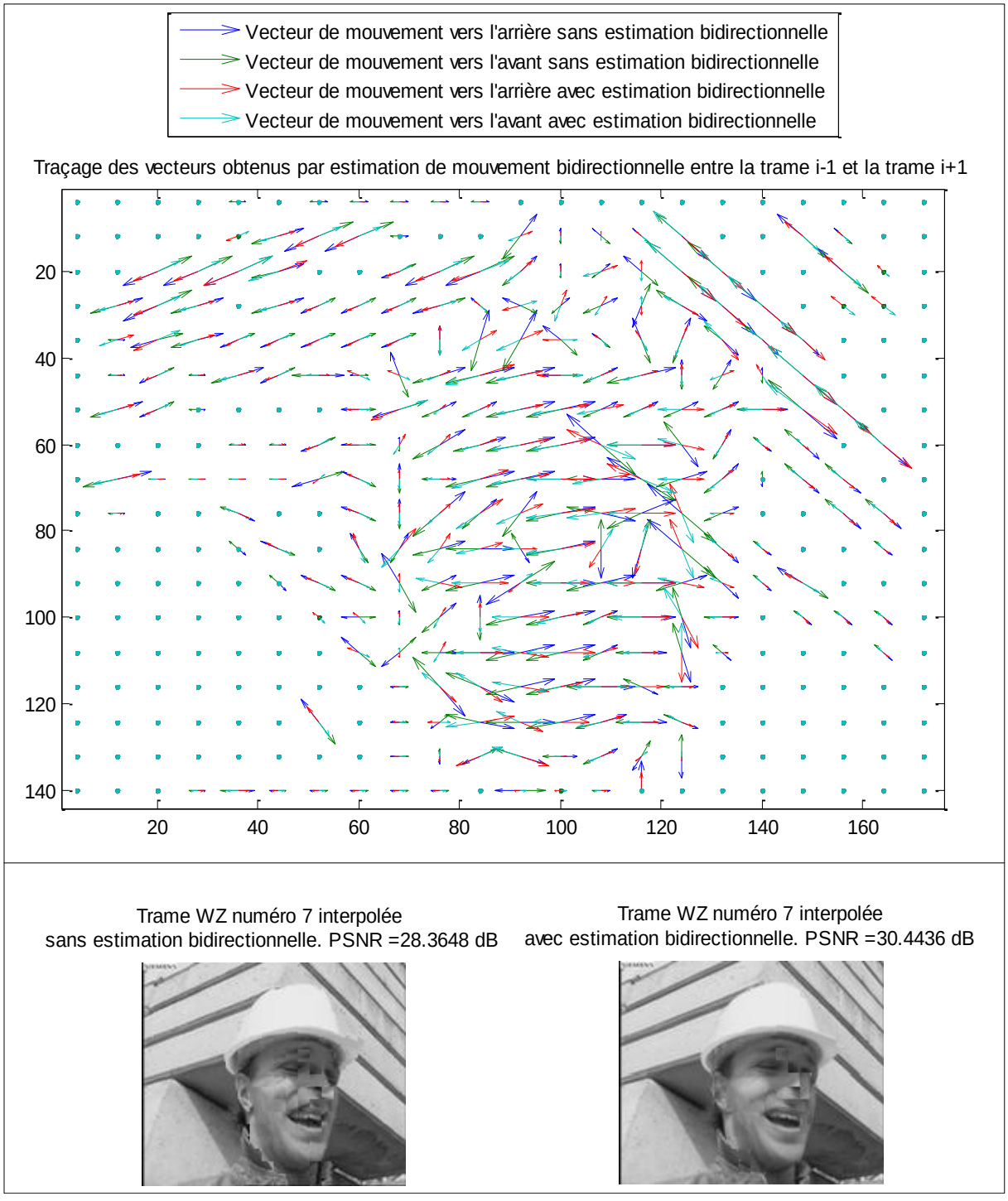

FIGURE 6.8: *Algorithme d'interpolation de mouvement bidirectionnelle (PSNR = 30.44 dB) .*

où x_c représente le vecteur candidat obtenu par l'algorithme de la section précédente pour le bloc en question.

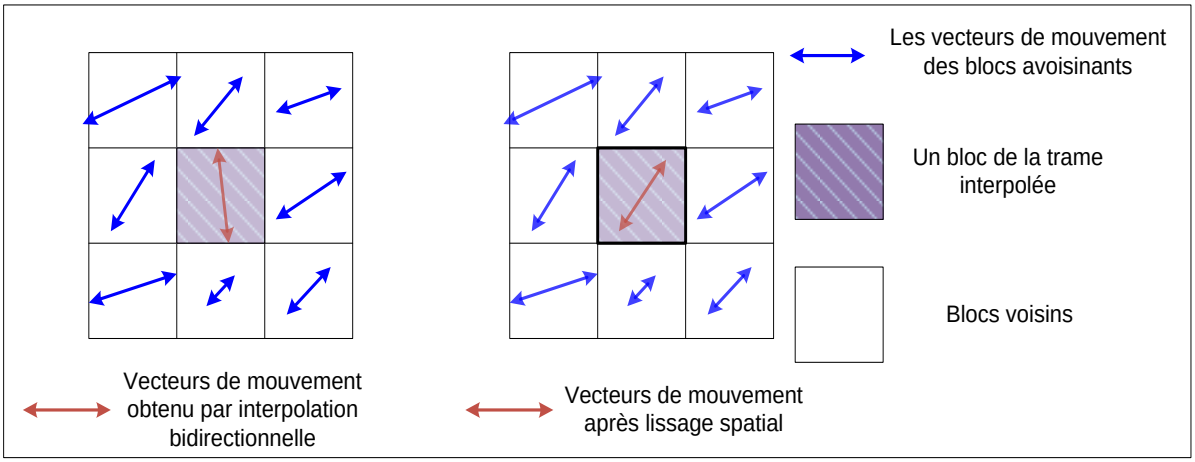

FIGURE 6.9: *Lissage spatial.*

6.5.1 Implémentation du lissage spatial de mouvement

L'utilisation de l'algorithme de lissage spatial du mouvement permet de favoriser la cohérence spatiale des vecteurs des blocs voisins. On peut observer ce fait à partir du traçage des vecteurs de mouvement avec et sans lissage à la figure 6.10. Le lissage spatial permet d'avoir un gain significatif du PSNR de 4.3 dB correspondant à un passage de 30.44 dB à 34.74 dB.

6.6 Estimation du mouvement par la méthode des flux optique de Lucas-Kanade

6.6.1 Flux optique

Le flux optique décrit la nature du mouvement des différents éléments dans une image (objets, arêtes, textures, ...) causé par le déplacement de la caméra et de la scène entre une trame à l'instant t et une trame à l'instant $t+\delta t$. Un pixel de la trame à l'instant t, de position définie par les coordonnées x et y, subit un déplacement de $(\delta x, \delta y)$ dans la trame à l'instant δt en respectant l'hypothèse d'intensité constante donnée par [55] :

$$E(x+\delta x, y+\delta y, t+\delta t) = E(x,y,t) \tag{6.4}$$

où E désigne l'intensité lumineuse du pixel en question. D'un autre côté, en supposant que le mouvement est infinitésimal, le développement en série de Taylor permet d'obtenir l'équation suivante [55] :

$$E(x+\delta x, y+\delta y, t+\delta t) = E(x,y,t) + \frac{\partial E}{\partial x}\delta x + \frac{\partial E}{\partial y}\delta y + \frac{\partial E}{\partial t}\delta t + \text{ T .O .S} \tag{6.5}$$

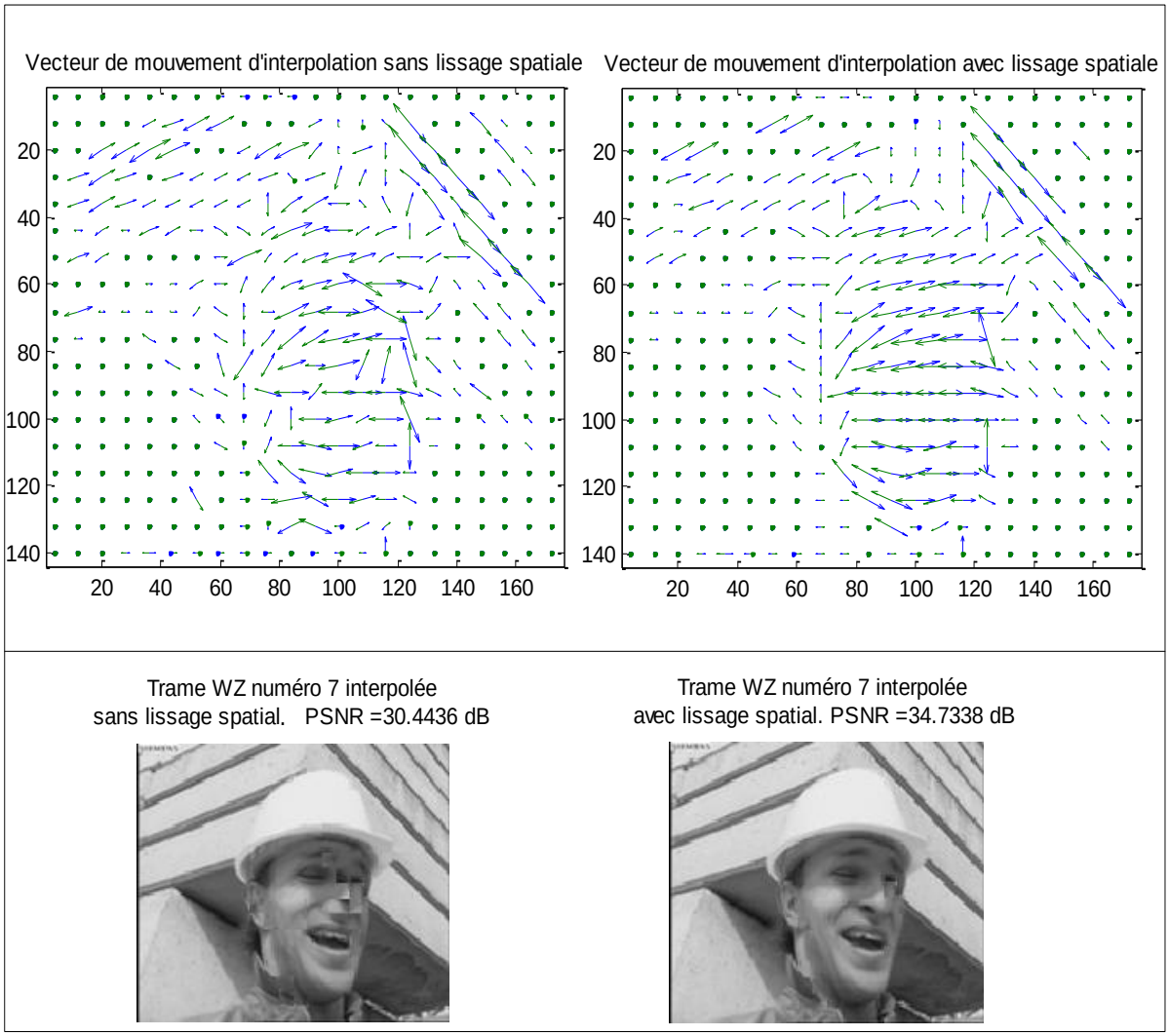

FIGURE 6.10: *Estimation de mouvement bidirectionnelle avec lissage spatial (PSNR =34.73 dB).*

où T.O.S. désigne les termes d'ordre supérieur. Ainsi, on peut déduire à partir des équations (6.4) et (6.5) que :

$$\begin{aligned} &\frac{\partial E}{\partial x}\delta x + \frac{\partial E}{\partial y}\delta y + \frac{\partial E}{\partial t}\delta t = 0 \\ &\frac{\partial E}{\partial x}\frac{\delta x}{\delta t} + \frac{\partial E}{\partial y}\frac{\delta y}{\delta t} + \frac{\partial E}{\partial t}\frac{\delta t}{\delta t} = 0 \\ &\frac{\partial E}{\partial x}V_x + \frac{\partial E}{\partial y}V_y + \frac{\partial E}{\partial t} = 0 \end{aligned} \tag{6.6}$$

où V_x et V_y désignent les composantes de vitesse suivant l'axe des x et l'axe des y respectivement. En dénotant $E_* = \frac{\partial E}{\partial *}$, on obtient l'équation de flux optique :

$$\begin{aligned} E_x V_x + E_y V_y + E_t &= 0 \\ (\nabla E)^T V + E_t &= 0 \end{aligned} \tag{6.7}$$

Ici ∇ désigne l'opérateur du gradient et $(\cdot)^T$ désigne la transposée. Cette dernière est une équation à deux inconnues, V_x et V_y, et ne peut pas être résolue sans l'ajout d'une contrainte supplémentaire [55]. L'ensemble des vecteurs V qui constituent une solution possible à cette équation est schématisé à la figure 6.11. Le problème de la non existence d'une solution bien définie est connu sous le nom du

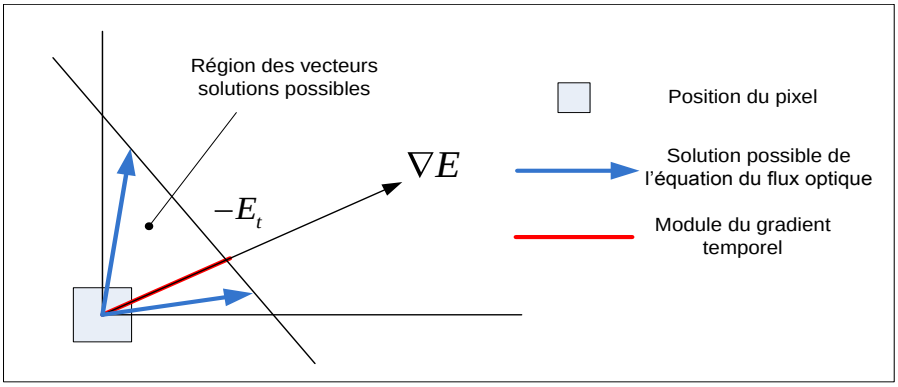

FIGURE 6.11: *Ensemble des vecteurs solution de l'équation de flux optique.*

problème d'ouverture car l'algorithme du flux optique calcule les gradients spatials horizontal E_x et vertical E_y dans le voisinage du pixel (une ouverture) en question, comme le montrent la figure 6.12 et l'équation suivante :

$$\begin{aligned} E_x &= \tfrac{N_5 - N_4}{2} \quad \text{ou} \quad E_x = \tfrac{(N_3 - N_1) + (N_5 - N_4) + (N_8 - N_6)}{2} \\ E_y &= \tfrac{N_7 - N_2}{2} \quad \text{ou} \quad E_y = \tfrac{(N_6 - N_1) + (N_7 - N_2) + (N_8 - N_3)}{2} \end{aligned} \tag{6.8}$$

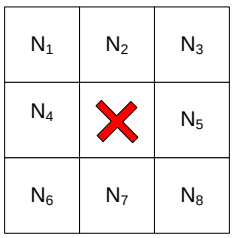

FIGURE 6.12: *Calcul du gradient spatial (voir l'équation (6.8)).*

Comme conséquence du problème d'ouverture, l'algorithme du flux optique ne pourra détecter que le mouvement parallèle à la direction du gradient spatial. Une schématisation de ce phénomène est donnée à la figure 6.13. Cet algorithme pourra cependant détecter le mouvement dans les deux directions si on a une texture, au lieu d'avoir une région uniforme lisse comme c'est le cas de la figure 6.13. Pour mieux visualiser le problème d'ouverture, le lecteur peut se référer à [56] où l'on présente des images animées qui montrent comment ce problème peut affecter la perception du mouvement. Pour résoudre

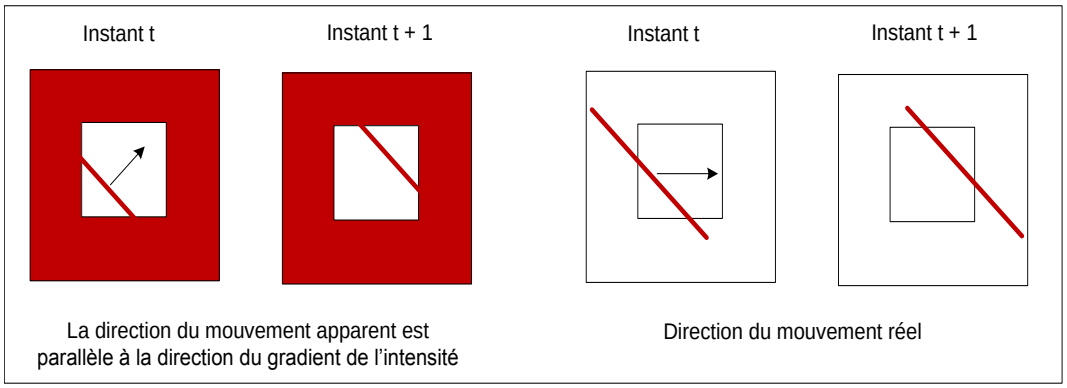

FIGURE 6.13: *Problème d'ouverture.*

l'équation du flux de mouvement, on trouve dans la littérature plusieurs méthodes qui peuvent être classées en des méthodes basées sur les blocs et des méthodes basées sur les équations différentielles. On présentera dans la section suivante la méthode différentielle de Lukas-Kanade qui est l'une des méthodes les plus connues d'estimation de mouvement entre 2 trames.

6.6.2 Méthode de Lukas-Kanade

La méthode de Lukas-Kanade rajoute une contrainte supplémentaire à l'équation du flux optique en supposant que le champ de mouvement demeure constant dans une petite région Q de l'image de taille $N \times N$. La valeur de N est typiquement égale à 5. Avec cette hypothèse, l'équation (6.7), donne les $n = N \times N = 25$ relations suivantes :

$$\begin{aligned} E_{x_1}V_x + E_{y_1}V_y &= -E_{t_1} \\ E_{x_2}V_x + E_{y_2}V_y &= -E_{t_2} \\ \vdots \qquad & \quad \vdots \quad \vdots \\ E_{x_n}V_x + E_{y_n}V_y &= -E_{t_n} \end{aligned} \tag{6.9}$$

Ainsi on obtient un système surdéterminé de 25 équations pour seulement deux inconnues :

$$\begin{bmatrix} E_{x_1} & E_{y_1} \\ E_{x_2} & E_{y_2} \\ \vdots & \vdots \\ E_{x_n} & E_{y_n} \end{bmatrix} \begin{bmatrix} V_x \\ V_y \end{bmatrix} = \begin{bmatrix} -E_{t_1} \\ -E_{t_2} \\ \vdots \\ -E_{t_n} \end{bmatrix} \quad \Rightarrow \quad A\vec{v} = -b \tag{6.10}$$

Pour résoudre ce système d'équations surdéterminé, on utilise la méthode des moindres carrés qui est une approche standard pour déterminer une solution approximative des systèmes surdéterminés [57]. La solution obtenue est alors donnée par [55] :

$$\begin{aligned}\vec{v}^* &= \arg\min_V \sum_{p\in Q} \left[E_{x_p}V_x + E_{y_p}V_y + E_{t_p}\right]^2 \\ \vec{v}^* &= \arg\min_V \sum_{p\in Q} \left[\left(\nabla E\left(p\right)\right)^T V + E_t\left(p\right)\right]^2\end{aligned} \tag{6.11}$$

où Q désigne le voisinage autour du pixel où le mouvement est constant. Tel que mentionné ci haut Q correspond à une région de taille 5×5. Pour déterminer le vecteur de mouvement permettant de satisfaire cette minimisation, on effectue la dérivée par rapport à V. On obtient, après développement la solution suivante [55] :

$$\vec{v}^* = \left(A^T A\right)^{-1} A^T b \tag{6.12}$$

où la matrice A et le vecteur b ont été déjà définis dans l'équation (6.10).

6.6.3 Implémentation de la méthode de Lukas-Kanade

On implémente maintenant la méthode d'estimation du flux optique de Lukas-Kanade pour déterminer les vecteurs de mouvement entre deux trames présentées à la figure 6.14 de la séquence vidéo `Bus`. Les

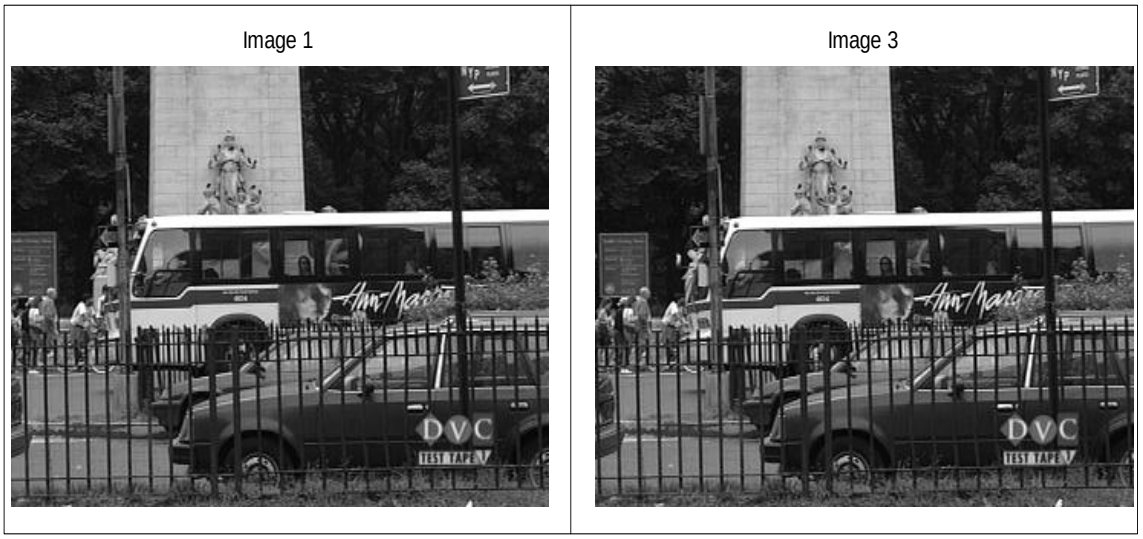

FIGURE 6.14: *Images utilisées dans l'implémentation de l'algorithme de Lukas-Kanade.*

vecteurs de mouvement entre l'image 1 et l'image 3 de la séquence `Bus` obtenus par l'algorithme de Lukas-Kanade sont présentés à la figure 6.15. On encercle dans cette figure les vecteurs de mouvement qui semblent être bruités et qui semblent ne pas concorder avec la nature du mouvement dans le voisinage. Par la suite, on effectue la compensation de mouvement pour estimer l'image 2 à partir de

l'image 1 et on trace, à la figure 6.16, la différence entre l'estimation et l'image originale. On encercle la région qui présente certaines erreurs d'interpolation et qui coïncide avec l'emplacement des vecteurs de mouvement bruités encerclés précédemment. Pour une estimation de mouvement moins sensible au bruit, on introduit dans la section suivante, l'approche multi-résolution de l'algorithme d'estimation du flux optique de Lukas-Kanade.

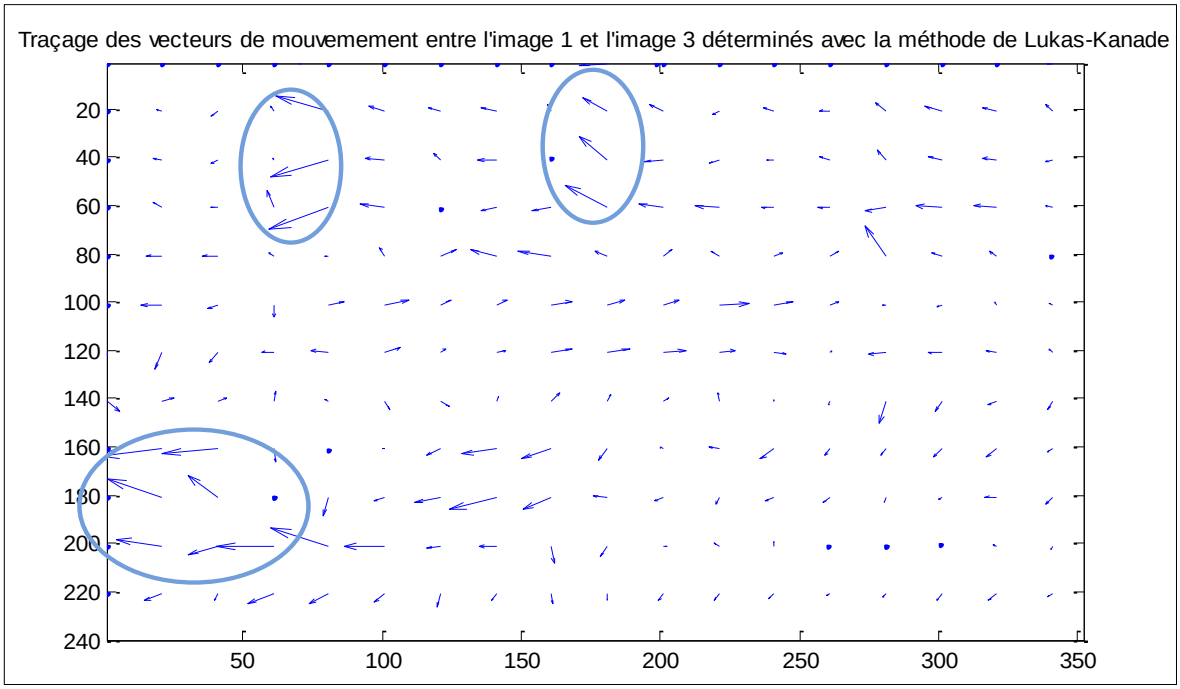

FIGURE 6.15: *Estimation du flux de mouvement avec l'algorithme de Lukas-Kanade.*

6.6.4 Implémentation de la méthode pyramidale de Lukas-Kanade

L'approche multi-résolution de l'algorithme de Lukas-Kanade consiste à calculer les vecteurs de mouvement progressivement en commençant par des trames filtrées et réduites comme le montre la figure 6.17. Ainsi la détermination des vecteurs de mouvement est de plus en plus précise et le résultat final obtenu est moins sensible au bruit. On présente à la figure 6.18 les vecteurs de mouvement obtenus par l'approche multi-résolution à 3 niveaux tout en les comparant aux vecteurs obtenus à partir de l'approche à simple résolution. On remarque bien que sans multi-résolution, l'algorithme capte le bruit qui est généralement causé par l'apparition ou la disparition (occlusion) d'objets dans la trame d'une image à l'autre. Avec l'approche multi-résolution, plus robuste face au bruit, on parvient à mieux capter la vraie nature du mouvement. Le résultat de la compensation de mouvement de l'approche multi-résolution permet d'atteindre une erreur quadratique moyenne de 285 (voir la figure 6.19). On rappelle que l'erreur quadratique moyenne de l'approche sans multi-résolution obtenu précédemment est de 1012.

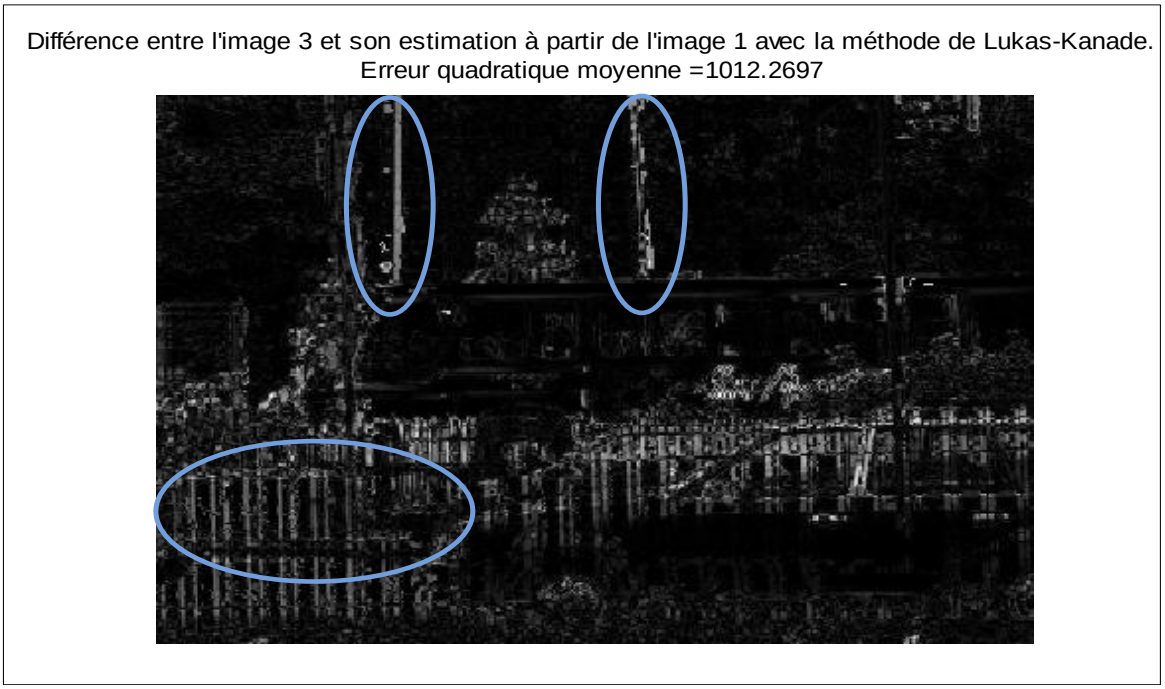

FIGURE 6.16: *Erreurs d'interpolation par l'algorithme de Lukas-Kanade (EQM=1012.27).*

6.6.5 Adaptation de l'algorithme de Lukas-Kanade avec pyramide au contexte distribué

L'algorithme de l'estimation et de la compensation de mouvement par la méthode des flux optique vise à déterminer des vecteurs de mouvement entre deux images données. Dans le contexte de l'architecture de codage vidéo distribué, l'interpolation vise plutôt à déterminer une approximation de la trame à l'instant t à partir des deux trames adjacentes aux instant $t-1$ et $t+1$. Pour adapter l'algorithme multi-résolution de Lukas-Kanade, on effectue une estimation pyramidale des vecteurs de mouvement pour interpoler la trame $t+1$ à partir de la trame $t-1$. Les vecteurs de mouvement obtenus sont ensuite divisés par 2, vu que la trame t est supposée à mi-chemin du mouvement entre les deux trames adjacentes. Ces vecteurs de mouvement divisés sont finalement utilisés pour la compensation de mouvement à partir de la trame $t-1$, conduisant ainsi à une première estimation de la trame t. Ces opérations sont répétées en partant d'une estimation pyramidale des vecteurs de mouvement pour interpoler cette fois-ci la trame $t-1$ à partir de la trame $t+1$. On obtient de la même façon, une deuxième estimation de la trame t. Ces deux estimations sont finalement moyennées pour obtenir une version de la trame interpolée utilisant l'algorithme multi-résolution de Lukas-Kanade adaptée à l'architecture du codage vidéo distribué.

FIGURE 6.17: *Approche multi-résolution.*

Implémentation de l'algorithme de Lukas-Kanade (approche pyramidale)

On implémente l'algorithme, tel décrit, à la trame WZ numéro 7 de la séquence vidéo `Foreman`, tout comme précédemment. Les vecteurs de mouvement vers l'avant et vers l'arrière obtenus sont affichés à la figure 6.20.

La compensation de mouvement bidirectionnelle effectuée à partir de ces vecteurs de mouvement donne une trame interpolée ayant un PSNR de 30.92 dB. On compare à la figure 6.21, cet algorithme avec les algorithmes décrit précédemment d'interpolation bidirectionnelle avec lissage et sans lissage spatial. La méthode de Lukas-Kanade permet d'avoir un meilleur PSNR que la méthode bidirectionnelle sans lissage mais la méthode bidirectionnelle avec lissage la dépasse de 3 dB.

Toutefois, on ne peut pas affirmer que la méthode d'estimation bidirectionnelle est meilleure que la méthode proposée basée sur l'algorithme de Lukas-Kanade à partir d'une seule trame. L'approche de Lukas-Kanade présente certaines limites lorsque le mouvement dans la fenêtre n'est pas constant, l'intensité n'est pas constante ou le mouvement est très grand [55]. Mais si ces conditions sont satisfaites, l'algorithme de Lukas-Kanade pourrait résulter en de meilleures performances. Dans la section suivante, on présente le résultat d'interpolation des différentes techniques d'interpolation étudiées pour une série de séquences vidéo afin de mener une étude comparative.

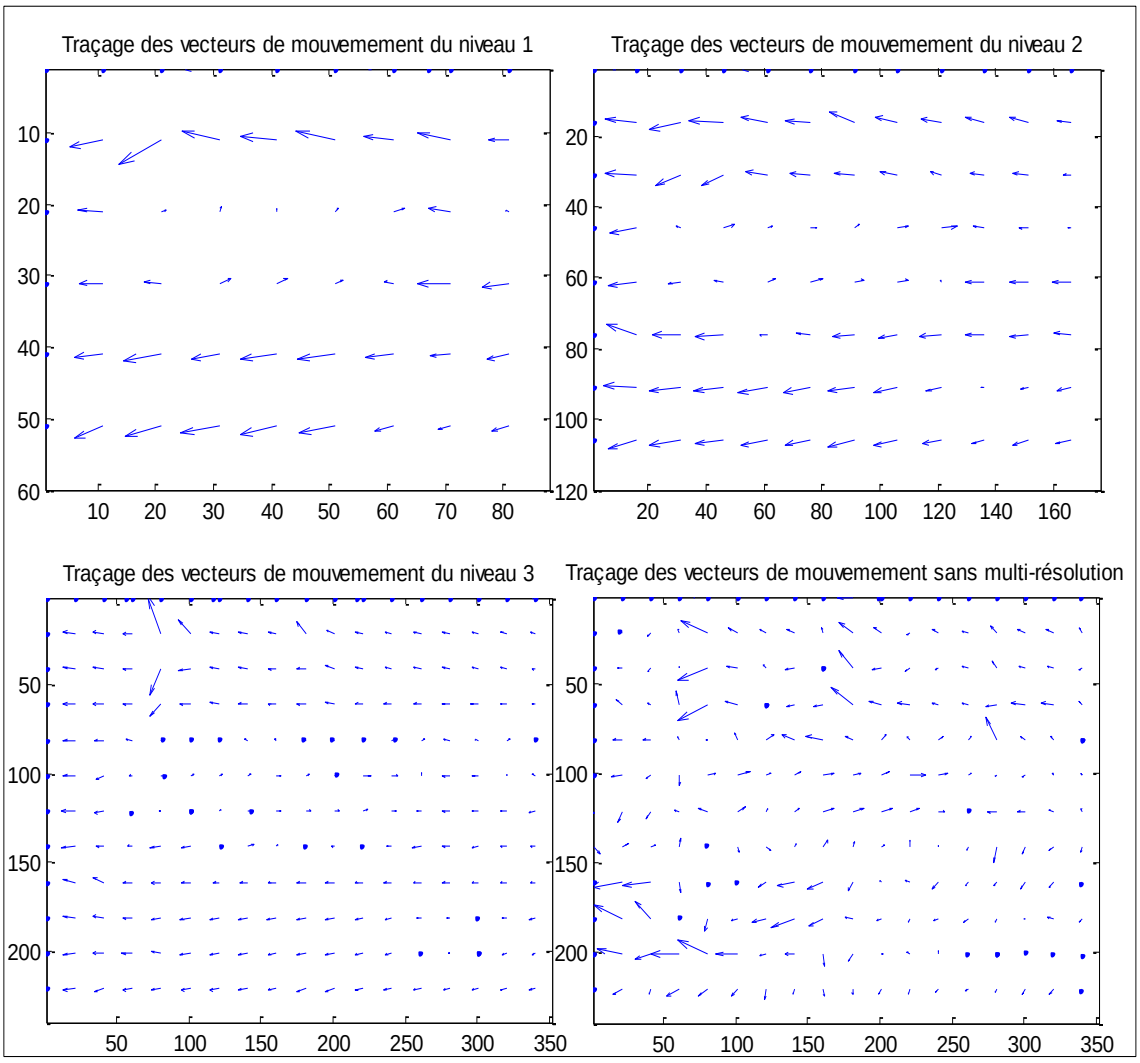

FIGURE 6.18: *Traçage des vecteurs de mouvemement entre l'image 1 et l'image 3 déterminés avec la méthode de Lukas-Kanade avec et sans pyramide.*

6.7 Application des différentes techniques d'interpolation bidirectionnelle pour diverses séquences vidéo

On effectue dans cette section une étude comparative entre les 5 différentes techniques d'interpolation présentées précédemment. On rappelle ci-dessous ces techniques :

1. **Interpolation par moyennage** : c'est la technique la plus simple vu qu'elle consiste à prendre la moyenne entre la trame précédente et la trame suivante. Cette méthode ne peut donner de bonnes performances que lorsqu'il n'y a pas beaucoup de mouvements rapides dans la séquence vidéo.

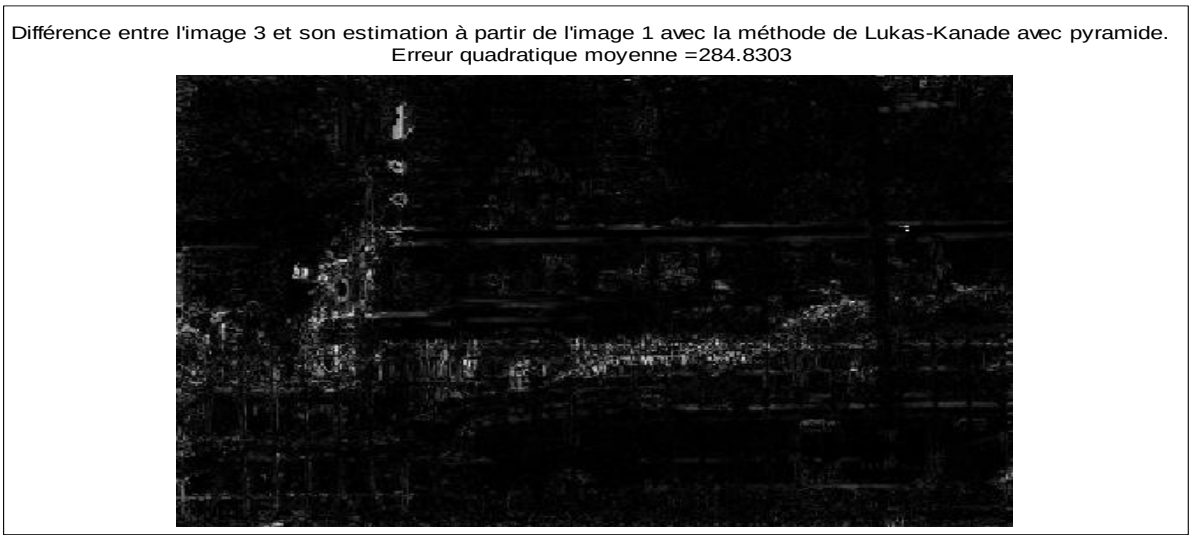

FIGURE 6.19: *Réduction de l'erreur d'interpolation avec l'approche pyramidale de l'algorithme de Lukas-Kanade (EQM =284.83).*

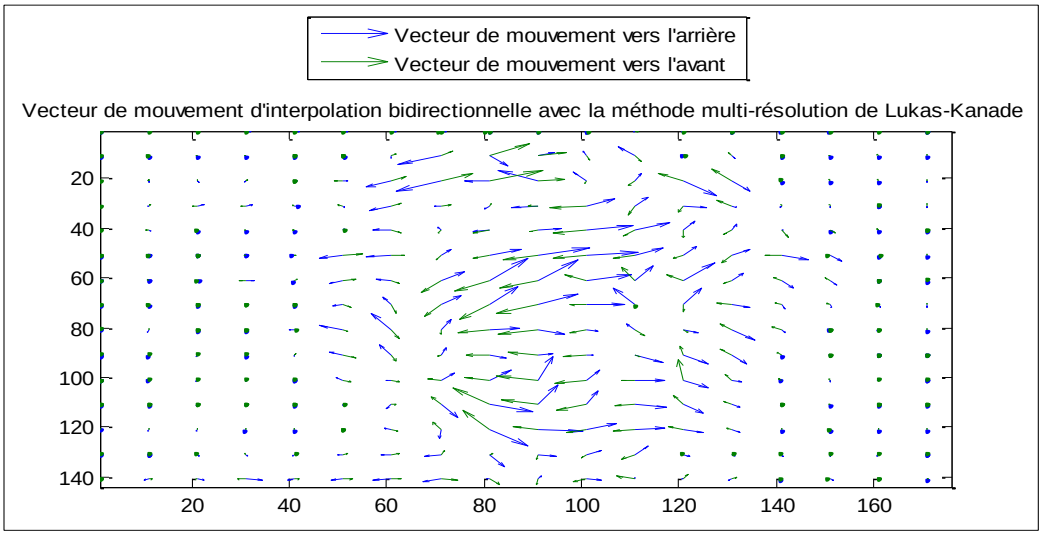

FIGURE 6.20: *Vecteurs de mouvement bidirectionnels obtenus avec l'approche pyramidale de l'algorithme de Lukas-Kanade.*

FIGURE 6.21: *Résultat de la compensation de mouvement bidirectionnelle de l'algorithme de Lukas-Kanade en comparaison avec les méthodes présentées précédemment.*

2. **Interpolation par la méthode multidimensionnelle de Lukas-Kanade** : cette technique présente une complexité calculatoire élevée : elle évalue un vecteur de mouvement pour chaque pixel. Cette technique donne de bonnes performances lorsque le mouvement est faible et constant dans le voisinage.
3. **Interpolation par estimation de mouvement vers l'avant** : cette technique est désignée par FME pour "*Forward Motion Estimation*" et elle constitue la première étape dans l'algorithme d'interpolation présenté dans [13].
4. **Interpolation par estimation bidirectionnelle de mouvement** : cette technique est désignée par BiME pour "*Bidirectionnel Motion Estimation*" et elle constitue la deuxième étape dans l'algorithme d'interpolation présenté dans [13].
5. **Interpolation par estimation bidirectionnelle de mouvement avec lissage spatial** : cette technique est désignée par BiMESS pour "*Bidirectionnel Motion Estimation with Spatial Smoothing*" et elle a été proposée dans [13] pour les applications de codage vidéo distribué.

Les simulations sont effectuées pour une série de séquences vidéo téléchargées du site Web du centre de l'Université de Stanford pour l'ingénierie des systèmes et des images [53] et de media.xiph.org [58], un site pour la distribution libre des séquences de test. On considère l'interpolation bidirectionnelle des différentes trames WZ correspondant aux trames paires et on calcule le PSNR moyen de toutes les trames interpolées. Les résultats de simulation pour les différentes séquences et pour les 5 techniques d'interpolation sont rapportés au tableau 6.1. On fournit dans ce tableau les détails des séquences utilisées ainsi que le nombre total de trames utilisées pour les simulations.

On remarque d'après ce tableau que la technique d'interpolation de mouvement bidirectionnelle avec lissage spatial est la mieux adapté aux différentes séquences et donne les meilleures valeurs de PSNR. La technique d'interpolation bidirectionnelle de Lukas-Kanade permet aussi d'avoir dans certains cas des valeurs similaires de PSNR surtout quand la nature du mouvement dans la séquence correspond aux hypothèses nécessaires pour cette technique. On remarque aussi que dans certains cas, l'opération

de moyennage donne de bons résultats : ceci correspond à des séquences de faible mouvement où la corrélation entre trames successives est très importante.

Pour comparer les résultats des différentes techniques d'interpolation au fil des trames, on trace aux figures 6.22, 6.23 et 6.24, les valeurs du PSNR pour chacune des trames WZ interpolées. Dans ces figures, on ne considère que 3 techniques d'interpolation : moyennage, Lukas-Kanade et interpolation bidirectionnelle de mouvement avec lissage spatial. On remarque que même si le moyennage donne un PSNR moyen élevé, le PSNR de certaines trames interpolées décroit abruptement alors que les 2 autres techniques d'interpolation (Lukas-Kanade et BiMESS) donnent dans la quasi-totalité des cas un PSNR élevé sans décroissance brusque. La comparaison entre les différentes techniques d'interpolation ne doit pas se restreindre à la valeur du PSNR moyen mais à chacun des PSNRs des trames WZ.

Séquence et résolution temporelle	# total de trames (K+WZ)	Moyennage	Lukas Kanade	FME	BiME	BiMESS
`Foreman` @ 30 fps	399	32.72	35.14	31.05	34.25	35.39
`Foreman` (Siemens) @ 30 fps	299	33.26	35.31	31.18	34.69	36.006
`Carphone` @ 30 fps	381	31.75	32.12	30.26	31.57	32.14
`Salesman` @ 30 fps	399	43.61	43.22	40.19	42.65	44.04
`Mother and Daughter` @ 30 fps	399	45.77	44.72	40.64	44.55	45.69
`Bus` @ 15 fps	75	19.90	21.44	20.42	21.87	23.17
`Grandma` @ 30 fps	299	46.94	46.72	42.49	46.52	47.08
`Coastguard` @ 25 fps	299	31.43	36.85	32.76	36.30	36.93
`Hall Monitor` @ 30 fps	299	40.02	40.25	36.95	39.60	40.38
`Miss America` @ 30 fps	149	45.07	45.21	40.93	44.32	45.67
`Suzie` @ 30 fps	149	38.60	40.22	35.73	40.02	40.96

Tableau 6.1: *PSNR moyen en dB des trames WZ interpolées avec différentes techniques.*

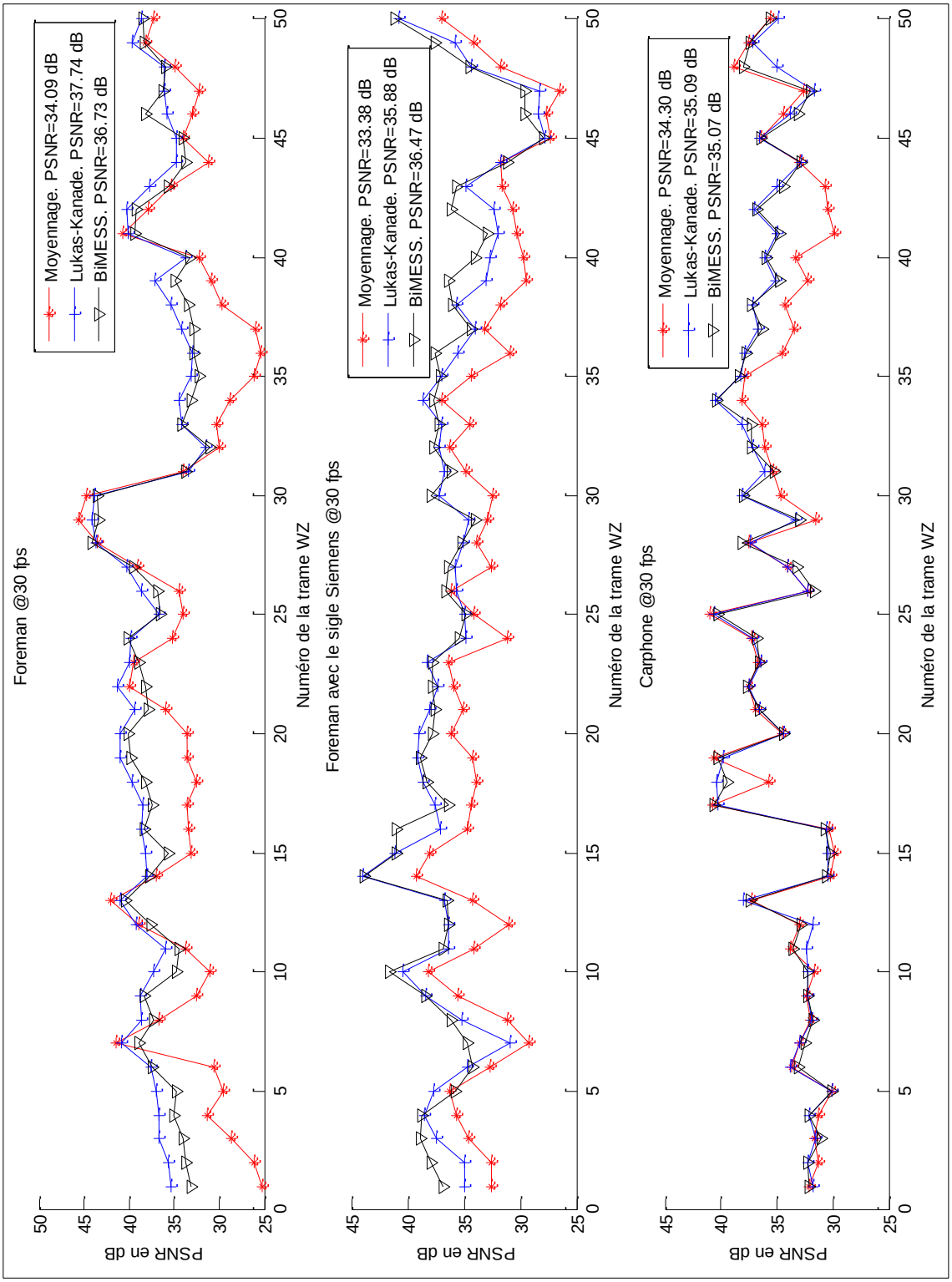

FIGURE 6.22: *PSNR en dB de chacune des trames WZ interpolées avec 3 techniques bidirectionnelles.*

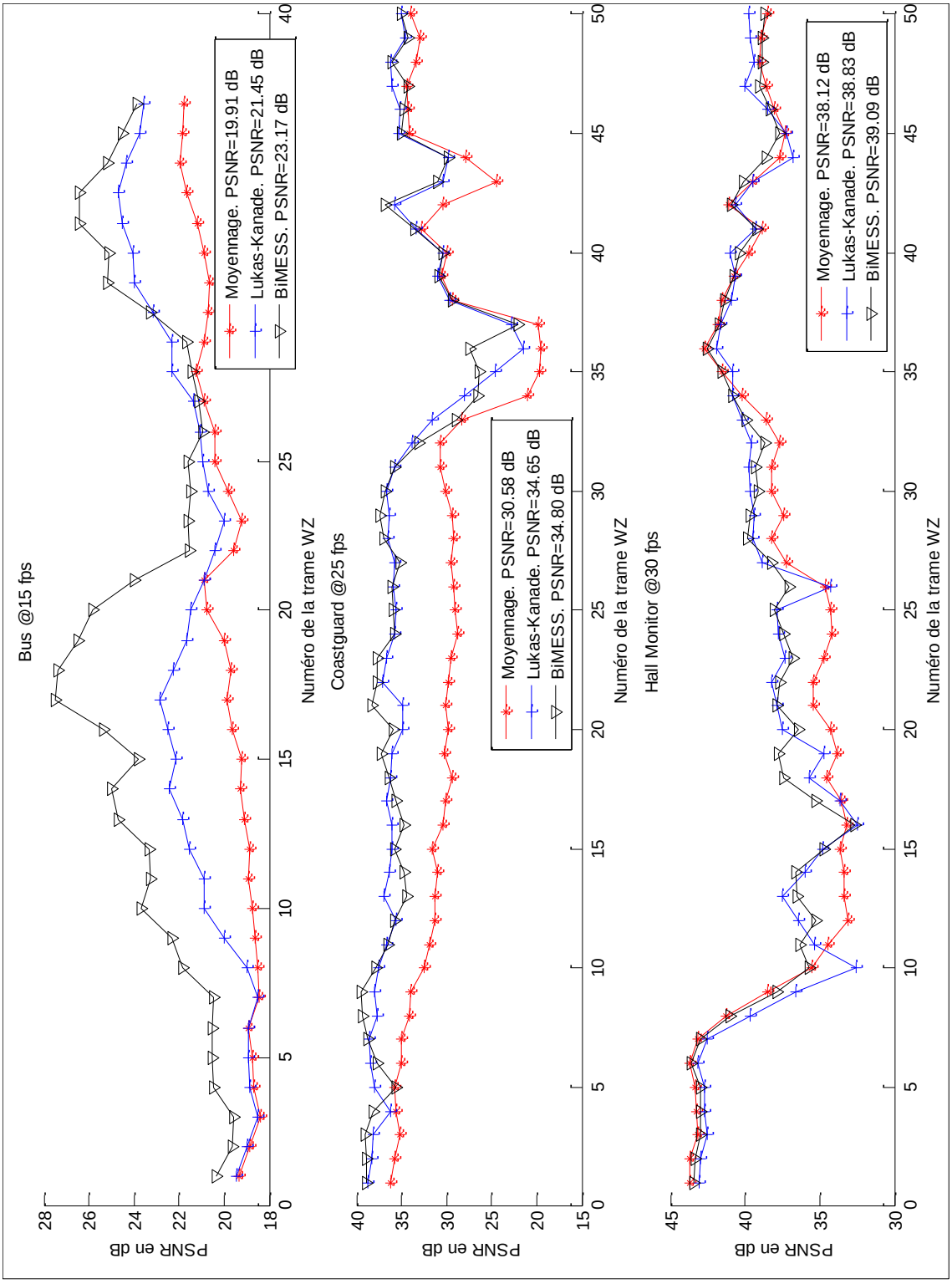

FIGURE 6.23: *PSNR en dB de chacune des trames WZ interpolées avec 3 techniques bidirectionnelles (suite 1).*

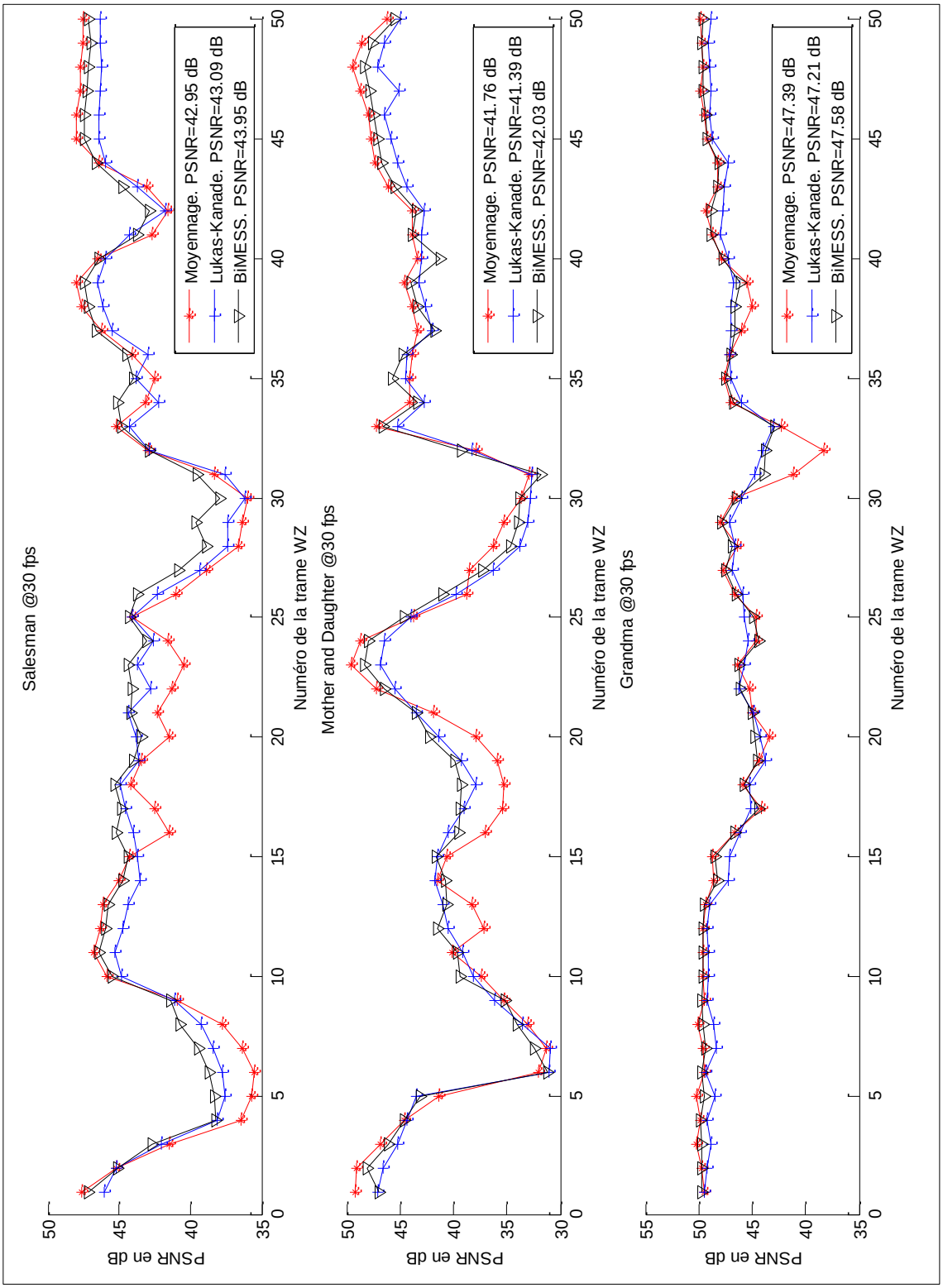

FIGURE 6.24: *PSNR en dB de chacune des trames WZ interpolées avec 3 techniques bidirectionnelles (suite 2).*

6.8 Incorporation du bloc d'interpolation dans l'architecture DVC

Dans cette section, on présente les résultats de simulations d'un codeur/décodeur WZ avec les différentes techniques d'interpolation. On commence tout d'abord par une comparaison des résultats obtenus avec certains résultats publiés dans la littérature. Cette comparaison permet de valider l'implémentation des divers blocs de l'architecture DVC dans le domaine pixel et dans le domaine transformé. Dans l'article de Brites et al. [11], on présente les performances du système DVC dans les deux domaines pour la séquence `Foreman` et pour la séquence `Mother and Daughter` en considérant les 101 premières trames (50 trames WZ et 51 trames K). Pour le calcul du débit, on prend en considération le nombre des bits nécessaires pour la transmission de la plage dynamique des bandes des coefficients. Ce nombre correspond à 16 bits pour chaque bande [11]. En reproduisant, de façon générale, les mêmes conditions de leur expérimentation (même quantificateur, méthode d'interpolation et technique de reconstruction), on parvient à obtenir des résultats semblables [1] comme on peut le constater à la figure 6.25 et à la figure 6.26. Cependant, on obtient, dans nos simulations, des PSNRs légèrement supérieurs et ceci peut être dû à l'un des points suivants :

- **Le choix du paramètre α de la Laplacienne** : le paramètre α, dans [11], est fixé pour chaque bandes de coefficients suite à une phase d'apprentissage mais on n'explicite pas exactement cette procédure d'apprentissage ni les données utilisées. Dans nos simulations, le choix du paramètre α se base sur un apprentissage hors ligne en considérant les bandes de coefficients de toutes les trames de la séquence vidéo.
- **Implémentation du bloc d'interpolation** : lors de l'implémentation de la technique d'estimation bidirectionnelle de mouvement avec lissage spatial, on a suivi les étapes énoncées dans [13] tout en laissant libre cours à notre compréhension de l'algorithme dans les étapes implicites. Ainsi, vu les différents aspects de cette technique, une certaine différence peut exister dans l'implémentation et cette différence peut apporter un certain gain de PSNR.
- **Processus de décodage turbo** : la partie interne du processus du décodage turbo peut constituer une différence d'un algorithme à un autre, surtout que dans la littérature on ne donne pas généralement les détails complets de l'implémentation turbo. Ce qui est compréhensible vu les divers aspects de l'algorithme de ce code incorporé dans une architecture distribuée (choix du nombre d'itérations, de l'entrelaceur, des conditions d'arrêt du processus de décodage ...).

Aux figures 6.25 et 6.26, on compare aussi les techniques d'interpolation suivantes : interpolation par moyennage, interpolation par estimation de mouvement avec la technique pyramidale de Lukas-Kanade et interpolation par estimation bidirectionnelle de mouvement avec lissage spatiale (BiMESS). Cette comparaison est effectuée dans le domaine pixel et dans le domaine transformé. Pour la séquence `Mother and Daughter`, on remarque que la technique d'interpolation BiMESS et de moyennage sont plus appropriées pour l'estimation du mouvement de cette séquence que la technique d'interpolation de Lukas-Kanade. Cette dernière technique donne un PSNR de 1 dB moindre que la technique

1. On trouve des résultats de même ordre mais pas exactement les mêmes. Ceci permet tout de même de valider, de point de vue d'ensemble, l'implémentation des différents blocs interdépendants de l'architecture DVC.

Nom de la séquence	Mother and daugther	Foreman	Coastguard	Bus
Échantillon				
# total de trames	961	400	299	75
# de trames utilisées	101	101	101	75
# de trames WZ	50	50	50	37
# de trames clés	51	51	51	38
Résolution spatiale	QCIF (176×144)	QCIF (176×144)	QCIF (176×144)	QCIF (176×144)
Résolution temporelle	30 trames par seconde	30 trames par seconde	25 trames par seconde	15 trames par seconde
Format de téléchargement	*.qcif [53]	*.qcif [53]	*.yuv [58]	*.y4m [58]

Tableau 6.2: *Caractéristiques des séquences vidéo utilisées dans les simulations.*

BiMESS. Cependant, pour la séquence `Foreman`, on remarque que la technique d'interpolation de Lukas-Kanade donne de meilleures performances de débit-distorsion que la technique d'interpolation BiMESS. On note, dans ce cas, un gain pouvant aller jusqu' à 0.7 dB.

Après avoir présenté les résultats de simulations pour validation, on compare à la figure 6.27 les performances de la technique d'interpolation de Lukas-Kanade et la technique d'interpolation BiMESS. On considère dans cette comparaison 2 modes de reconstruction (standard et optimale), 2 mesures de la distorsion (PSNR et SSIM) et 4 séquences vidéos (`Mother and Daughter`, `Foreman`, `Coastguard` et `Bus`. Les caractéritiques de ces séquences sont regroupées dans le tableau 6.2.

On remarque d'après la figure 6.27, que la technique d'interpolation BiMESS donne de meilleures performances que la technique de Lukas-Kanade pour toutes les séquences sauf pour la séquence `Foreman`. En fait, en défilant cette séquence vidéo on remarque qu'il y a un mouvement rapide de la caméra alors que la technique BiMESS est limitée à une zone de recherche de ± 8 pixels. Pour les 3 autres séquences vidéo les performances de la technique BiMESS dépasse, mais de peu, les performances de la technique de Lukas-Kanade.

6.9 Conclusion

Dans ce chapitre, on s'est intéressé aux techniques d'interpolation qui peuvent être utilisées pour la génération de l'information latérale dans un système de codage vidéo distribué. La procédure standard d'estimation et de compensation de mouvement a été révisée pour pouvoir l'appliquer à la situation où la trame de référence n'existe pas, mais plutôt, les trames qui lui sont voisines.

On a repris, en premier lieu, la technique d'interpolation employée typiquement par la communauté du codage vidéo distribué. Cette technique consiste à estimer les vecteurs de mouvement de façon bidirectionnelle puis à maintenir la corrélation spatiale entre les vecteurs de mouvement des blocs voisins, d'où elle est désignée par la technique d'interpolation de mouvement bidirectionnelle avec lissage spatial (des vecteurs obtenus) avec l'alias BiMESS. En second lieu, on a exposé le problème d'estimation du flux optique et on s'est concentré sur la technique multi-résolution de Lukas-Kanade. Cette technique est ensuite réadaptée au contexte du codage vidéo distribué en estimant les vecteurs de mouvement entre les trames aux instants $i-1$ et $i+1$ et vice versa entre les trames aux instants $i+1$ et $i-1$. Par la suite, ces vecteurs de mouvement sont divisés par 2 et la trame interpolée est reconstituée en moyennant les deux versions obtenus par compensation de mouvement à partir de la trame $i-1$ et la trame $i+1$.

Ces deux techniques sont ensuite employées pour la génération de l'information latérale pour une multitude de séquences. La technique BiMESS est plus efficace que la technique de Lukas-Kanade, dans le contexte distribué, pour la plupart des séquences. Cependant, la technique de Lukas-Kanade peut donner dans certains cas de meilleurs résultats si les hypothèses reliées à son application sont satisfaites.

Finalement, on a mis en place le bloc d'interpolation dans l'architecture DVC pour valider la phase d'implémentation en effectuant une comparaison avec des résultats de la littérature. Cette étape a permis de valider le fonctionnement d'ensemble, dans le domaine pixel et dans le domaine transformé, du système DVC qui est constitué de différents blocs relativement complexes. D'autres simulations sont effectuées, par la suite, pour d'autres séquences vidéo pour comparer entre les performances des deux techniques d'interpolation étudiée, une fois incorporées dans le décodeur WZ.

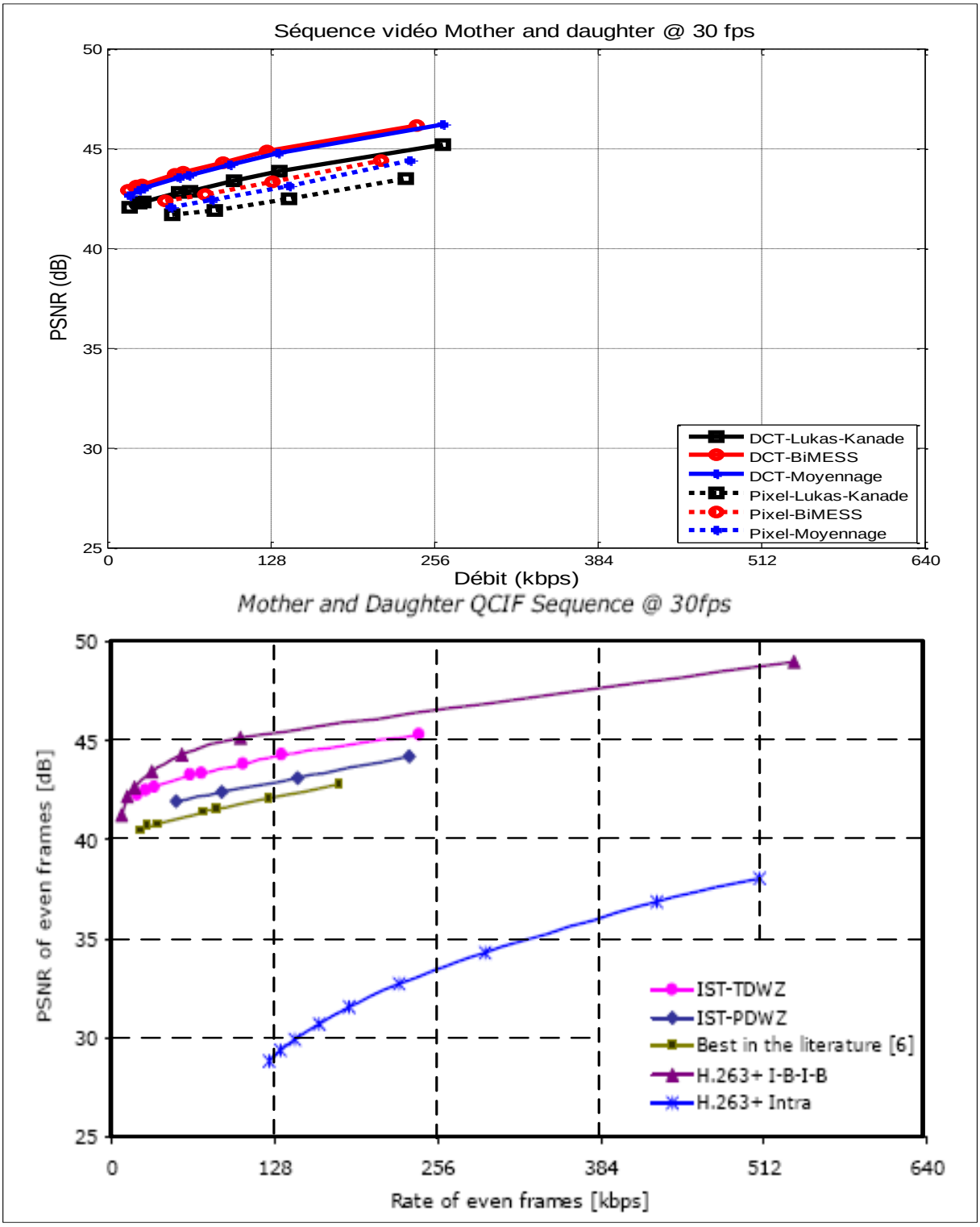

FIGURE 6.25: *Comparaison entre les performances des systèmes DVC utilisant les différentes techniques d'interpolation et validation de la phase d'implémentation avec les résultats extraits de [11] (séquence* `Mother and Daughter`*). La figure du haut représente les résultats des simulations effectuées et la figure du bas est obtenue par un "imprime écran" à partir de l'article de Brites et al. [11].*

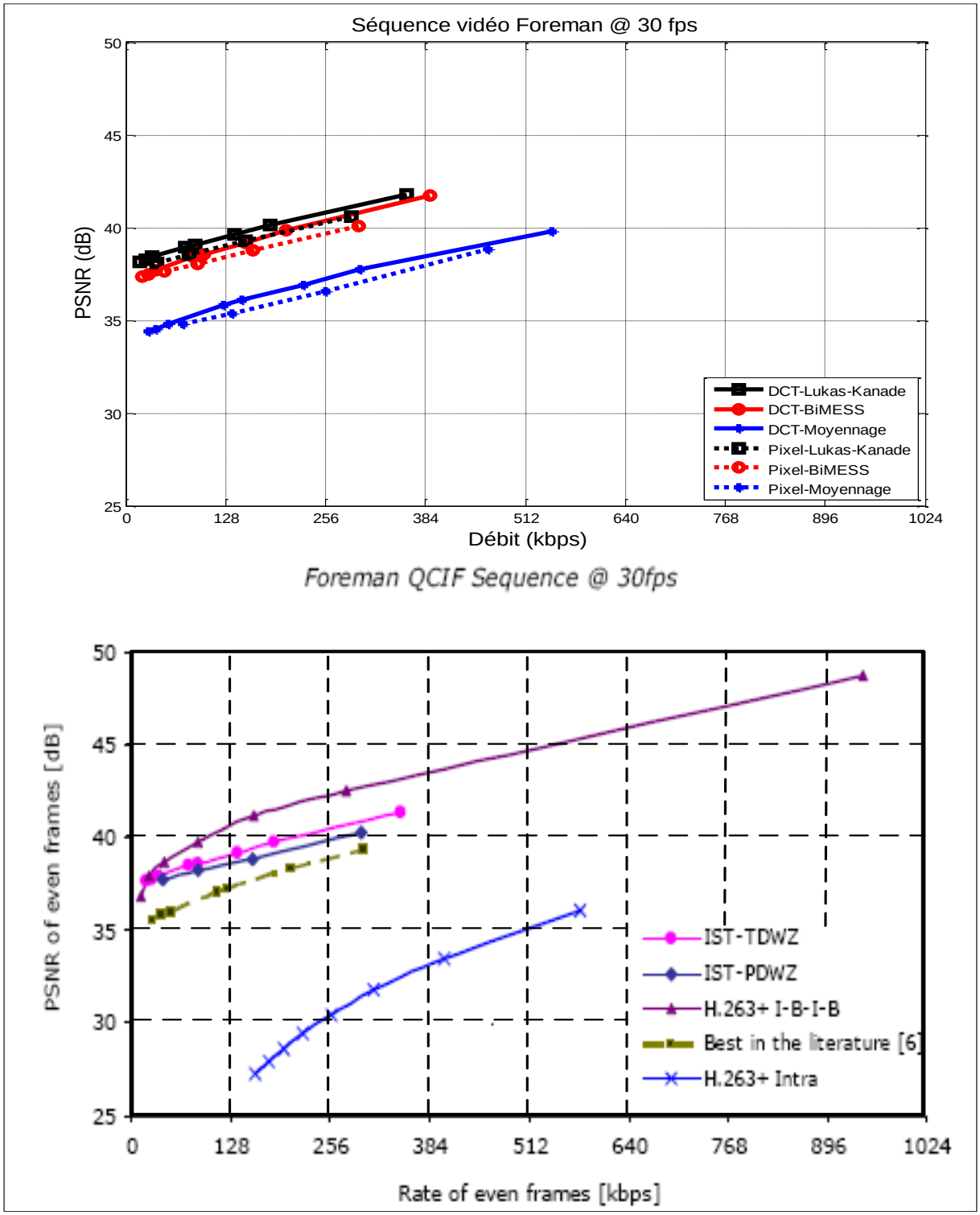

FIGURE 6.26: *Comparaison entre les performances des systèmes DVC utilisant les différentes techniques d'interpolation et validation de la phase d'implémentation avec les résultats extraits de [11] (séquence* `Foreman`*). La figure du haut représente les résultats des simulations effectuées et la figure du bas est obtenue par un "imprime écran" à partir de l'article de Brites et al. [11].*

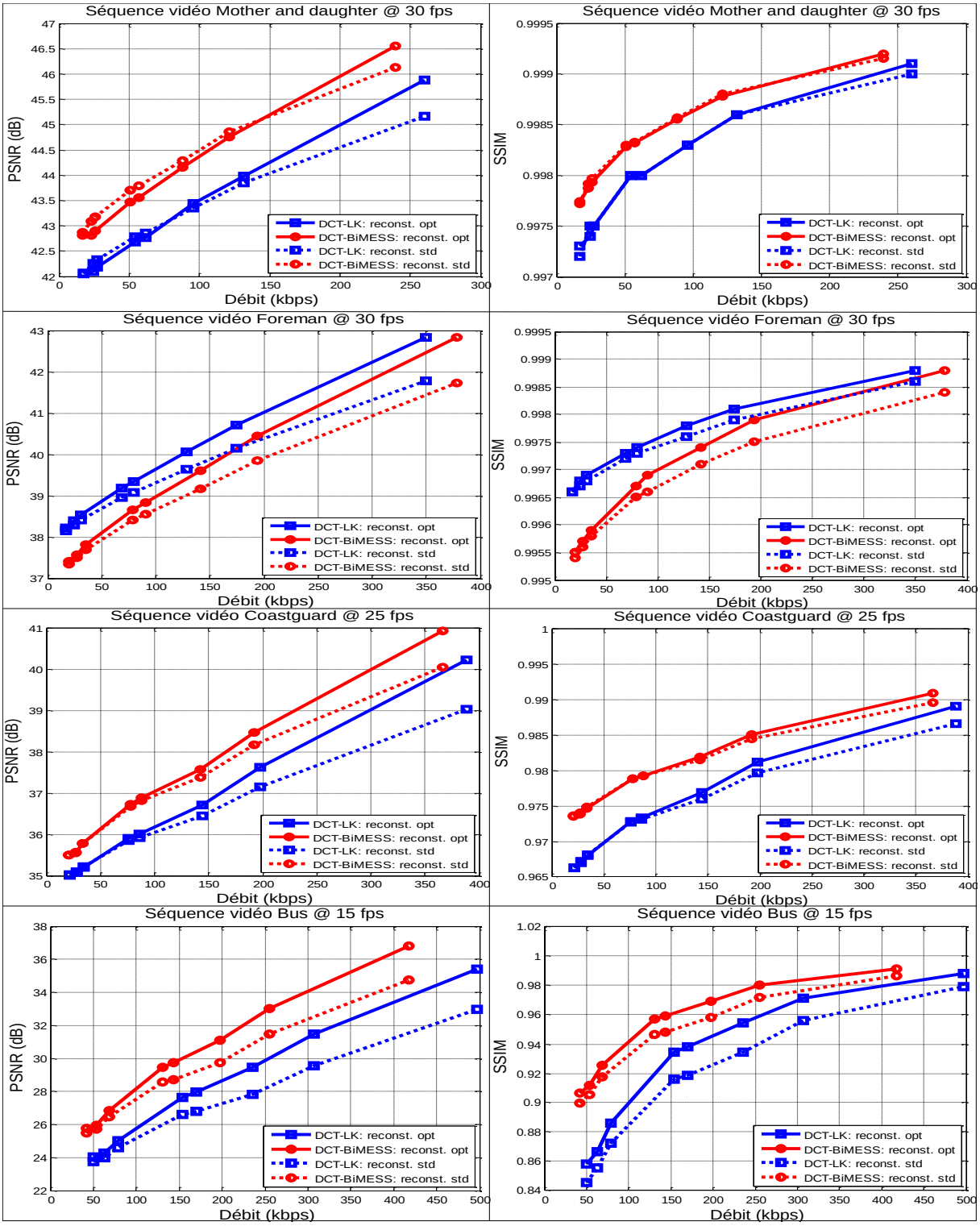

FIGURE 6.27: *Comparaison entre les performances des systèmes DVC utilisant la technique d'interpolation de Lukas-Kanade et la technique d'interpolation BiMESS.*

Chapitre 7

Étude et implémentation des divers mécanismes dans le projet DISCOVER

7.1 Introduction

Dans les chapitres précédents, l'intérêt s'est porté principalement à reproduire l'implémentation de l'architecture DVC de l'Université de Stanford [25] et l'architecture DVC de l'Institut technique supérieur au Portugal [11]. Lors de l'implémentation de ces architectures, une compréhension approfondie du fonctionnement des divers composants du système DVC a été possible. Par ailleurs, des résultats similaires ont été obtenus permettant de s'assurer du bon fonctionnement des divers composants interdépendants que l'on rappelle ici [1] : (1) la transformation DCT, (2) la quantification uniforme des coefficients DC et la quantification avec un intervalle centré autour de zéro pour les coefficients AC, (3) l'extraction des plans de bits, (4) l'encodeur turbo à l'aide de 2 RSCs 1/2 séparés par un entrelaceur et un dispositif de poinçonnage permettant d'adapter le débit à la variation du canal virtuel, (5) le bloc d'interpolation pour la génération de l'information latérale basé sur la technique d'estimation de mouvement bidirectionnelle avec lissage spatial, (6) le décodeur turbo utilisant l'algorithme BCJR log-MAP avec demande de bits de parité supplémentaires à travers un canal de retour en cas de non convergence, (7) le bloc de reconstruction optimale permettant de décider du coefficient DCT de façon statistiquement optimale, (8) le bloc de la transformation DCT inverse pour le retour au domaine pixel.

Lors de cette étude de l'état de l'art, nous avons pu non seulement comprendre le fonctionnement du codeur/décodeur WZ mais aussi proposer certaines techniques et idées pour améliorer les performances des courbes de débit-distorsion : technique de poinçonnage adaptatif et technique de codage progressif. Dans ce chapitre, on s'intéresse à un autre système DVC, le système DVC Discover (DIStributed COding for Video sERvices) [9], utilisé intensivement comme référence (*benchmark*) par la communauté du codage vidéo distribué en raison des avantages qu'il présente. Non seulement les

1. L'architecture dans le domaine pixel ne comporte pas de transformation DCT et elle utilise un quantificateur scalaire uniforme.

performances de ce système sont très concurrentielles, mais aussi il permet de diminuer les inconvénients d'ordre pratique, ouvrant ainsi la porte vers la commercialisation et l'industrialisation du paradigme du codage vidéo distribué. De plus, Discover incorpore un certain nombre de mécanismes et de techniques qui font l'objet d'une étude dans ce chapitre : (1) mécanisme de contrôle adaptatif de la taille des groupes d'images (GOP), (2) système d'interpolation pour un GOP flexible, (3) dispositif de modélisation du canal virtuel au décodeur au niveau du coefficient.

Ce chapitre est organisé comme suit. On commence par présenter une description du projet de Discover et ses objectifs. Ensuite, on s'intéresse à l'étude et l'implémentation des nouveaux composants introduit par Discover. Ce travail vise à préparer une nouvelle plateforme DVC à partir de laquelle on peut incorporer les techniques de codage progressif que nous allons élaborer ultérieurement.

7.2 Description du projet Discover

Discover est un projet qui s'est étalé sur une durée de 27 mois initié par un programme de l'Union Européenne, lancé le 1er septembre 2005. Ce projet est le fruit de la collaboration entre 6 groupes de recherche dans diverses universités européennes comme le montre le tableau 7.1.

L'objectif principal de l'atelier Discover est de canaliser les efforts d'une vaste communauté de chercheurs et d'industriels dans le monde entier qui ont manifesté un grand intérêt pour le codage vidéo Wyner-Ziv. Ce thème est devenu, en quelques années, un sujet de recherche attrayant et stimulant dans la communauté de codage vidéo en raison des nouveautés conceptuelles, théoriques et fonctionnelles qu'il apporte. Discover a permis de récapituler les avancées effectuées dans le codage vidéo distribué et d'établir les outils visant à doter ce domaine d'un fort potentiel fonctionnel pour migrer des laboratoires de recherche vers les applications réelles.

Discover est basé sur le codeur vidéo Wyner-Ziv développé par Brites *et al.* [11] . Ce dernier propose des améliorations sur le codeur vidéo Wyner-Ziv proposé dans les travaux de recherche originaux sur le codage vidéo distribué par Aaron et Girod [22, 25]. Même si le projet Discover reproduit le schéma DVC de la littérature, il a incorporé un certain nombre de nouveaux composants visant non seulement à augmenter la performance mais aussi à établir une architecture opérationnelle et réaliste. Ces nouveaux composants sont mis en relief dans la figure 7.1 où l'on présente l'architecture globale du benchmark Discover.

Avant Discover les algorithmes proposés souffraient généralement de deux aspects limitant l'implémentation pratique du DVC :

1. Le premier aspect est l'utilisation de l'information latérale au niveau de l'encodeur pour estimer le paramètre α de la distribution Laplacienne. Ce paramètre est ensuite envoyé au décodeur et sert pour la génération des valeurs de vraisemblance canal lors du décodage turbo. La connaissance de l'information latérale constitue une entrave à la notion distribuée : sa génération au niveau de l'encodeur suppose que ce dernier dispose des ressources matérielles nécessaires

Partenaire	Acronyme	Sigle	Pays	Contact
Universitat Politècnica de Catalunya (coordinator)	UPC		Espagne	Luis Torres
Instituto Superior Técnico	IST		Portugal	Fernando Pereira
Ecole Polytechnique Fédérale de Lausanne	EPFL		Suisse	Touradj Ebrahimi
Leibniz Universität Hannover	UH		Allemagne	Joern Ostermann
Institut National de Recherche en Informatique et en Automatique	INRIA		France	Christine Guillemot
Università di Brescia	UNIBS		Italie	Riccardo Leonardi

Tableau 7.1: *Partenariat de recherche pour la conception et l'implémentation du système de référence (benchmark) Discover pour diriger les travaux dans le domaine du codage vidéo distribué [5].*

pour cet effet. Cependant, l'architecture vidéo distribué part de l'hypothèse que l'encodeur ne doit supporter qu'une charge calculatoire réduite. Dans Discover, l'idée consiste à estimer le paramètre α au niveau du décodeur (estimation en ligne) en utilisant les trames clés adjacentes à la trame WZ [59, 60]. Plus de détails sont fournis dans la suite de ce chapitre.

2. Le second aspect suppose carrément la connaissance de l'information originale au niveau du décodeur lors de l'estimation du taux d'erreur binaire à la sortie du décodeur turbo ou LDPC. En fonction de ce taux, une demande de bits de parité supplémentaires est envoyée ou non à l'encodeur. La solution adoptée pour un décodeur LDPC afin d'estimer le taux d'erreur binaire sans utilisation de l'information originale consiste à utiliser un code CRC (*Cyclic Redundancy Check*) de longueur $N = 8$ envoyé par l'encodeur [39]. Pour le décodage turbo, la probabilité d'erreur peut être estimée en observant les statistiques du flux décodé (les valeurs de vraisemblance) et la convergence du turbo code.

En ce qui concerne les nouvelles techniques déployée par Discover afin d'augmenter les performances de débit-distorsion, on dénote 3 points principaux qui sont étudiés par la suite afin de les implémenter :

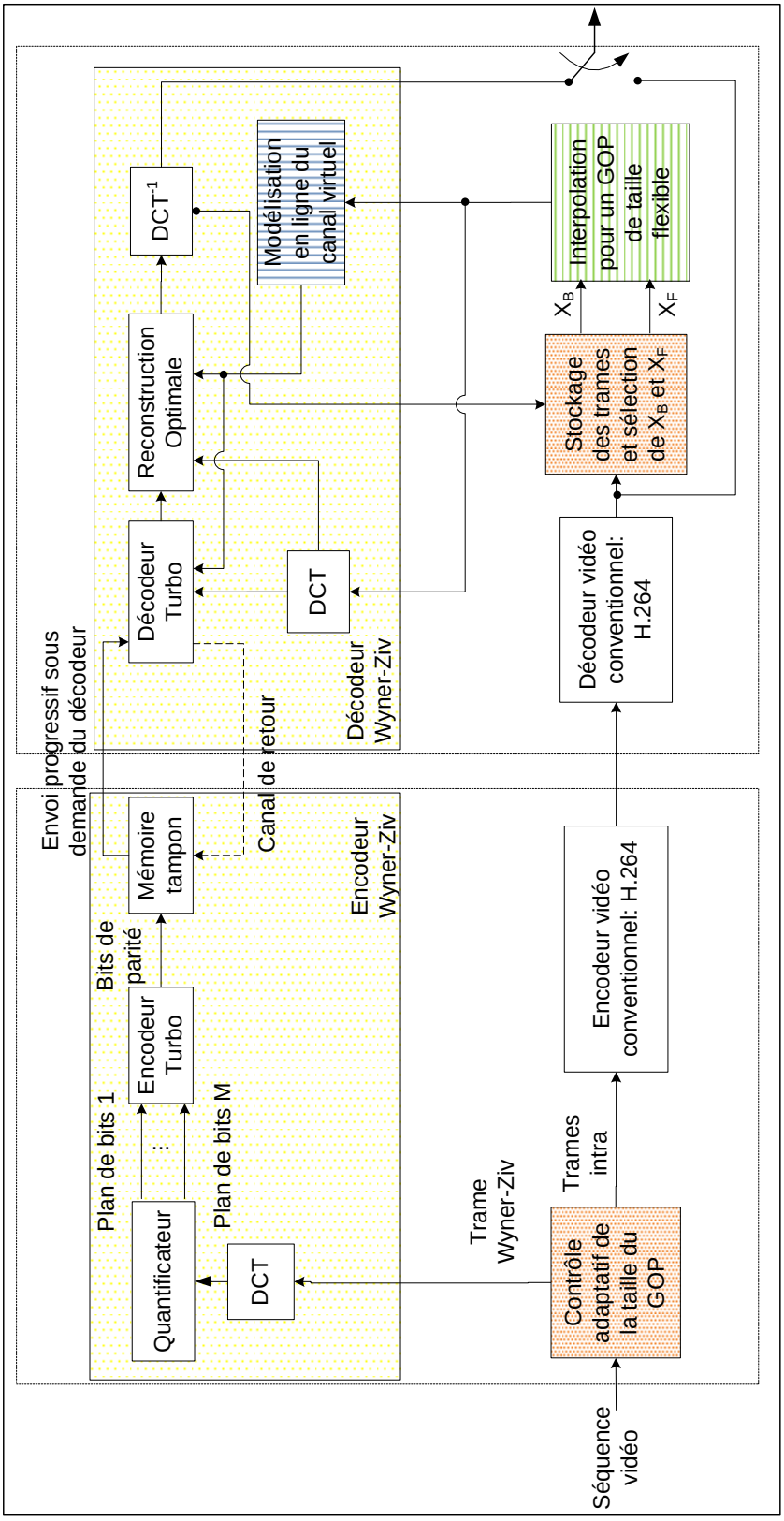

FIGURE 7.1: *Diagramme bloc de l'architecture DVC proposée par Discover.*

1. **Mécanisme de contrôle adaptatif de la taille des groupes d'images :** Ce mécanisme consiste à optimiser les endroits d'insertion des trames clés. D'un coté, au moment où la vidéo présente un mouvement rapide, un groupe d'images (GOP) de taille réduite est utilisé. D'un autre côté, lors des mouvements lents, un GOP de taille élevée peut être utilisé.
2. **Algorithme d'interpolation pour une taille du groupe d'images flexible :** Face à la variation adaptative du GOP le long de la séquence vidéo, l'algorithme d'interpolation doit présenter une certaine flexibilité pour permettre de générer une information latérale de bonne qualité même s'il s'agit d'une taille de GOP élevée, ou qui n'est pas une puissance de 2 (non symétrique).
3. **Modélisation en ligne du canal virtuel au niveau des coefficients :** Le codec DVC Discover permet d'estimer le paramètre α au niveau du décodeur (en ligne) pour éviter la génération de l'information latérale au niveau de l'encodeur. De plus, Discover utilise une estimation du paramètre α au niveau des coefficients. C'est-à-dire que l'on utilise différents paramètres α par trame avec un paramètre α pour chaque coefficient DCT. Ainsi, le décodeur turbo peut bénéficier d'une certaine information sur la confiance qu'il doit accorder à l'information latérale à un instant donné.
4. **Estimation du débit minimal au niveau de l'encodeur :** Une technique hybride mettant en œuvre une collaboration entre l'encodeur et le décodeur est utilisée pour essayer de prédire le débit minimal nécessaire pour la convergence du processus de décodage turbo. Ainsi un nombre de bits de parité égal à cette estimation minimale est envoyé d'un coup et si le décodeur ne converge pas une demande de bits supplémentaires est envoyée via le canal de retour. De cette manière le décodeur peut sauver quelques itérations dans son approche essai-erreur. Ceci réduit la latence du système ainsi que la charge calculatoire de décodage.

Les points énumérés sont détaillés dans ce qui suit.

7.3 Mécanisme de contrôle adaptatif de la taille des groupes d'images

Les solutions de codage vidéo distribué proposent généralement un découpage des trames de la séquence vidéo entre trames clés et trames Wyner-Ziv. Un certain ensemble de trames sont, en premier lieu, intra-codées avec un encodeur vidéo conventionnel, le H.264/AVC en mode intra, c'est-à-dire sans exploiter la corrélation temporelle au sein de la vidéo puis envoyées au récepteur. Cet ensemble de trames constitue les trames clés (trames K). En second lieu, les trames restantes (trames WZ) sont encodées par un encodeur Wyner-Ziv et décodées en utilisant une information latérale obtenue à partir des trames K et les trames WZ déjà décodées. Si la corrélation entre l'information latérale et la trame WZ est élevée, alors un nombre réduit de bits de parité serait nécessaire pour le décodage Wyner-Ziv. Pour augmenter la qualité de l'information latérale, outre l'utilisation d'un algorithme performant d'interpolation, l'insertion des trames clés au bon endroit peut s'avérer aussi très utile. Généralement, les trames clés sont insérées périodiquement, résultant ainsi en un groupe d'images fixes (GOP=2, 4 ou 8). Pour GOP$=2$, on encode une trame intra sur 2, on obtient ainsi le patron de trames suivant qui se qui se répète périodiquement : I-WZ-I-WZ-I-WZ-I-WZ-I. Pour le décodage de chaque trame WZ,

une information latérale est générée à partir d'une interpolation bidirectionnelle utilisant les trames clés adjacentes.

L'idée de l'insertion adaptative des trames clés, dont l'efficacité d'encodage ne nécessite aucune corrélation temporelle, consiste à les placer là où il n'y a pas beaucoup de corrélation à exploiter, tout en laissant cette corrélation porter profit à l'encodage des trames WZ. Supposons, à titre d'exemple, que les quatre premières trames sont très corrélées c'est-à-dire qu'il n'y a pas beaucoup de mouvement et que les deux dernières trames sont très différentes alors le découpage de la séquence suivra un séquencement non périodique comme par exemple : I-WZ-WZ-I-WZ-I-WZ-I-I.

7.3.1 Les métriques d'activité de mouvement

L'objectif de ce nouveau mécanisme est d'exploiter la redondance temporelle dans la vidéo en adaptant la longueur du GOP. On retrouve une telle approche aussi dans la compression vidéo traditionnelle dans l'arrangement dynamique des trames intra (I), des trames prédictives (P) et les trames bidirectionnelles (B). Cependant, cet arrangement dynamique exige de l'encodeur une charge supplémentaire pour effectuer une analyse du mouvement à travers la séquence, pour détecter le changement de scènes et ainsi pour déterminer la position des différents types de trames (I, P, ou B) dans le GOP. Pour le paradigme DVC, l'encodeur aux ressources limitées, ne peut supporter l'analyse du mouvement dans la vidéo. Étant donné cette limitation, l'adaptation de la taille du GOP doit se restreindre à des techniques efficaces et à faible complexité calculatoire. Ces techniques doivent être capables d'évaluer l'activité le long de la vidéo sans toutefois supporter les charges de l'estimation de mouvement.

Les métriques utilisées dans Discover pour évaluer l'activité de mouvement sont extraites des travaux effectués dans [61]. Ces métriques simples et efficaces utilisent les mêmes caractéristiques de bas niveau qu'on retrouve dans les applications d'analyse et d'indexation des bases de données vidéo de grande taille [62]. Le mécanisme de contrôle adaptatif de la taille du GOP se base sur la combinaison des 4 mesures suivantes [61] :

1. Différence des histogrammes (DH) :

$$DH(i,j) = \frac{1}{D_f} \sum_{k=0}^{L} \left| h_i(k) - h_j(k) \right| \tag{7.1}$$

2. Histogramme de la différence (HD) :

$$HD(i,j) = \frac{1}{D_f} \left(\sum_{k=0}^{\frac{L}{2}-\alpha} h_{i-j}(k) + \sum_{k=\frac{L}{2}+\alpha}^{L} h_{i-j}(k) \right) \tag{7.2}$$

3. Histogrammes de la différence des blocs (BHD) :

$$BHD(i,j) = \sum_{b=0}^{\frac{D_f}{D_B}} \sum_{k=0}^{L} \left| h_i(b,k) - h_j(b,k) \right| \tag{7.3}$$

4. Différence de la variance des blocs (BVD) :

$$BVD(i,j) = \sum_{b=0}^{\frac{D_f}{D_B}} \sum_{k=0}^{L} \left| \sigma_i^2(b,k) - \sigma_j^2(b,k) \right| \tag{7.4}$$

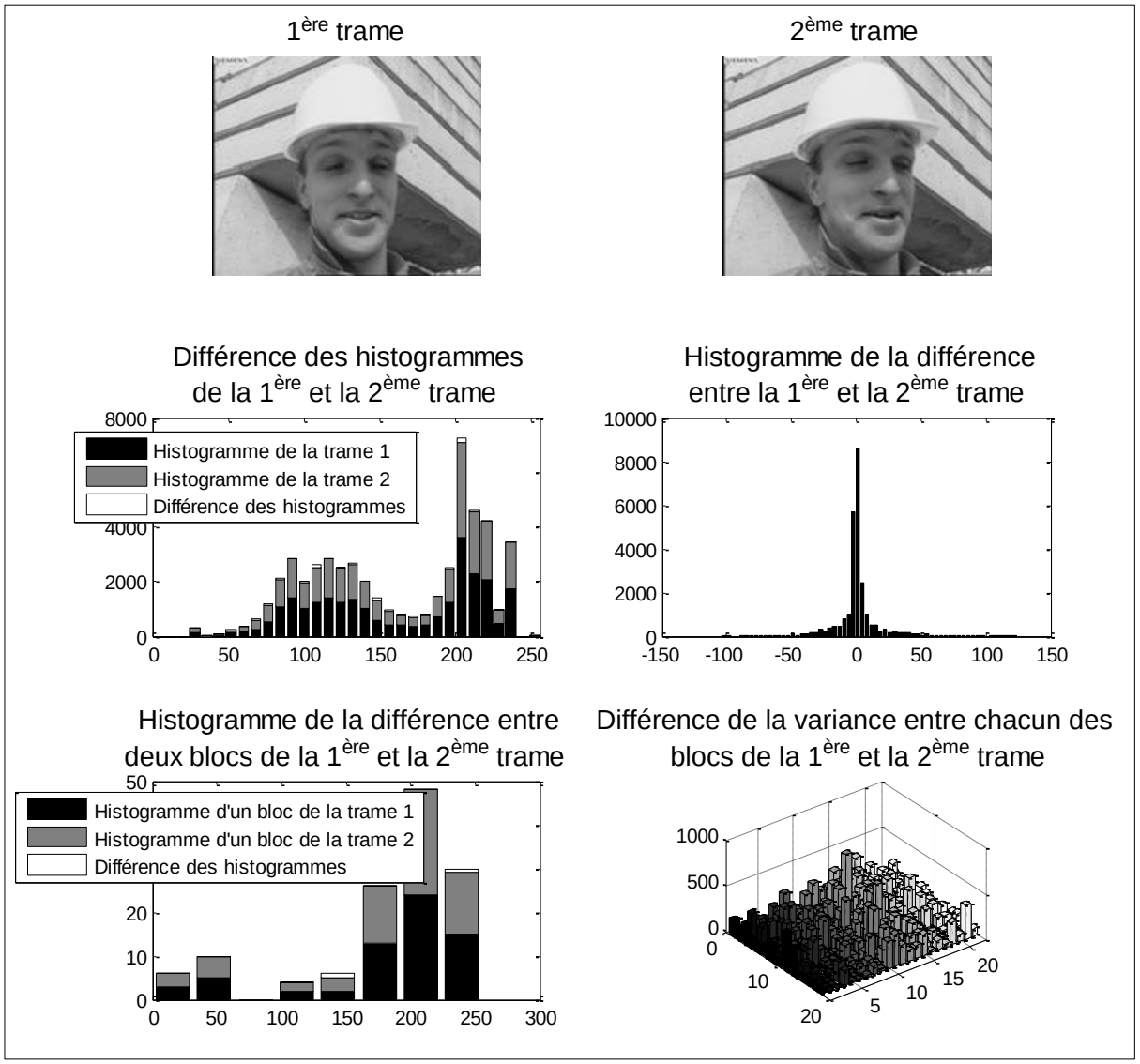

FIGURE 7.2: *Calcul des métriques de mouvement entre la trame 1 et la trame 2 de la séquence* `Foreman`.

Dans ces 4 équations i et j représentent les indices des trames entre lesquelles on souhaite évaluer l'activité du mouvement, h désigne l'opérateur de l'histogramme ayant L niveaux, D_f correspond à la taille de l'image, D_B à la taille du bloc et σ^2 à la variance. Pour la métrique HD, α représente le seuil de proximité à l'origine. En se basant sur des expérimentations, les auteurs de [61] ont trouvé que les meilleures performances sont obtenues avec $L = 32$ pour DH, $L = 64$ pour HD, $L = 8$ pour BHD, $D_B = 8 \times 8$ et $\alpha = 16$.

Les deux premières métriques, comme on le constate à partir de leur expression, opèrent au niveau de la trame en détectant le mouvement global comme le zoom, le mouvement panoramique et le changement de scènes. La métrique HD est particulièrement efficace pour détecter un mouvement global dans la trame. En effet, lors d'un changement significatif dû à un mouvement rapide par exemple, l'histogramme des pixels subit une grande modification. Les deux dernières métriques opèrent plutôt localement au niveau des blocs. Ces métriques sont plus efficaces pour détecter les mouvements locaux comme le déplacement d'objets dans un arrière plan statique. Un exemple de calcul de ces quatre métriques pour évaluer le mouvement entre la première trame et la deuxième trame de la séquence `Foreman` est donné à la figure 7.2.

7.3.2 Groupement hiérarchique basé sur l'activité de mouvement

Après avoir déterminé les 4 métriques de mouvement entre les différentes trames de la séquence, l'encodeur peut exploiter ces métriques pour effectuer la sélection de la taille du GOP le long de la vidéo. Intuitivement, lorsque le mouvement est élevé entre les trames, la corrélation est faible et un GOP de taille réduite est sélectionné. Lorsque le mouvement est faible, la corrélation est importante et un GOP de taille élevée est donc sélectionné. Ainsi les trames clés (intra) sont insérées lorsqu'il y a un grand mouvement. Lorsqu'il y a peu de mouvement, les trames WZ peuvent bénéficier d'une information latérale de bonne qualité grâce à la corrélation importante entre les trames clés adjacents et la trame WZ en question. Dans cette sous section, on décrit l'algorithme qui va nous permettre d'exploiter les métriques de mouvement pour décider de la taille du GOP. Cet algorithme est basé sur le groupement des images de façon hiérarchique et il se résume dans les étapes suivantes :

1. Calcul des 4 métriques entre toutes les paires de trames adjacentes pour construire les $N_c = M-1$ vecteurs de dimension 4 suivants :

$$x_i = [DH(i-1,i) \quad HD(i-1,i) \quad BHD(i-1,i) \quad BVD(i-1,i)], \quad 2 \leq i \leq N_c \tag{7.5}$$

2. Normalisation du vecteur comme suit :

$$x_i = \frac{N_c x_i}{\sum_{j=0}^{N_c} x_j}, \quad 1 \leq i \leq N_c \tag{7.6}$$

3. Accumuler le mouvement entre toute paire de trames avec le mouvement des paires de trames qui les précèdent (ou qui les suivent) :

$$y_i = x_i + x_{i-1}, \quad 2 \leq i \leq N_c \tag{7.7}$$

4. Déterminer l'indice c du minimum des valeurs de mouvement accumulé y_i.

$$c = \arg \min_{2 \leq i \leq N_c} \|y_i\| \tag{7.8}$$

Lors de l'implémentation, nous avons pris la norme 1 : $\|y_i\| = y_i^1 + y_i^2 + y_i^3 + y_i^4$.

5. Regroupement des deux trames ayant le minimum de mouvement (le maximum de corrélation) ainsi que leurs métriques de mouvement :

$$x_{i-1} = \begin{cases} x_i, & i \geq c+1 \\ y_i, & i = c \end{cases} \quad , \quad c \leq i \leq N_c \tag{7.9}$$

6. Décrémenter le nombre des trames clés :

$$N_c = N_c - 1 \tag{7.10}$$

7. Boucler vers l'étape 3 jusqu'à ce que la condition d'arrêt suivante soit remplie :

$$N_c = N_{\text{trames clés}} - 1 \tag{7.11}$$

où $N_{\text{trames clés}}$ est le nombre des trames clés que l'on souhaite garder. Ce nombre est fixé en fonction du nombre des trames dans la séquence vidéo M et de la taille du GOP moyen : $N_{\text{trames clés}} = \frac{M+1}{GOP_{\text{moyen}}}$. Si, par exemple, le $GOP_{\text{moyen}} = 2$ et $M = 149$, on obtient alors $N_{\text{trames clés}} = 75$ et $N_{\text{trames WZ}} = 74$. On note ici que généralement le choix est fait de telle sorte que la première et la dernière trame soient des trames intra.

On propose maintenant l'implémentation de l'algorithme décrit précédemment pour les 149 trames de la séquence `Foreman` en format QCIF à 15 trames par seconde. On présente aux figures 7.3 et 7.4, deux portions de 8 trames de cet algorithme qui s'étend sur 149 trames. Dans la première portion, on obtient deux GOP de taille 3 (I-WZ-WZ-I) et dans la deuxième portion, on obtient un GOP de taille 5 (I-WZ-WZ-WZ-WZ-I). Pour de telles valeurs de GOP (élevée et qui ne sont pas des puissances de 2), certains changements dans l'algorithme d'interpolation bidirectionnelle de mouvement se font nécessaires. Ceci fait l'objet de la section suivante.

7.4 Algorithme d'interpolation pour un GOP de taille flexible

L'algorithme d'interpolation des trames constitue un élément clé pour assurer de bonnes performances. En effet, l'interpolation permet de générer l'information latérale qui va intervenir non seulement pour le décodage canal et pour la reconstruction mais aussi pour la modélisation du canal virtuel comme on peut le voir à la figure 7.1. Dans cette section, on s'intéresse à l'interpolateur utilisé dans Discover qui se base principalement sur l'algorithme d'estimation de mouvement bidirectionnelle avec lissage spatial (BiMESS) [13]. Discover propose certaines modifications à cet algorithme pour assurer son efficacité même pour des séquences vidéo de résolution temporelle faible (soit 15 trames par seconde (tps) au lieu de 30 tps) et pour des GOP de taille élevée et non de puissance de 2.

7.4.1 Algorithme d'estimation de mouvement bidirectionnelle avec lissage spatial (BiMESS)

.

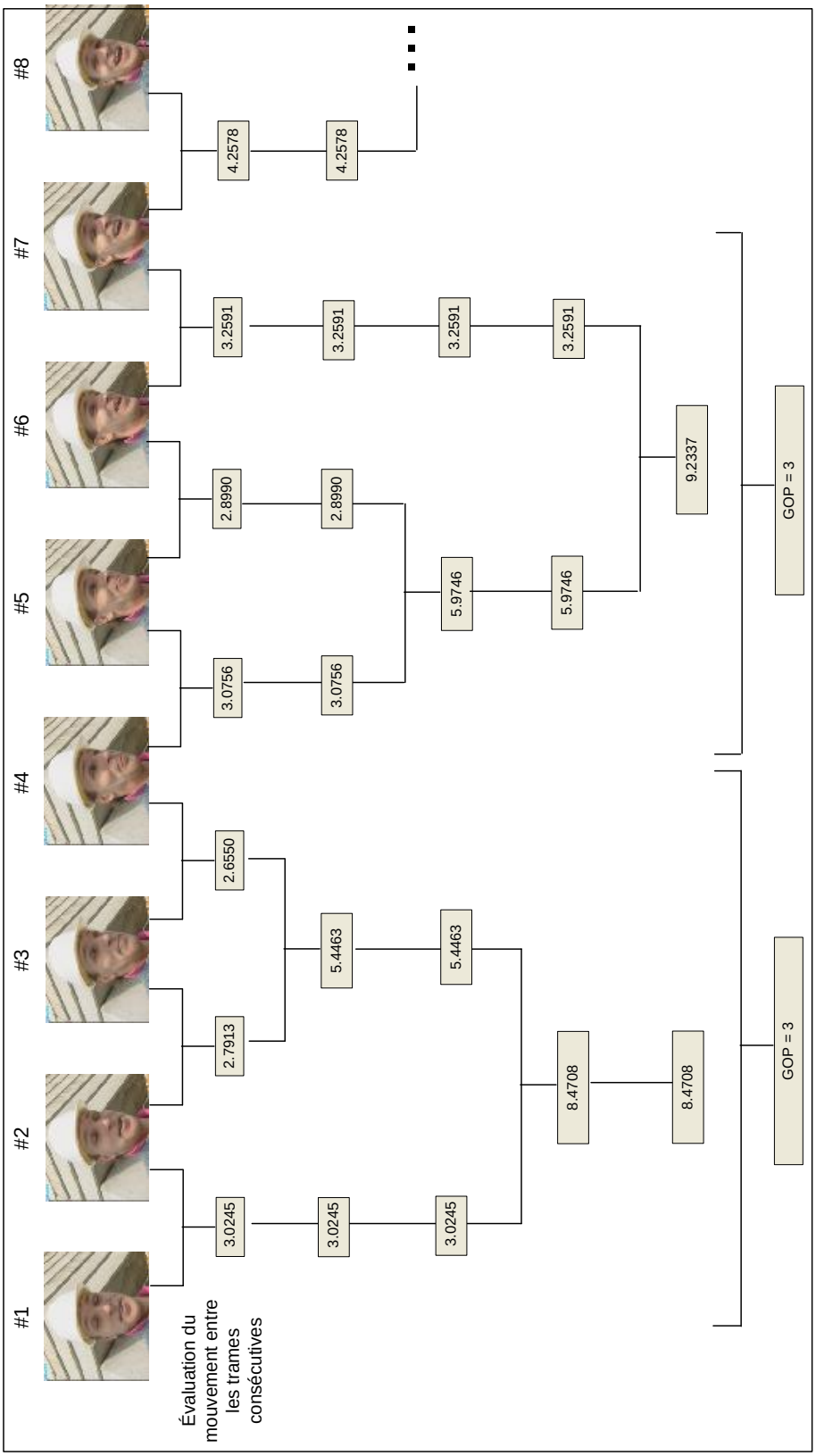

FIGURE 7.3: *Technique adaptative de contrôle de la taille des groupes d'images basée sur la classification hiérarchique 1.*

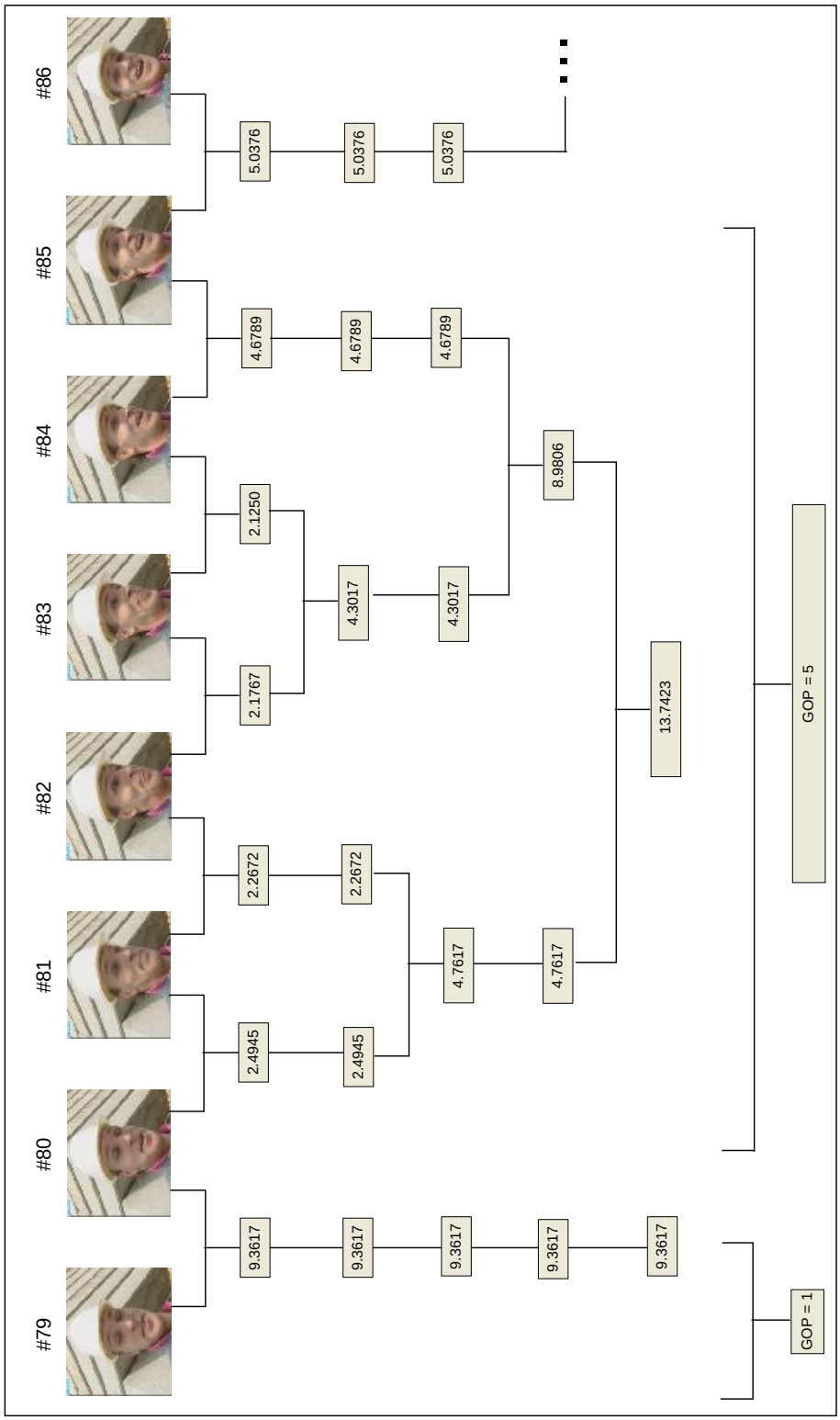

FIGURE 7.4: *Technique adaptative de contrôle de la taille des groupes d'images basée sur la classification hiérarchique 2.*

On décrit ici brièvement l'algorithme BiMESS afin de mieux comprendre les améliorations apportées par Discover. Les étapes de l'algorithme BiMESS tel qu'il a été proposé par Asensco et *al.* [13] sont récapitulées dans la figure 7.5. Cet algorithme fonctionne bien avec des séquences vidéo de 30 trames par seconde et pour des GOP=2. Dans ce cas, il y a une grande corrélation entre la trame à interpoler (la trame WZ) et les deux trames avoisinantes : une zone de recherche de taille $\pm$ 8 pixels est suffisante pour détecter le mouvement.

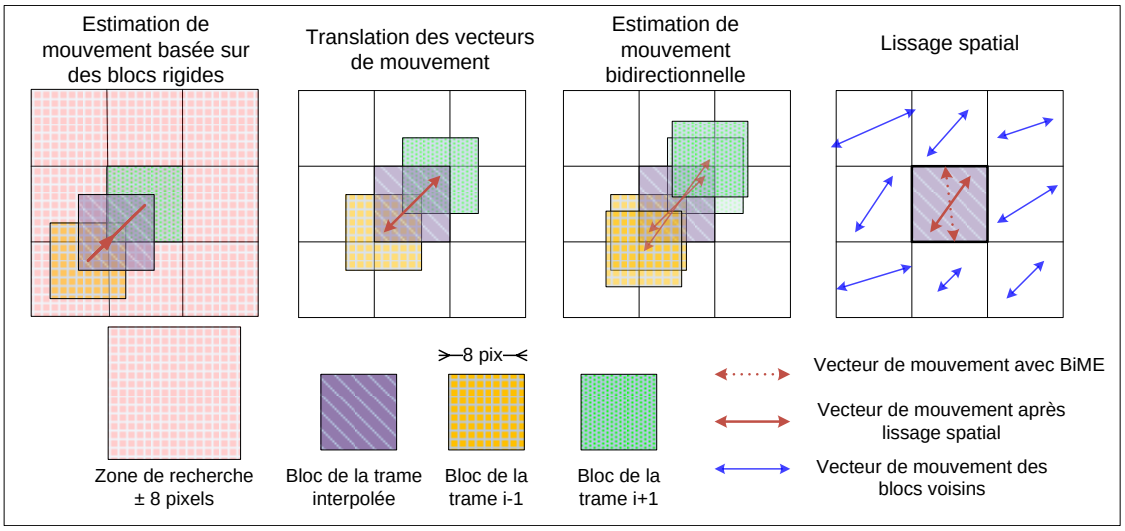

FIGURE 7.5: *Étapes de l'algorithme d'estimation de mouvement bidirectionnelle avec lissage spatial.*

Cependant pour un GOP élevé et pour une résolution temporelle faible, une telle zone de recherche est insuffisante pour capter la variation des blocs et l'augmentation de la zone de recherche risque de générer des vecteurs de mouvement bruités. Dans la section suivante, on expliquera la solution simple et efficace proposée par les concepteurs de Discover pour augmenter la zone de recherche sans pour autant résulter en des vecteurs bruités. De plus, on observe à la figure 7.5 que les vecteurs de mouvement bidirectionnels sont symétriques. La symétrie des vecteurs de mouvement ne peut être supposée que lorsque le GOP est puissance de 2, c'est-à-dire que lorsque la trame à interpoler se trouvera toujours au milieu des trames avoisinantes. Dans le contexte du contrôle adaptatif du groupe d'images, la taille de ce dernier (le GOP) varie le long de la vidéo en fonction de l'activité du mouvement. Ainsi, on peut avoir un GOP qui est une puissance de 2 comme on peut avoir un qui n'est pas une puissance de 2. On expliquera dans la section suivante les changements apportés à l'algorithme d'estimation de mouvement bidirectionnelle (BiME) pour le cas d'un GOP qui n'est pas puissance de 2.

7.4.2 Système d'interpolation de Discover

L'algorithme d'interpolation présenté à la figure 7.6 et proposé par les auteurs [61], a été adopté dans l'architecture de Discover. Le fonctionnement de chacun des blocs de cet interpolateur a été ajusté

pour créer une information latérale efficace en utilisant un GOP de taille arbitraire. On détaille ci-dessous les modifications apportées au niveau de chaque bloc. Plus de détails sur le fonctionnement du BiMESS ont été fournis dans le chapitre précédent.

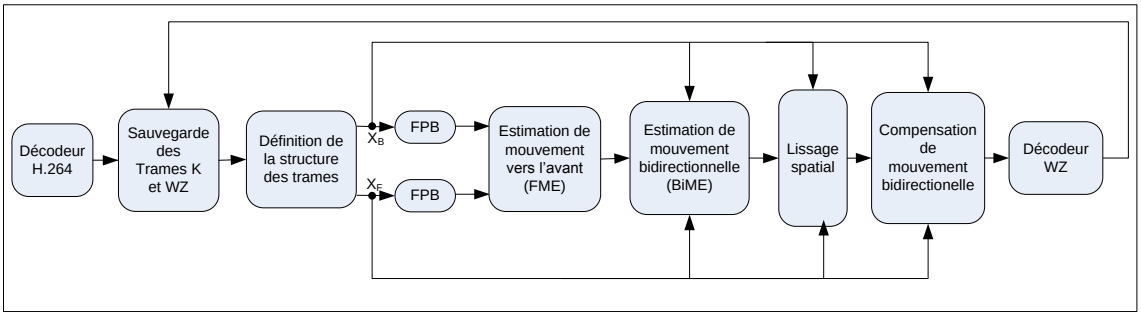

FIGURE 7.6: *Diagramme bloc de l'interpolation pour un GOP de taille flexible.*

Définition de la structure d'interpolation des trames

Comme le GOP varie de façon adaptative le long de la vidéo, la structure d'interpolation varie aussi. Pour un GOP=2, X_B et X_F sont les trames Intra adjacentes. Pour d'autres tailles du GOP plus élevées, la structure d'interpolation suit la même technique que celle proposée dans [63]. Cependant, cette technique ne fonctionne que pour un GOP de taille qui est une puissance de 2. La définition de la structure proposée par [61] fonctionne pour toute taille du GOP. On illustre à la figure 7.7 une structure de trames pour un GOP= 5. Les chiffres dans cette figure indiquent l'ordre du décodage. Les trames WZ décodées sont sauvegardées dans une mémoire tampon pour éventuellement servir à l'interpolation pour d'autres trames WZ de la même structure.

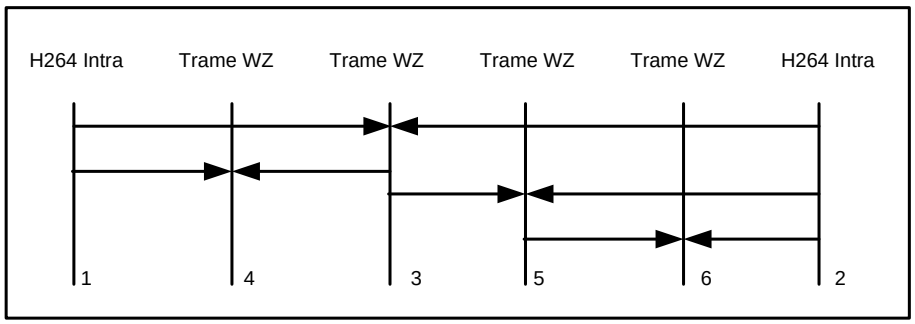

FIGURE 7.7: *Structure d'interpolation des trames pour un GOP=5.*

Estimation de mouvement vers l'avant

Le changement principal apporté pour le bloc d'estimation de mouvement vers l'avant est l'évaluation du critère de ressemblance entre un bloc de la trame X_B et un bloc de la trame X_F. Habituellement,

l'algorithme FME tend à trouver la meilleure correspondance entre un bloc de X_B et un bloc de X_F en utilisant une fonction de coût, CF, qui se résume à la moyenne des différences absolues (MAD) :

$$(v_x, v_y) = \arg\min_{d_x, d_y} CF(d_x, d_y), \quad d_x \in [-M, M], d_y \in [-M, M] \tag{7.12}$$

$$CF(d_x, d_y) = MAD(d_x, d_y) = \frac{1}{N} \sum_{(x,y)\in B} |X_F(x,y) + X_F(x+d_x, y+d_y)| \tag{7.13}$$

où M désigne la zone de recherche prenant la valeur ± 8 pixels, N est le nombre de pixels dans un bloc B de la trame X_F ($N = 8 \times 8$) et (v_x, v_y) est le vecteur de mouvement sélectionné. On a mentionné précédemment qu'une zone de recherche se limitant à ± 8 pixels n'est pas efficace si le GOP est élevé et la résolution temporelle est faible. Dans ce dernier cas, il est primordial d'agrandir la zone de recherche. Mais si on garde la même fonction de coût, l'algorithme de la FME risque de résulter en des vecteurs de mouvement bruités vu que la zone de recherche est grande. Une alternative possible pour augmenter la zone de recherche est d'ajouter un terme de pénalité à la fonction de coût pour obtenir ce qui suit :

$$CF(d_x, d_y) = MAD(d_x, d_y) \times \left(1 + K \times \sqrt{d_x^2 + d_y^2}\right) \tag{7.14}$$

où K est une constante de lissage qui contrôle la quantité de pénalisation des vecteurs de mouvement de larges valeurs. Les expérimentations effectuées dans [61] suggèrent une valeur de $K = 0.05$. Avec cette fonction de coût modifiée, l'augmentation de la zone de recherche de ± 8 à ± 32 pixels est possible. En effet, on démontre à la figure 7.8 que l'augmentation de la zone de recherche en gardant l'ancienne fonction de coût ($K = 0$) résulte bien en des vecteurs de mouvement bruités et de grande valeur. La fonction de coût modifiée régularise le champ de mouvement en favorisant les vecteurs de mouvement proches de l'origine.

Estimation bidirectionnelle de mouvement

Après avoir déterminé le champ de mouvement entre la trame X_B et X_F, les vecteurs de mouvement sont ajustés lors de l'estimation bidirectionnelle de mouvement (BiME). L'algorithme BiME présente 3 points importants :

1. **Estimation bidirectionnelle de mouvement BiME avec une variation hiérarchique de la taille du bloc (16×16 après 8×8) :** cette approche hiérarchique permet de détecter les vecteurs de mouvement rapides surtout pour les GOP de taille élevée, durant la première itération avec des blocs de 16×16. Ensuite, la seconde itération avec des blocs de 8×8 permet d'effectuer une BiME avec plus de précision. Cette approche est représentée à la figure 7.9.
2. **Adaptation de la zone de recherche :** Pour mieux exploiter la corrélation spatiale importante du champ de mouvement réel, la détermination de la zone de recherche adaptative d'un vecteur de mouvement donné inclut de l'information provenant des vecteurs de mouvement voisins. Cette technique est plus efficace que l'utilisation d'une zone de recherche fixe car elle restreint

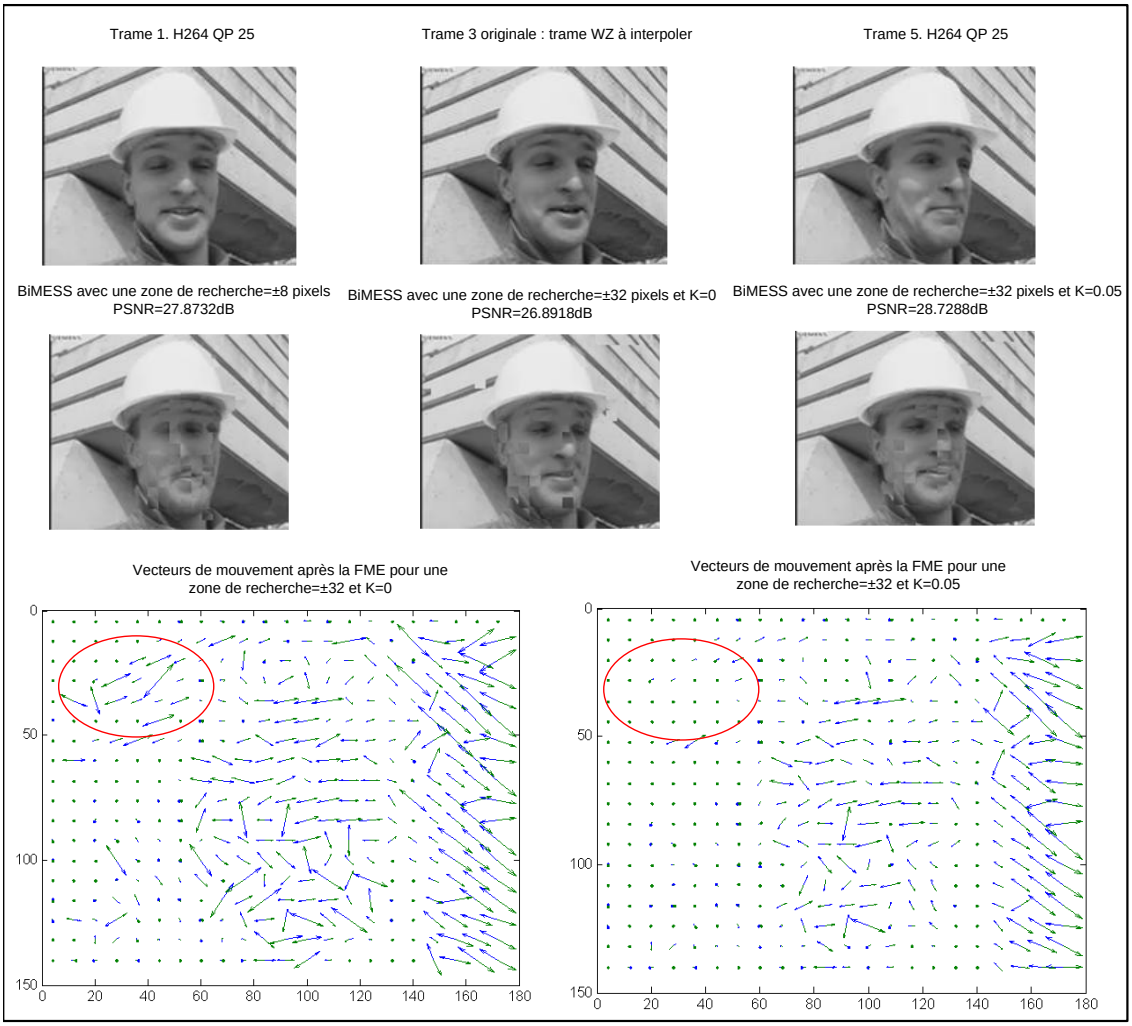

FIGURE 7.8: *Performance de l'interpolation en augmentant la zone de recherche avec l'ancienne fonction de coût, MAD ($K = 0$) et avec la fonction de coût modifié ($K = 0.05$).*

la trajectoire des vecteurs de mouvement à être en phase avec le champ de mouvement réel. La zone de recherche adaptative est déterminée comme suit :

$$\begin{aligned} x_H + N \leq d_x \leq x_B - N \\ y_G + N \leq d_y \leq y_D - N \end{aligned} \tag{7.15}$$

À la figure 7.10, on montre un exemple de détermination de la zone de recherche où N désigne la taille du bloc et x_H, x_B, y_G, y_D désignent les coordonnées des vecteurs de mouvement voisins.

3. **Calcul de la fonction de coût :** Comme la trame WZ n'est pas forcément à distance égale des trames X_B et X_F, les vecteurs de mouvement bidirectionnels ne sont pas toujours symétriques.

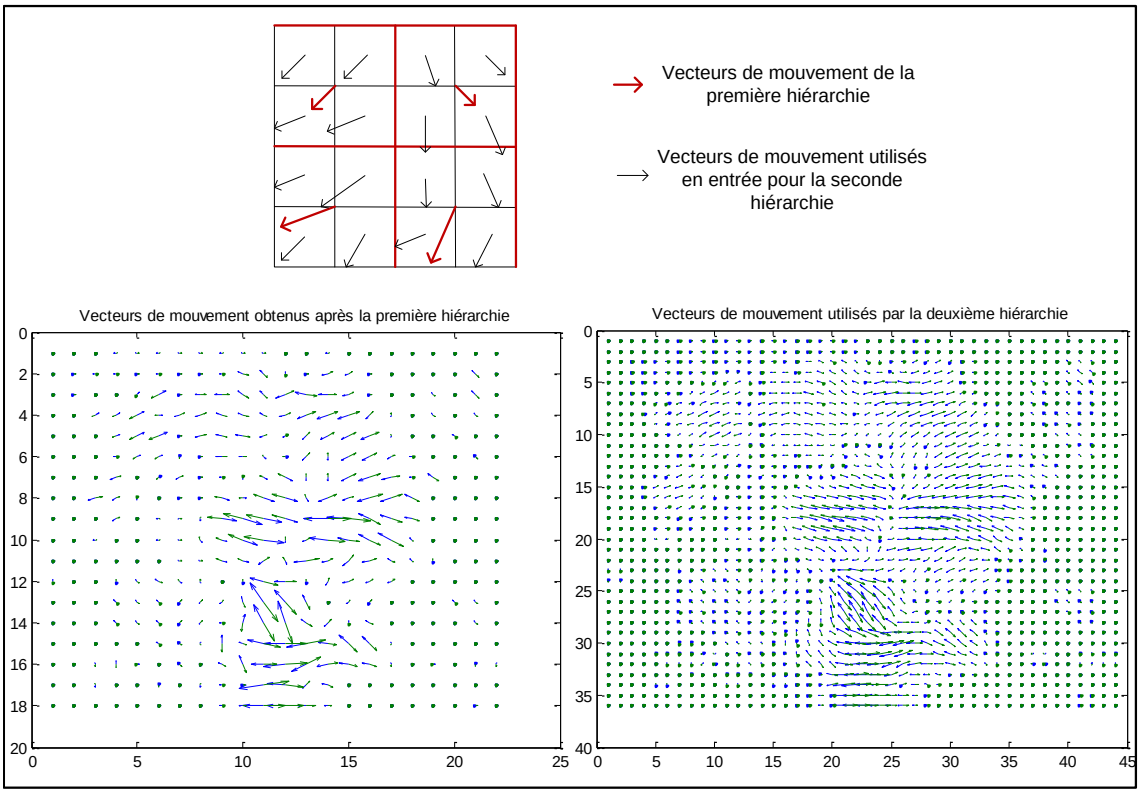

FIGURE 7.9: *Approche hiérarchique lors de la BiME en utilisant au départ une taille du bloc de 16×16, et ensuite de 8×8.*

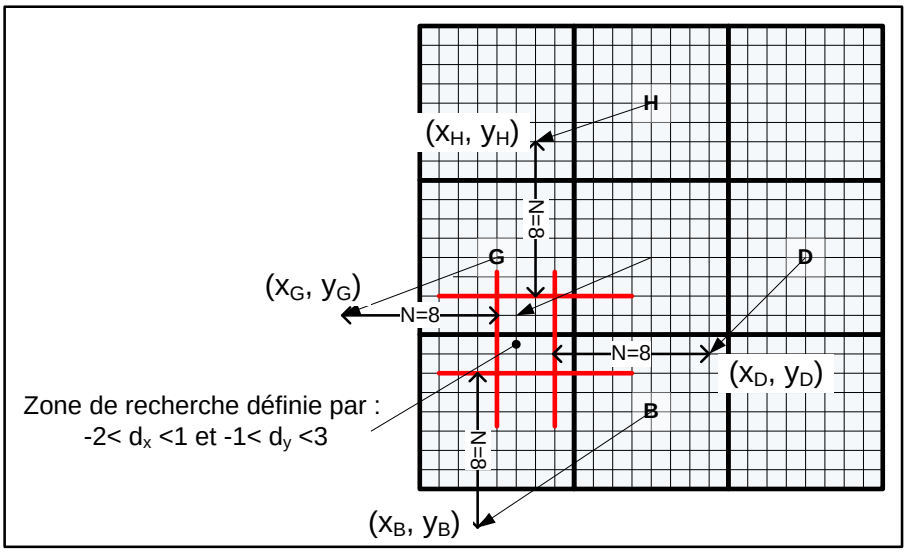

FIGURE 7.10: *Adaptation de la zone de recherche lors de l'estimation bidirectionnelle de mouvement (BiME).*

Le calcul de la moyenne des différences absolues se fait comme suit :

$$MAD\left(d_x,d_y\right)=\frac{1}{N\times N}\sum_{(x,y)\in B}\left|X_B\left(x-N_1d_x,y-N_1d_y\right)+X_F\left(x+N_2d_x,y+N_2d_y\right)\right| \tag{7.16}$$

où N_1 et N_2 désignent la distance temporelle entre la trame WZ et les trames X_B et X_F, respectivement. Pour un GOP $=3$, $N_1=1$ et $N_2=2$. Le vecteur de mouvement (d_x,d_y) est déduit à partir du vecteur de mouvement obtenu par FME en effectuant une mise à l'échelle adéquate en fonction de N_1 et N_2.

$$(d_x,d_y)_{\text{BiME}}=\frac{(d_x,d_y)_{\text{FME}}}{N_1+N_2} \tag{7.17}$$

Estimation de mouvement avec une précision d'un demi-pixel

Dans certains cas avec la compression vidéo conventionnelle H.264/AVC [1], lors de l'estimation de mouvement entre une trame inter et une trame de référence, le décalage entre les blocs a une résolution d'un demi-pixel (ou d'un quart de pixel). Cependant, les sous-échantillons aux positions des demi-pixels n'existent pas dans la trame de référence et on doit donc les créer en utilisant une interpolation à partir des pixels aux positions entières. Dans le contexte de la compression vidéo distribuée, le décodeur ne dispose pas de la trame WZ en question mais il dispose de deux trames adjacentes : l'estimation de mouvement bidirectionnelle est effectuée entre ces deux trames avoisinantes. Des vecteurs de mouvements avec une précision à l'échelle du demi-pixel et du quart de pixel permettent d'avoir une information latérale de meilleure qualité tel qu'énoncé dans [64]. Les concepteurs de Discover ont aussi adopté cette même idée en utilisant une précision de demi-pixel et une interpolation spatiale des trames adjacentes X_B et X_F. La génération des échantillons aux positions des demi-pixels utilise un filtre de Wiener à 6 échantillons tel que défini dans le standard H264/AVC [1]. Les coefficients de ce filtre sont donnés par :

$$\frac{(1,\ -5,\ -20,\ 20,\ -5,\ 1)}{32} \tag{7.18}$$

On présente à la figure 7.11, les étapes de génération des échantillons interpolés aux positions demi-pixels : (1) On commence par placer les pixels aux positions entières. (2) On interpole ensuite les sous-pixels adjacents aux pixels entiers. (3) Les sous-pixels restants et adjacents aux sous pixels générés à l'étape (2) sont alors calculés.

Il est à noter que l'estimation de mouvement vers l'avant (FME) est effectuée avec la précision d'un pixel entier seulement. La FME résulte en des vecteurs de mouvement candidats qui seront ajustés avec la précision du demi-pixel lors de l'estimation bidirectionnelle de mouvement (BiME). Si le vecteur de mouvement est un entier, la compensation de mouvement (MC) reprend des pixels aux positions entières (c.-à-d. des pixels de la trame originale avant interpolation spatiale). Si le vecteur de mouvement est avec une précision du demi-pixel, alors la compensation de mouvement utilise des pixels interpolés. Afin de décrire l'implémentation de la précision du demi-pixel dans l'architecture DVC, on schématise les diverses étapes décrites ci-dessus à la figure 7.12.

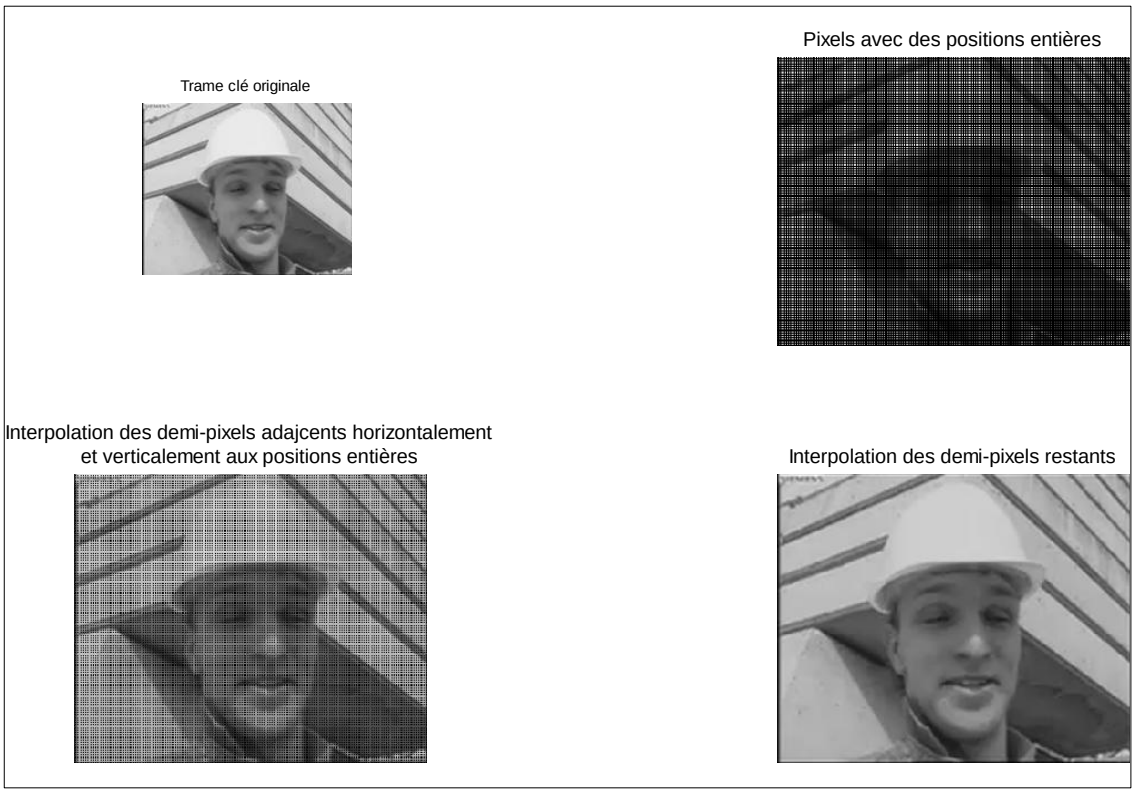

FIGURE 7.11: *Interpolation des pixels dans les positions demi-pixel avec le filtre de Wiener tel que défini dans le standard H264 [1].*

7.4.3 Évaluation de la qualité de l'information latérale générée par Discover

Nous avons vu dans cette section le grand intérêt porté par les développeurs de Discover pour augmenter la qualité de l'information latérale en déployant diverses techniques : (1) contrôle adaptatif du GOP, (2) utilisation d'une fonction de coût permettant d'agrandir la zone de recherche en évitant les vecteurs de mouvement bruités, (3) approche hiérarchique pour la variation de taille des blocs (16×16 puis 8×8) lors de la BiME, (4) adaptation de la zone de recherche lors de la BiME, et (5) augmentation de la précision des vecteurs de mouvement au demi-pixel. Dans cette sous-section, on essaie de quantifier l'apport de ces techniques en termes de PSNR de l'information latérale résultante en considérant la séquence `Foreman` (à 15 trames par seconde) téléchargée à partir du site d'évaluation de Discover [5]. On trace à la figure 7.13 le PSNR des 74 trames WZ de cette séquence pour : (1) l'utilisation d'une taille de GOP fixe de valeur GOP=2, (2) l'utilisation du mécanisme de contrôle adaptatif de la taille du GOP tout en gardant un GOP_{moyen}=2, et (3) l'utilisation d'un GOP flexible mais avec une précision des vecteurs de mouvement d'un demi-pixel.

On remarque bien que le contrôle adaptatif du GOP permet d'obtenir un gain de PSNR de 1.15 dB. Ce gain est tout à fait attendu vu que l'on place les trames H.264 Intra lorsque le mouvement dans la vidéo le nécessite. Ainsi, on évite d'effectuer une interpolation au niveau d'un groupe d'images de

faible corrélation. Ensuite on constate que l'augmentation de la précision des vecteurs de mouvement constitue un gain du PSNR supplémentaire de 0.36 dB.

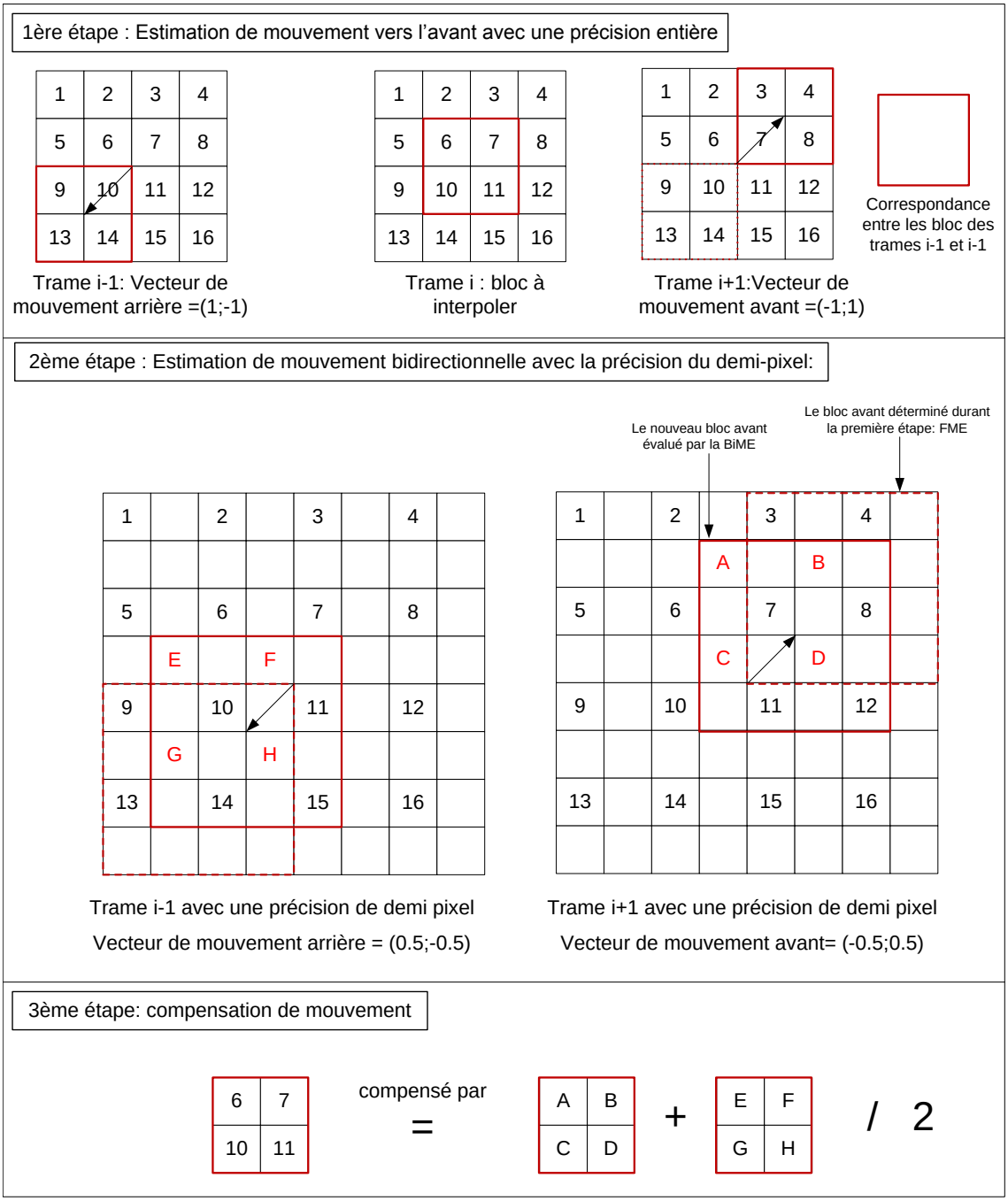

FIGURE 7.12: *Vecteurs de mouvement avec la précision d'un demi-pixel dans le contexte DVC.*

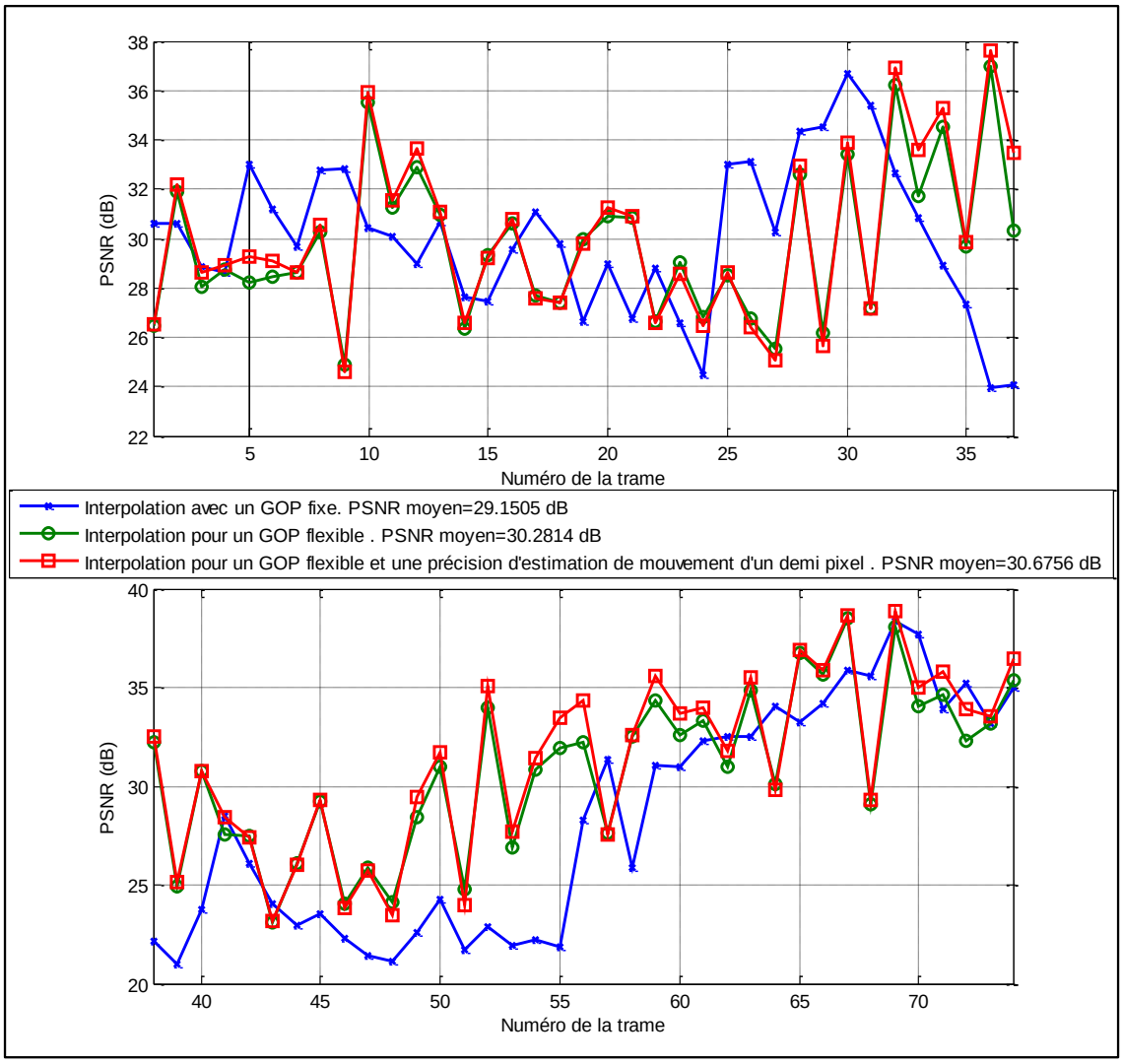

FIGURE 7.13: *Qualité de l'information latérale pour différentes architectures de codage vidéo distribué.*

7.5 Modélisation du canal virtuel

La différence entre l'information originale (des pixels ou des coefficients DCT) et l'information latérale dans les systèmes DVC est interprétée et modélisée comme un bruit additif introduit par un canal qui n'existe que virtuellement. Cette visualisation de la différence permet de concevoir un décodeur canal dont l'entrée est l'information latérale qui est supposée être une version bruitée de l'information originale. Le fonctionnement du décodeur canal nécessite la connaissance des statistiques du bruit, ce qui revient, dans le contexte DVC, à modéliser la corrélation entre les information originale et latérale. Les études expérimentales ont démontré que cette corrélation est bien caractérisée par une distribution Laplacienne. De plus, cette distribution est intéressante de point pratique vu qu'elle est complètement définie par un paramètre unique α et vu qu'elle est mathématiquement facile à manipuler. Depuis la naissance du codage vidéo distribué, la quasi-unanimité des chercheurs s'est résolue au choix de la Laplacienne comme modèle de corrélation.

Dans ce cadre de recherche, les études se sont dirigées principalement vers la manière d'évaluer de façon réaliste et précise le paramètre définissant cette distribution [59, 60]. Concernant l'aspect réaliste, l'évaluation du paramètre α doit se faire au niveau du décodeur sans connaissance de l'information originale. Il s'agit dans ce cas de la modélisation en ligne du canal virtuel. Il est tout de même parfois intéressant d'évaluer le gain de performances maximales que peut apporter une modélisation précise du canal virtuel. Pour cette raison, il y a un intérêt à étudier et à implémenter, de manière hors ligne, la modélisation du canal virtuel, c'est-à-dire en disposant de l'information originale au décodeur lors de l'évaluation de α.

Pour ce qui est de la précision du modèle de corrélation, on considère dans la littérature 4 niveaux de granularité dans le domaine des pixels : (1) Estimation de α au niveau de la séquence vidéo, (2) Estimation de α au niveau de la trame, (3) Estimation de α au niveau du bloc, et (4) Estimation de α au niveau du pixel. Dans le domaine des transformées, on ne considère que 3 niveaux de granularité : (1) Estimation de α au niveau de chaque bande DCT par séquence vidéo, (2) Estimation de α au niveau de chaque bande DCT par trame, et (3) Estimation de α au niveau de chaque coefficient DCT par trame. L'architecture de référence de Discover vise des performances optimales dans tous ses choix techniques, notamment en ne considérant que le domaine des transformées. Dans ce qui suit, on s'intéresse à la modélisation dans le domaine transformé uniquement.

7.5.1 Modélisation hors ligne au niveau de chaque bande DCT par trame

L'approche de modélisation au niveau de chaque bande DCT par trame, implique que pour chaque trame de la séquence vidéo on dispose de Nb_{bandes} Laplaciennes. Ici, Nb_{bandes} représente le nombre de bandes DCT à envoyer : il dépend de la matrice de quantification Q_i. Dans le domaine transformé, on modélise la corrélation entre les coefficients DCT originaux et les coefficients DCT latéraux. Comme le paramètre α_b pour chaque bande est recalculé de trame en trame, cette approche permet d'effectuer une certaine adaptation de la distribution Laplacienne le long de la séquence. Pour une modélisation

hors ligne, on suppose la connaissance de l'information originale au décodeur (ou de façon plus réaliste, on suppose que l'on contrevient au paradigme distribué et que l'on génère l'information latérale à l'encodeur) et on calcule le paramètre α_b en évaluant la variance σ_b^2 vu que pour une distribution Laplacienne on a la propriété suivante : $\alpha_b = \sqrt{\frac{2}{\sigma_b^2}}$

La procédure à suivre pour déterminer de façon hors ligne le paramètre α_b pour une bande donnée par trame se résume dans les étapes suivantes :

1. **Génération de la trame résiduelle :** On commence tout d'abord par calculer la trame résiduelle R qui consiste en la différence entre la trame WZ originale et la trame latérale SI :

$$R(x,y) = WZ(x,y) - SI(x,y) \tag{7.19}$$

   ici (x,y) désigne les coordonnées des pixels dans la trame.

2. **Transformation DCT de la trame R :** On applique ensuite la DCT 4×4 sur la trame résiduelle R pour obtenir la trame T des coefficients DCT :

$$T(u,v) = DCT[R(x,y)] \tag{7.20}$$

   ici (u,v) désigne les coordonnées des coefficients dans la trame T. Pour chaque bloc de 4×4 pixels de la trame R, on associe un bloc de coefficients DCT dans la trame T.

3. **Calcul de la variance de chaque bande de la trame T :** On commence par extraire les coefficients de chaque bande et par les regrouper dans T_b. On calcule ensuite pour chaque bande b la variance des coefficients formant T_b :

$$\sigma_b^2 = E\left[T_b^2\right] - \left(E\left[T_b\right]\right)^2 = \frac{1}{J}\sum_{j=1}^{J} T_b^2(j) - \left(\frac{1}{J}\sum_{j=1}^{J} T_b(j)\right)^2 \tag{7.21}$$

   ici J désigne la taille de la bande DCT : $J = \frac{25344}{16} = 1584$.

4. **Calcul du paramètre α_b pour la bande b des coefficients DCT :** On peut déduire finalement les Nb_{bandes} paramètres de la Laplacienne en utilisant :

$$\alpha_b = \sqrt{\frac{2}{\sigma_b^2}} \tag{7.22}$$

Ce processus hors ligne de calcul du paramètre α exploite temporellement la variation de la corrélation de trame en trame. On peut bénéficier davantage du processus hors ligne en exploitant aussi spatialement la variation de la corrélation.

7.5.2 Modélisation hors ligne au niveau de chaque coefficient DCT par trame

Dans la sous section précédente, on a évalué un paramètre α_b pour chaque bande DCT. On propose dans cette section une modélisation du canal virtuel à la granularité la plus fine possible, soit au niveau des coefficients DCT. Ceci revient à estimer $J = 1584$ paramètres par bande, permettant ainsi une précision inégalée lors de l'estimation des valeurs de vraisemblance canal pour le décodage turbo. Pour

ce niveau maximum de granularité, avec une estimation de la corrélation coefficient par coefficient, le paramètre α ne peut être déduit à partir de la variance, la variance d'un singleton étant égale à zéro. Il faut alors trouver une mesure alternative afin de déduire α et qui apporte la même signification de l'information apportée par la variance σ^2. La mesure utilisée est la trame résiduelle transformée $T(u,v)$: c'est la variable même dont on a calculé la variance à la sous-section précédente. La trame transformée $T(u,v)$ est une mesure efficace pour remplacer $var(T(u,v))$. Les étapes à suivre pour déterminer les paramètres α au niveau de chaque coefficient par trame sont les suivantes :

1. **Génération de la trame résiduelle :**

$$R(x,y) = WZ(x,y) - SI(x,y) \tag{7.23}$$

2. **Transformation DCT de la trame R :**

$$T(u,v) = DCT[R(x,y)] \tag{7.24}$$

3. **Calcul du paramètre α pour chaque coefficient DCT :**
   Le paramètre α de la Laplacienne est finalement déduit de $T(u,v)$ comme s'il s'agissait de la variance σ^2. Ainsi on obtient pour chaque coefficient de la trame résiduelle transformée un paramètre α comme suit :

$$\alpha = \begin{cases} \sqrt{\frac{2}{[T(u,v)]^2}}, & T(u,v) \geq 1 \\ \sqrt{2}, & T(u,v) < 1 \end{cases} \tag{7.25}$$

   ici la deuxième ligne de l'équation est utilisée pour assurer une stabilité numérique.

La modélisation hors ligne du bruit de corrélation (*Offline CNM*) est irréaliste, mais elle est étudiée ici à titre exploratoire pour appréhender les performances limites que l'on pourrait atteindre. Dans les deux sous-sections suivantes, on présente deux techniques pour la modélisation en ligne du bruit de corrélation (*Online CNM*) pour bénéficier d'un gain de performances dans un cadre réaliste.

7.5.3 Modélisation en ligne au niveau de chaque bande DCT par trame

Pour le passage de la technique CNM (*Correlation Noise Model*) hors ligne vers la technique CNM en ligne, il faut se mettre dans la situation réelle du décodage où l'on ne dispose pas de l'information originale et où l'on souhaite formuler une prévision sur la variance de la trame résiduelle $R(x,y)$ donnée par l'équation (7.19). La technique d'estimation de $R(x,y)$ présentée dans la littérature utilise la fiabilité des champs de mouvement estimés lors de la génération de la trame interpolée dont la transformée constitue l'information latérale (SI). La technique CNM en ligne au niveau de chaque bande DCT par trame suit les étapes suivantes :

1. **Génération de la trame résiduelle estimée :** L'estimation de la trame résiduelle se base sur deux versions des trames X_B et X_F après compensation de mouvement :

$$R(x,y) = \frac{X_F(x+dx_f, y+dy_f) - X_B(x+dx_b, y+dy_b)}{2} \tag{7.26}$$

ici $X_B(x+dx_b, y+dy_b)$ et $X_F(x+dx_f, y+dy_f)$ représentent les trames précédente et subséquente après compensation de mouvement par les vecteurs de mouvement (dx_b, dyb) et (dx_f, dyf), respectivement. La paire (x,y) représente les coordonnées spatiales du pixel dans la trame R. L'expression (7.26) constitue une estimation de l'expression de l'équation (7.19). La ressemblance entre les versions compensées de X_B et de X_F constitue un indicateur sur la qualité de l'information latérale et ainsi que sur la différence entre la trame originale et la trame latérale.

2. **Transformation DCT de la trame R :**

$$T(u,v) = DCT[R(x,y)] \tag{7.27}$$

3. **Calcul de la trame $|T|$** : À partir d'une série d'essais expérimentaux, il a été constaté que la valeur absolue de la trame T permet d'avoir une estimation de la variance plus proche de la valeur obtenue hors ligne donnée par l'équation [7.21].

4. **Calcul de la variance de chacune des bandes b de $|T|$:**

$$\sigma_b^2 = E\left[|T|_b^2\right] - \left(E[|T|_b]\right)^2 \tag{7.28}$$

5. **Estimation du paramètre α de la Laplacienne :**

$$\alpha_b = \sqrt{\frac{2}{\sigma_b^2}} \tag{7.29}$$

7.5.4 Modélisation en ligne au niveau de chaque coefficient DCT par trame

Pour la modélisation hors ligne au niveau des coefficients de la DCT, la variance a été remplacée directement par la trame résiduelle transformée $T(u,v)$. Pour la modélisation en ligne, on classifie les coefficients de $|T|$ en 2 groupes, à savoir, les *coefficients en phase* et les *coefficients hors phase*.

1. **Coefficients en phase (*inlier coefficients*) :** Ce sont les coefficients dont les valeurs s'approchent de la moyenne de la bande DCT $\mu_b = E[|T|_b]$. Un coefficient est considéré proche de la moyenne si la distance entre le coefficient et la moyenne est inférieure à la variance :

$$[D_b(u,v)]^2 < \sigma_b^2 \tag{7.30}$$

2. **Coefficients hors phase (*outlier coefficients*) :** Ce sont les coefficients dont les valeurs sont très différents de la moyenne de la bande DCT μ_b. Un coefficient est considéré éloigné de la moyenne si la distance entre le coefficient et la moyenne est supérieure à la variance :

$$[D_b(u,v)]^2 \geq \sigma_b^2 \tag{7.31}$$

La technique CNM en ligne reproduit les quatre premières étapes décrites à la sous-section précédente que l'on rappelle ici brièvement pour faciliter la lisibilité et enchaîner avec l'étape 5 :

1. **Génération de la trame résiduelle estimée :**

$$R(x,y) = \frac{X_F(x+dx_f, y+dy_f) - X_B(x+dx_b, y+dy_b)}{2} \tag{7.32}$$

2. **Transformation DCT de la trame R :**

$$T(u,v) = DCT[R(x,y)] \tag{7.33}$$

3. **Calcul de la trame $|T|$**

4. **Calcul de la variance de chacune des bandes b de $|T|$:**

$$\sigma_b^2 = E\left[|T|_b^2\right] - (E[|T|_b])^2 \tag{7.34}$$

5. **Calcul de la distance entre les coefficients de $|T|$ et sa moyenne :** Pour chaque coefficient à la position (u,v), on calcule la distance avec la moyenne :

$$D_b(u,v) = |T|_b(u,v) - \mu_b \tag{7.35}$$

6. **Estimation du paramètre α pour le coefficient (u,v) :**

$$\alpha_b(u,v) = \begin{cases} \sqrt{\frac{2}{[D_b(u,v)]^2}}, & D_b(u,v) \geq \sigma_b^2 \\ \sqrt{\frac{2}{\sigma_b^2}}, & D_b(u,v) < \sigma_b^2 \end{cases} \tag{7.36}$$

Dans l'équation [7.36], on distingue deux situations :

a) Le coefficient est éloigné de la moyenne. Dans ce cas, il est fort probable que ce coefficient appartienne à un bloc ou l'erreur résiduelle est importante.

b) Le coefficient est proche de la moyenne. Ceci indique que le coefficient est vraisemblablement non bruité.

7.5.5 Comparaison entre les différentes techniques CNM

On présente à la figure 7.14, les courbes de débit-distorsion obtenues pour les 4 techniques de modélisation du bruit de corrélation décrites ci-dessus. On remarque que la technique de corrélation hors ligne au niveau de la bande DCT par trame donne des performances, en terme de PSNR pour un débit donné, légèrement supérieures à la technique en ligne. En effet, en effectuant le calcul de la variance sur toute la trame résiduelle estimée, l'impact de l'erreur de l'estimation est moins ressenti. Cependant, l'écart est élevé entre les performances hors ligne et en ligne de la technique de corrélation au niveau du coefficient. Si l'estimation de la trame résiduelle en un coefficient est erronée, le paramètre α de la Laplacienne le sera aussi, affectant ainsi la validité des valeurs de vraisemblance canal utilisées par le décodeur. Dans l'architecture Discover, on utilise le modèle de corrélation en ligne au niveau des coefficients de la DCT car il est à la fois réaliste et permet d'avoir une granularité fine.

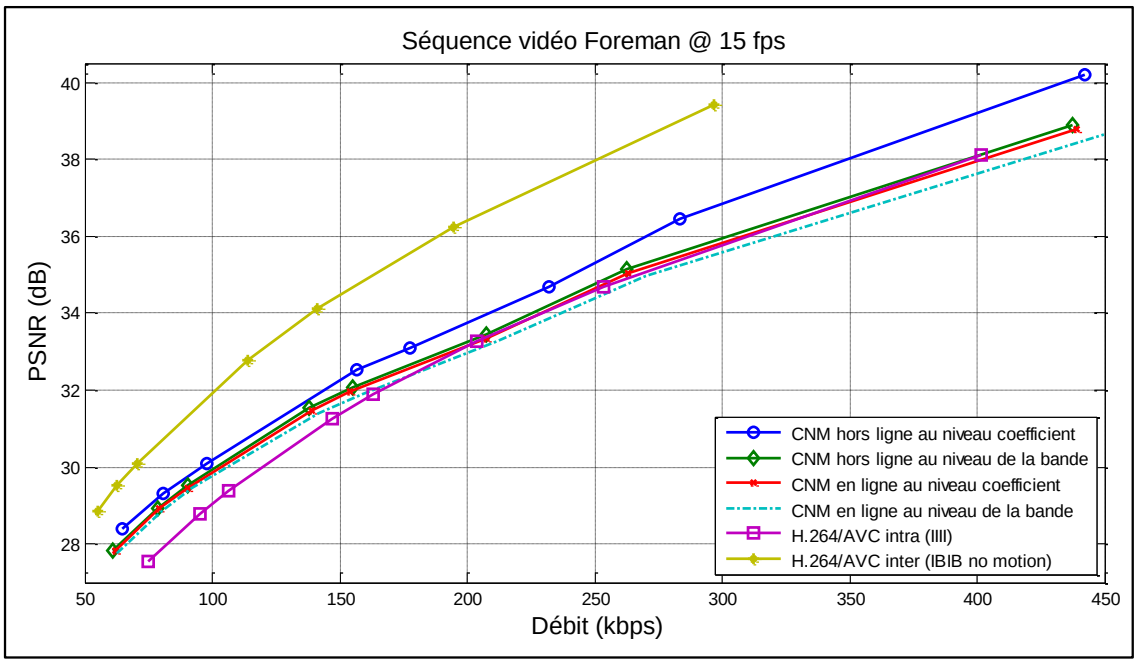

FIGURE 7.14: *Performance des différentes techniques de modélisation du canal virtuel.*

7.6 Estimation du débit minimal au niveau de l'encodeur

L'un des inconvénients majeurs de l'architecture DVC réside dans les délais considérables introduits lors du décodage turbo même si le décodeur est doté de ressources calculatoires importantes. L'encodeur envoie les bits de parité graduellement au décodeur et si ce dernier ne converge pas il informe à travers d'un canal de retour la nécessité d'un nombre supplémentaire de bits de parité. Ainsi l'algorithme BCJR log-MAP est exécuté en boucle jusqu'à convergence. La complexité de cet algorithme rend indispensable de minimiser les demandes en boucle des bits de parité supplémentaires. L'idée consiste à estimer au niveau de l'encodeur le nombre de bits de parité minimal, R_{min}, nécessaire pour la convergence du décodeur pour chaque plan de bits par bande. Ainsi on minimise la sollicitation du canal de retour et le nombre d'itérations du décodeur turbo.

7.6.1 Estimation de R_{min} en calculant l'entropie conditionnelle

Cette première méthode se base sur les travaux publiés dans [65]. Le taux minimal R_{min} est déterminé à partir de l'entropie conditionnelle $H(X|Y)$ entre les données originales WZ et l'information latérale SI. L'entropie conditionnelle à son tour est évaluée à partir de la probabilité de croisement $p_{cr} \equiv \Pr(X \neq Y)$ comme suit :

$$H(X|Y) = -p_{cr}\log_2 p_{cr} - (1-p_{cr})\log_2(1-p_{cr}) \tag{7.37}$$

La détermination de la probabilité de croisement nécessite la connaissance du modèle de corrélation entre l'information WZ et l'information latérale SI. Cette corrélation suit une loi Laplacienne dont le

paramètre est estimé au décodeur. Comme l'estimation de R_{min} se fait à l'encodeur, ce dernier doit avoir connaissance de ce paramètre qui est envoyé périodiquement par le canal de retour pour chaque bande de chaque trame. La probabilité de croisement est déterminée pour chaque plan de bits et elle correspond à la probabilité que le plan de bits x_{pb} soit différent du plan de bits $\hat{x}_{pb}$ estimé au décodeur en utilisant l'information latérale y et les plans de bits préalablement décodés $(x_{pb-1},\ \ldots, x_2,\ x_1)$:

$$\hat{x}_{pb} = \arg\max_{i=0,1} \Pr\left(x_{pb} = i \,\middle|\, y,\ x_{pb-1},\ \ldots, x_2,\ x_1\right) \tag{7.38}$$

où $\Pr\left(x_{pb} = i \,\middle|\, y,\ x_{pb-1},\ \ldots, x_2,\ x_1\right)$ désigne la probabilité à postériori de l'événement $x_{pb} = i$. Un exemple de calcul de $\hat{x}_{pb}$ est donné à la figure 7.15.

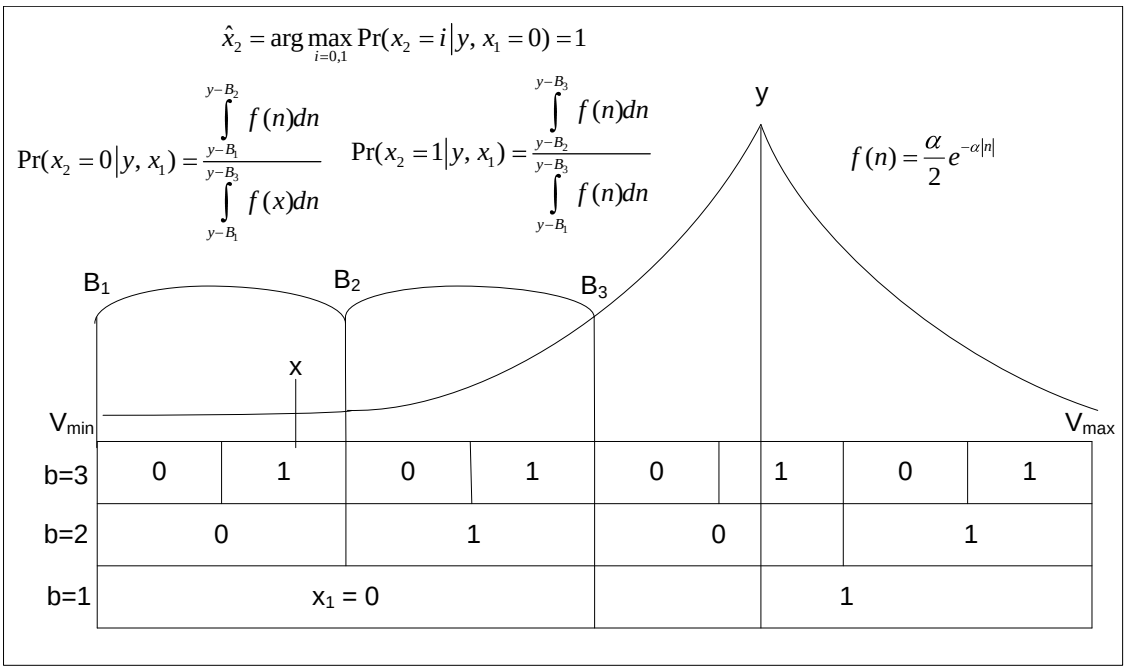

FIGURE 7.15: *Calcul de $\hat{x}_{pb}$ en fonction des probabilités conditionnelles sachant l'information latérale y et le bit précédent x_1.*

Le calcul de la probabilité de croisement se fait au niveau de l'encodeur qui ne dispose pas de l'information latérale. Cependant on remarque à partir de l'exemple de la figure 7.15 que le calcul nécessite la connaissance de y. L'idée consiste alors à calculer la moyenne de $\Pr\left(x_{pb} \neq \hat{x}_{pb}\right)$ sur toutes les valeurs possibles de y. Finalement on fait la moyenne sur toutes les valeurs des coefficients originaux de la bande WZ et on obtient ainsi :

$$p_{cr} = \frac{1}{N} \sum_{x \in WZ} \left[\int_{V_{\min}}^{V_{\max}} \Pr\left(x_{pb} \neq \hat{x}_{pb}\right) \frac{\alpha}{2} e^{-\alpha|y-x|} dy \right] \tag{7.39}$$

ici N désigne la longueur de la bande des coefficients. Pour le calcul de $\Pr(x_{pb} \neq \hat{x}_{pb})$, on se base sur l'exemple de la figure 7.15 et on a dans ce cas :

$$\Pr(x_2 \neq \hat{x}_2) = \frac{\int_{y-B_1}^{y-B_2} f(n)dn}{\int_{y-B_1}^{y-B_3} f(n)dn} = \frac{F(y-B_2) - F(y-B_1)}{F(y-B_3) - F(y-B_1)} \tag{7.40}$$

où $F(n)$ désigne la fonction de distribution cumulative (CDF) relative à la Laplacienne. L'utilisation de la CDF permet d'éviter d'effectuer une intégration. La fonction de distribution $F(n)$ pour une Laplacienne est donnée par :

$$F(n) = 0.5\left(1 + sign(n) - sign(n)\, e^{-\alpha|n|}\right) \tag{7.41}$$

Ainsi la détermination de la probabilité de croisement p_{cr} revient à calculer une moyenne de N intégrales relativement complexes. Ceci peut constituer une charge calculatoire considérable pour un encodeur censé être léger dans le paradigme de codage vidéo distribué. Pour chaque bande DCT de chaque trame le calcul du p_{cr} revient à calculer :

$$p_{cr} = \frac{1}{N} \sum_{x \in WZ} \left[\int_{V_{\min}}^{V_{\max}} \frac{F(y-B_2) - F(y-B_1)}{F(y-B_3) - F(y-B_1)} \frac{\alpha}{2} e^{-\alpha|y-x|} dy \right] \tag{7.42}$$

7.6.2 Impact de l'estimation de R_{min} au niveau de l'encodeur

L'estimation du débit minimal au niveau de l'encodeur a pour but principal de réduire le nombre d'utilisations du canal de retour. Ainsi pour évaluer les performances de cet estimateur on propose de rapporter dans les résultats de simulation le pourcentage de réduction de l'utilisation du canal de retour. Cependant ce pourcentage de réduction n'est pas suffisant pour évaluer l'efficacité de l'estimateur. En effet dans certain cas le nombre de bits estimé surpasse le nombre de bits nécessaires pour la convergence du turbo code. Donc on utilise un nombre de bits supérieur à ce qui est nécessaire entraînant une augmentation du débit. En définitive, on propose d'évaluer cet estimateur par le biais de deux indicateurs :

1. **Réduction de l'utilisation du canal de retour :** On compare le nombre des demandes via le canal de retour effectuées avec et sans estimation du débit minimal. La réduction des demandes de retour implique moins d'itérations et un décodage plus léger.
2. **Augmentation du débit engendrée :** Avec l'estimation de R_{min}, un nombre initial de bits de parité est envoyé d'un coup. Ce nombre peut dépasser le nombre nécessaire pour la convergence du décodeur turbo engendrant ainsi une augmentation du débit.

Les performances de l'estimateur du débit minimal pour 4 séquences vidéo différentes sont présentées aux figures 7.16 et 7.17. Dans ces histogrammes, on affiche le nombre moyen d'utilisations du canal de

	Pourcentage de réduction de l'utilisation du canal de retour	Pourcentage de l'augmentation du débit du codec Wyner-Ziv
`Foreman`	52.0702%	0.71229%
`Coastguard`	63.6154%	1.195%
`Hall`	76.7928%	16.933%
`Soccer`	51.1345%	0.23539%

Tableau 7.2: *Performances de l'estimateur du débit au niveau de l'encodeur.*

retour pour chaque plan de bits de chaque bande DCT avec et sans estimation de R_{min} pour la matrice de quantification de haut débit $Q_i = 8$. On évalue également pour chaque séquence le pourcentage de réduction de l'utilisation du canal de retour et le pourcentage de l'augmentation du débit du codec Wyner-Ziv. Ces pourcentages sont récapitulés dans le tableau 7.2.

D'après ce tableau récapitulatif on remarque que l'on parvient à réduire l'utilisation du canal de retour et la complexité calculatoire de l'algorithme de décodage turbo de 60.903% (moyenne pour les 4 séquences vidéos). Le prix qui en découle consiste en l'augmentation du débit de 4.7689%. En effet dans certains cas le débit minimal estimé dépasse le débit nécessaire.

Un meilleur estimateur devrait réduire l'utilisation du canal de retour sans toutefois engendrer une augmentation importante du débit. On propose ici certaines directions qui peuvent être considérées afin de raffiner la précision de cet estimateur. Après l'envoi des bits de parité relatifs à un plan de bits donné il y a deux situations qui se présentent. La première situation survient lorsque les bits de parité estimés et envoyés permettent au décodeur turbo de converger directement sans aucune demande de bits de parité supplémentaires. Il est fort probable dans cette situation que le nombre de bits estimé dépasse le nombre nécessaire. Pour le plan de bits suivant, le nombre de bits de parité estimé risque aussi de dépasser le nombre de bits nécessaires. Ainsi l'idée consiste à réduire ce nombre estimé par un certain facteur. La deuxième situation provient lorsque le nombre de bits estimé est inférieur au nombre de bits de parité nécessaire. Un certain nombre de demandes de bits de parité supplémentaire est alors effectué par le décodeur afin de converger. En fonction de ce nombre de demandes, le nombre estimé pour le plan de bit suivant peut être pondéré. Cette technique est principalement empirique et la pondération du nombre de bits estimé d'un plan de bits à un autre se baserait bien sur une étude expérimentale.

7.7 Résultats de simulation

Dans cette section, on présente les résultats des simulations effectuées, en rapportant les courbes de performance RD (*Rate-Distorsion*). Pour valider cette phase d'implémentation, on présente une comparaison entre les résultats obtenus avec les résultats fournis par Discover. Les simulations considèrent différentes valeurs pour le groupe d'images *GOP*=2, 4 ou 8. De plus, les performances du système DVC sont comparées avec deux standards de codage vidéo partageant une même propriété importante en terme de complexité de l'encodeur : la tâche d'estimation de mouvement n'est pas supportée par

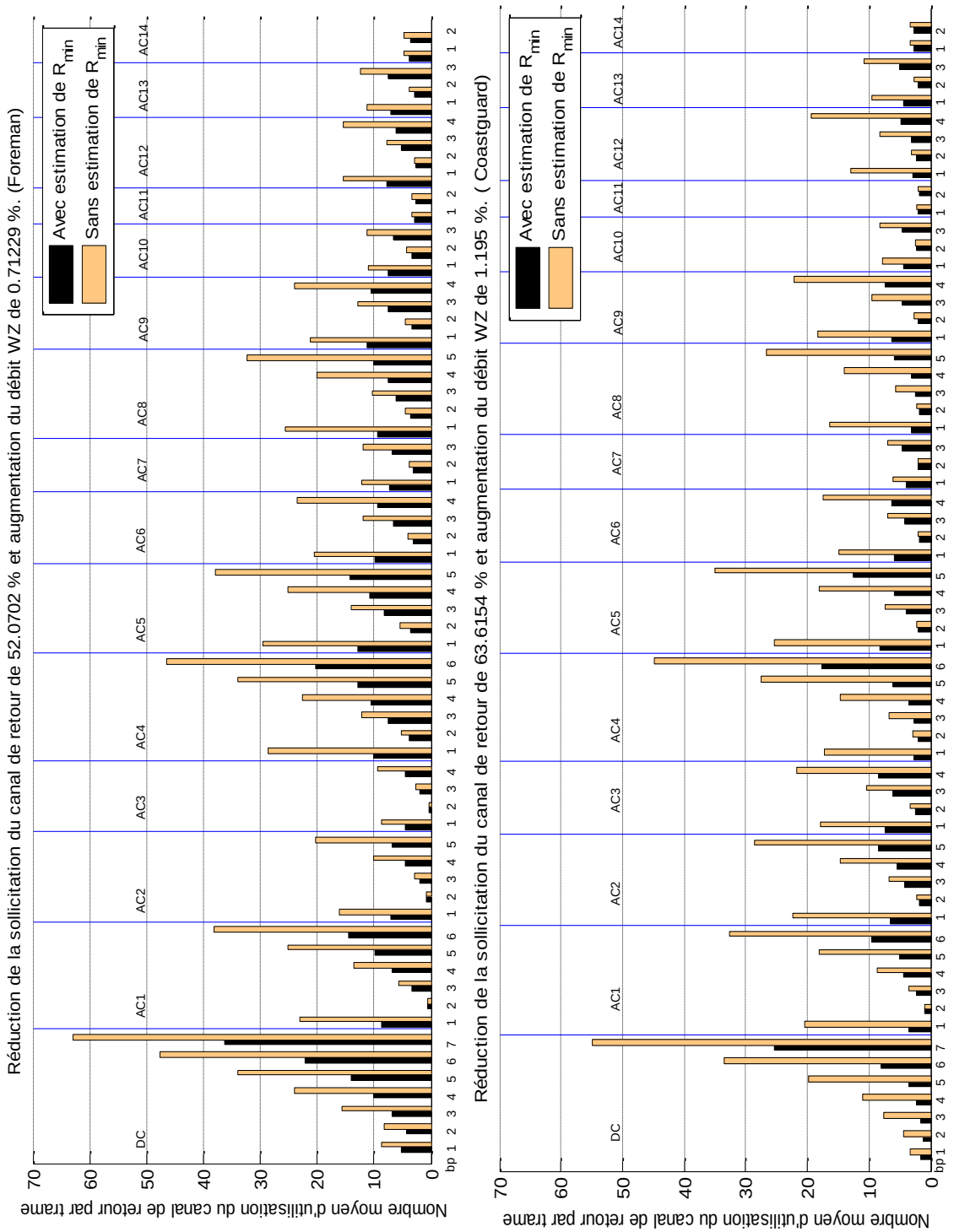

FIGURE 7.16: *Nombre d'utilisations du canal de retour pour chaque plan de bits de chaque coefficient DCT pour les séquences vidéos* `Foreman` *et* `Coastguard`.

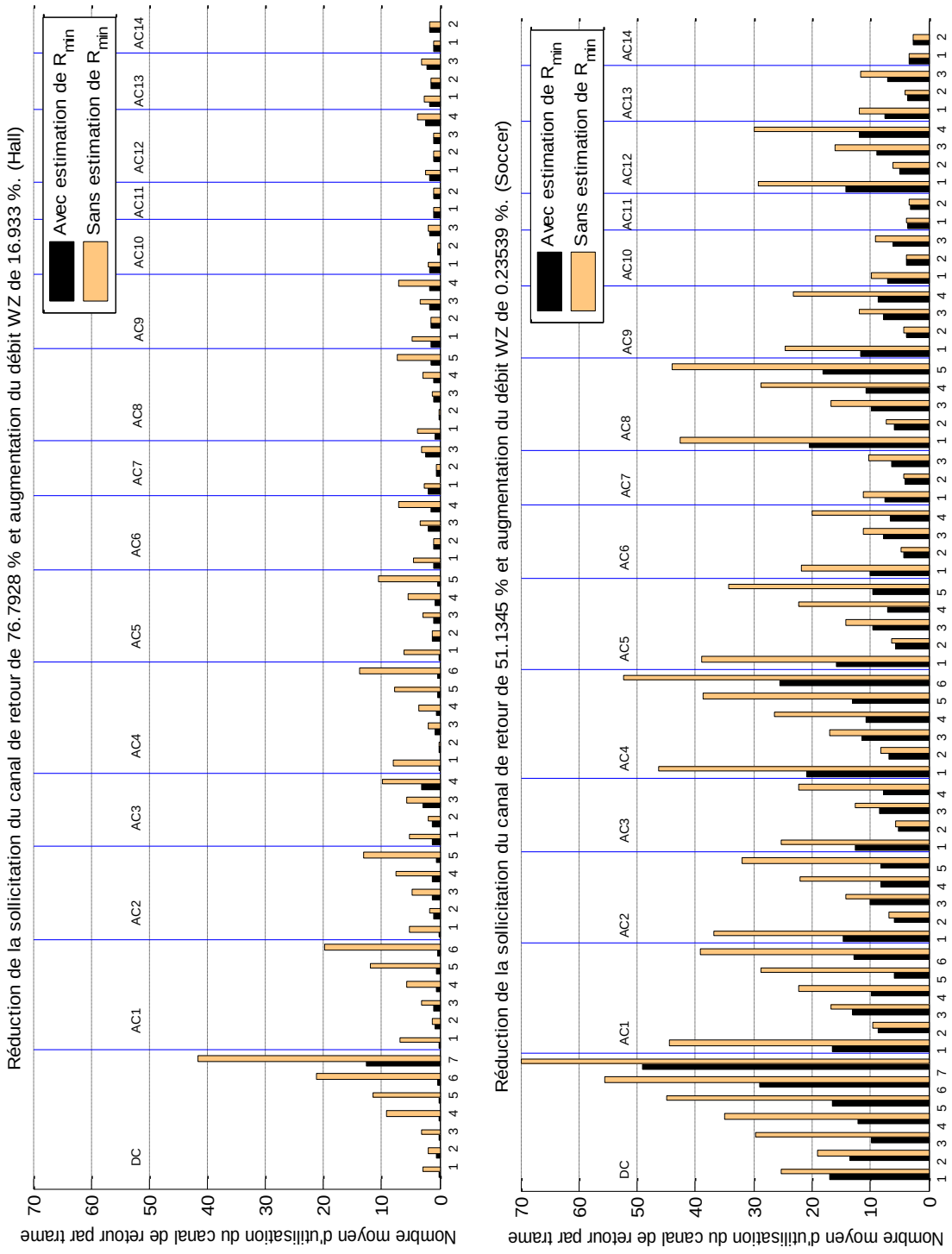

FIGURE 7.17: *Nombre d'utilisations du canal de retour pour chaque plan de bits de chaque coefficient DCT pour les séquences vidéos* `Hall` *et* `Soccer`.

l'encodeur. Ces deux standards sont les standards *H.264/AVC Intra* et *H.264/AVC Inter No motion* :

1. **H.264/AVC Intra [66] :** Ce standard de compression utilise le profil *Main* de la norme H.264/AVC sans aucune exploitation de la redondance temporelle. Chaque trame est ainsi codée indépendamment. Cependant, on note que le H.264/AVC Intra exploite très efficacement la corrélation spatiale avec plusieurs modes Intra 4×4 et 16×16. Aucune technique d'exploitation spatiale n'a été considérée pour le système Discover.
2. **H.264/AVC Inter No motion [66] :** Ce standard de compression utilise aussi le profil *Main* H.264/AVC avec une exploitation de la redondance temporelle selon une structure IBIB. Cependant, pour maintenir une complexité de codage raisonnable, aucune tâche d'estimation de mouvement n'est exécutée à l'encodeur.

7.7.1 Validation de la phase d'implémentation

On présente aux figures 7.18, 7.19 et 7.20, les performances du codec DVC pour un groupe d'images GOP = 2, 4 et 8 respectivement. On interpose les résultats fournis par le logiciel Discover aux résultats obtenus par simulation afin de valider la phase d'implémentation. Les simulations effectuées reprennent point par point les différentes techniques proposées dans Discover et détaillés le long de ce chapitre. A titre de rappel, Discover est le seul système de référence (benchmark) disponible à la communauté DVC fournissant les meilleures caractéristiques de point de vue performance et de point de vue pratique. L'aspect pratique, ici, renvoie au fait qu'on ne se base sur aucune connaissance des données originales à l'encodeur pour l'estimation du paramètre de la Laplacienne ou pour décider de la convergence du décodage turbo.

Les simulations effectuées fournissent des performances quasiment similaires à celle de Discover. Il est à noter que lors de l'implémentation des différentes techniques, certains détails sont parfois absents ou vagues dans la documentation relative à Discover (les articles et les livrables publics). Dans ces cas, on a tenté d'effectuer les choix qui nous semblaient les plus pertinents. L'implémentation des diverses techniques de Discover nous a permis non seulement de comprendre de plus près le fonctionnement de toute l'architecture mais aussi de disposer d'un codec compétitif afin d'incorporer et de tester de nouvelles techniques.

7.7.2 Comparaison entre les architectures DVC avec différentes valeurs de groupe d'images GOP

On présente à la figure 7.21, une comparaison des performances du codec DVC avec un groupe d'image GOP =2 (IW-IW), 4 (IWWW-IWWW) et 8 (IWWWWWWW-IWWWWWWW). La trame "W" dénote la trame Wyner-Ziv qui se limite à envoyer les bits de parité générés par le turbo code pour corriger la version interpolée servant d'information latérale au décodeur. La génération de ces bits de parité ne demande pas une grande complexité calculatoire à l'encodeur. Ceci consiste une caractéristique favorable au paradigme DVC stipulant le transfert de la complexité de la compression vidéo de l'encodeur de faible puissance vers le décodeur. La trame "I" est la trame intra envoyée par l'encodeur

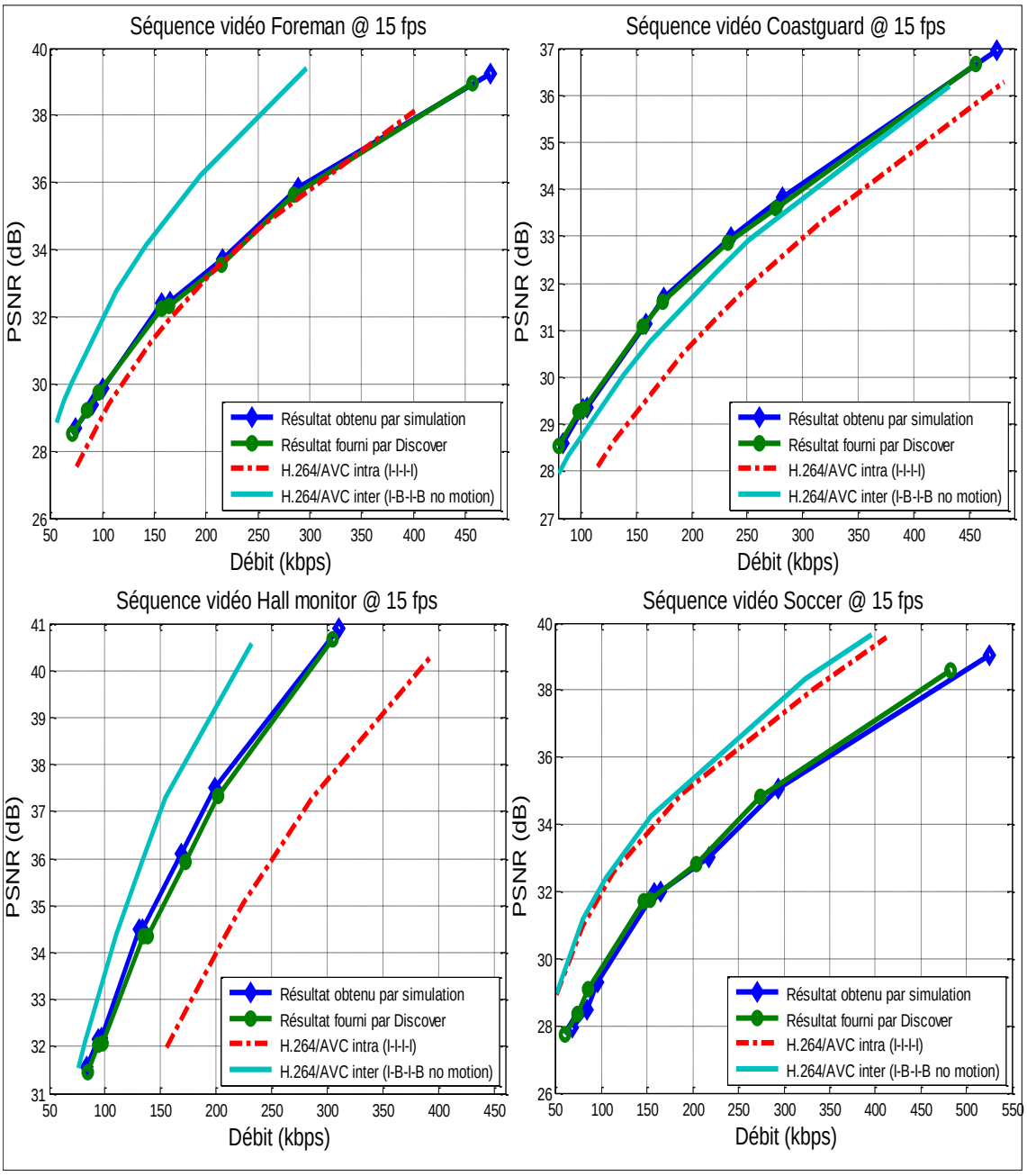

FIGURE 7.18: *Comparaison des performances de l'architecture DVC et les standards de compression vidéo conventionnelle pour un groupe d'images GOP =2.*

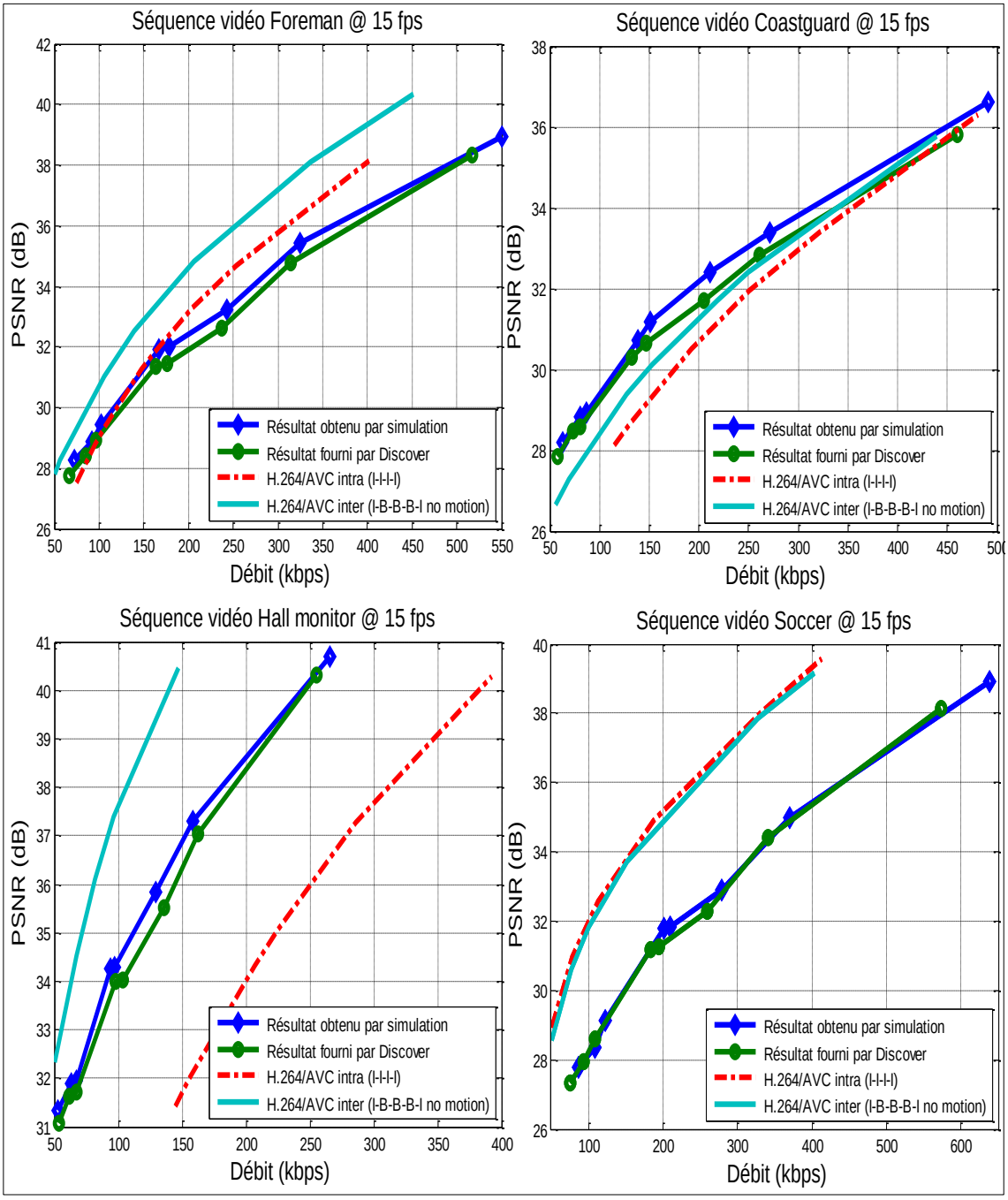

FIGURE 7.19: *Comparaison des performances de l'architecture DVC et les standards de compression vidéo conventionnelle pour un groupe d'images GOP =4.*

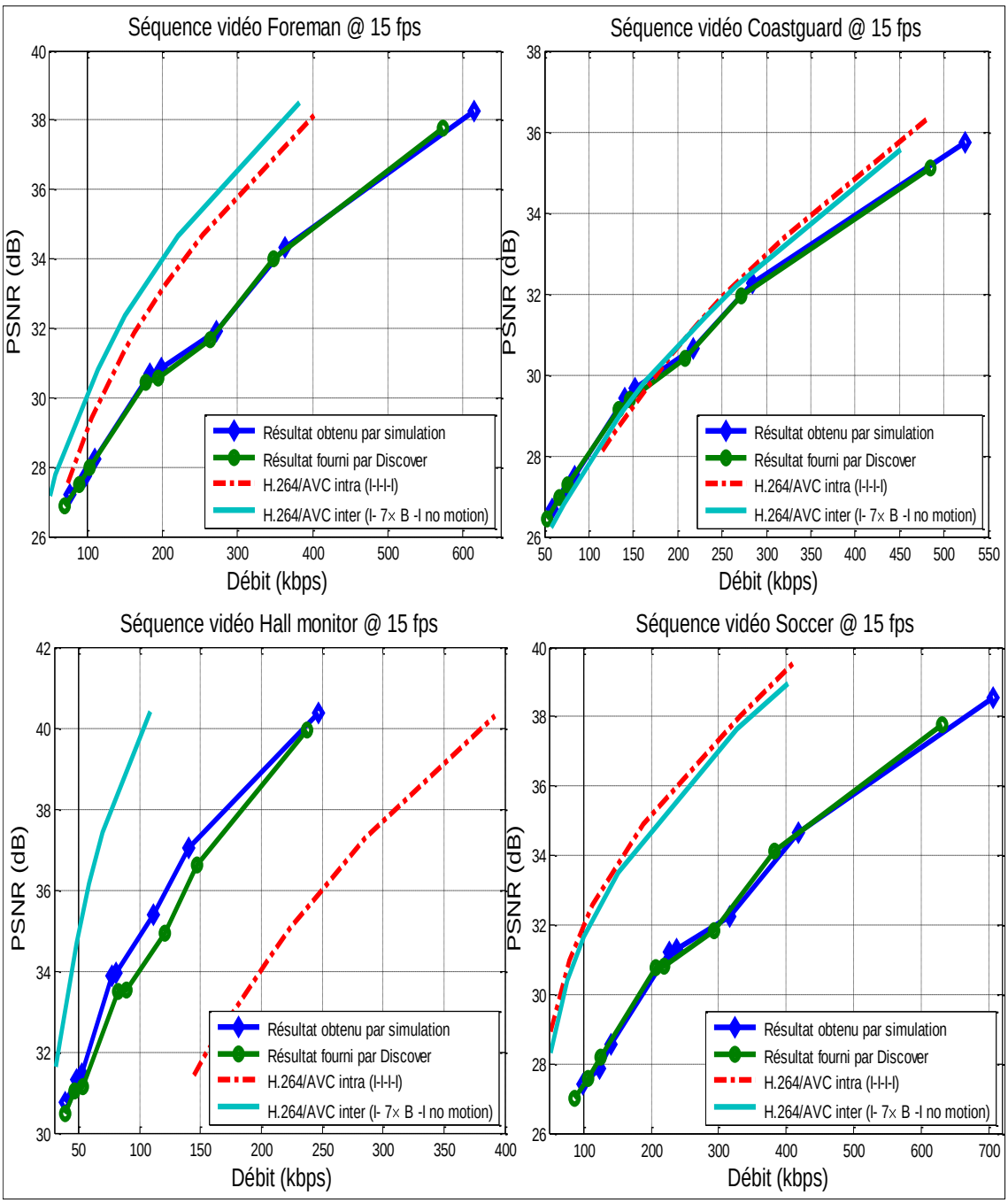

FIGURE 7.20: *Comparaison des performances de l'architecture DVC et les standards de compression vidéo conventionnelle pour un groupe d'images GOP =8.*

vidéo conventionnel H264/AVC en mode intra et engendrant une certaine complexité calculatoire à l'encodeur. En considération du paradigme DVC, il est souhaitable de garder un nombre aussi faible que possible des trames intras "I" diminuant au maximum la complexité de l'encodeur. Cependant la réduction de trames intras se répercute sur les performances du générateur de la trame interpolée qui opère avec des trames de référence avant et après de plus en plus éloignées temporellement.

D'après la figure 7.21, on remarque que pour les séquences rapides (surtout pour le cas de la séquence `Soccer`) l'augmentation du GOP entraine un certaine dégradation des performances. Pour les séquences rapides, l'interpolation nécessite des trames clés rapprochées temporellement afin de fournir une information latérale de bonne qualité. Cependant, pour la séquence vidéo `Hall`, qui est une séquence lente, l'augmentation du GOP entraine une amélioration des performances. Pour une telle séquence, il n'est pas nécessaire de dépenser énormément en terme de débit pour l'envoi d'un grand nombre de trames clés. On envoie dans ce cas des trames WZ ne nécessitant pas un débit important car le bloc d'interpolation opère avec des trames clés avant et après ayant une forte corrélation avec la trame WZ.

7.7.3 Comparaison entre les architectures avec un GOP fixe et un GOP flexible

On a présenté précédemment dans ce chapitre une nouvelle technique permettant de placer les trames intras I de façon flexible et adaptative selon la rapidité du mouvement dans la vidéo. Lorsqu'il y a une variation temporelle importante l'encodeur transmet une trame intra et lorsque la variation est lente il transmet une trame Wyner-Ziv. Lorsque le mouvement est lent le bloc d'interpolation est plus susceptible de fournir une information latérale de bonne qualité. Il faut alors détecter les phases lentes dans la vidéo pour placer les trames WZ. On a vu précédemment (voir la figure 7.13) que l'utilisation d'un GOP flexible permet une amélioration de l'information latérale de 1.15 dB pour la séquence `Foreman`. On présente à la figure 7.22 une comparaison des performances des architectures DVC avec un GOP$=2$ fixe et flexible. Pour une comparaison équitable il faut considérer dans les deux situations le même nombre de trames Intras. L'utilisation d'un groupe d'images flexible permet de fournir une certaine amélioration comme on le voit pour les séquences `Foreman` et `soccer`. Pour les séquences vidéo plus ou moins lente, `Coastguard` et `Hall monitor`, l'utilisation d'un GOP flexible n'affiche pas une amélioration des performances par rapport au GOP fixe.

7.8 Conclusion

Dans ce chapitre, on présente une étude de l'architecture de codage vidéo distribué Discover. Discover a regroupé une panoplie de techniques pour assurer la compétitivité des performances et pour soulever les problèmes d'ordre pratique. C'est ce qui a valu à Discover d'être le premier standard DVC pouvant servir de système de référence (*benchmark*) pour guider les efforts de recherche en codage vidéo distribué. Ce travail de ré-implémentation est important, premièrement pour comprendre les diverses techniques de Discover et deuxièmement, pour disposer d'un codec de référence auquel on peut in-

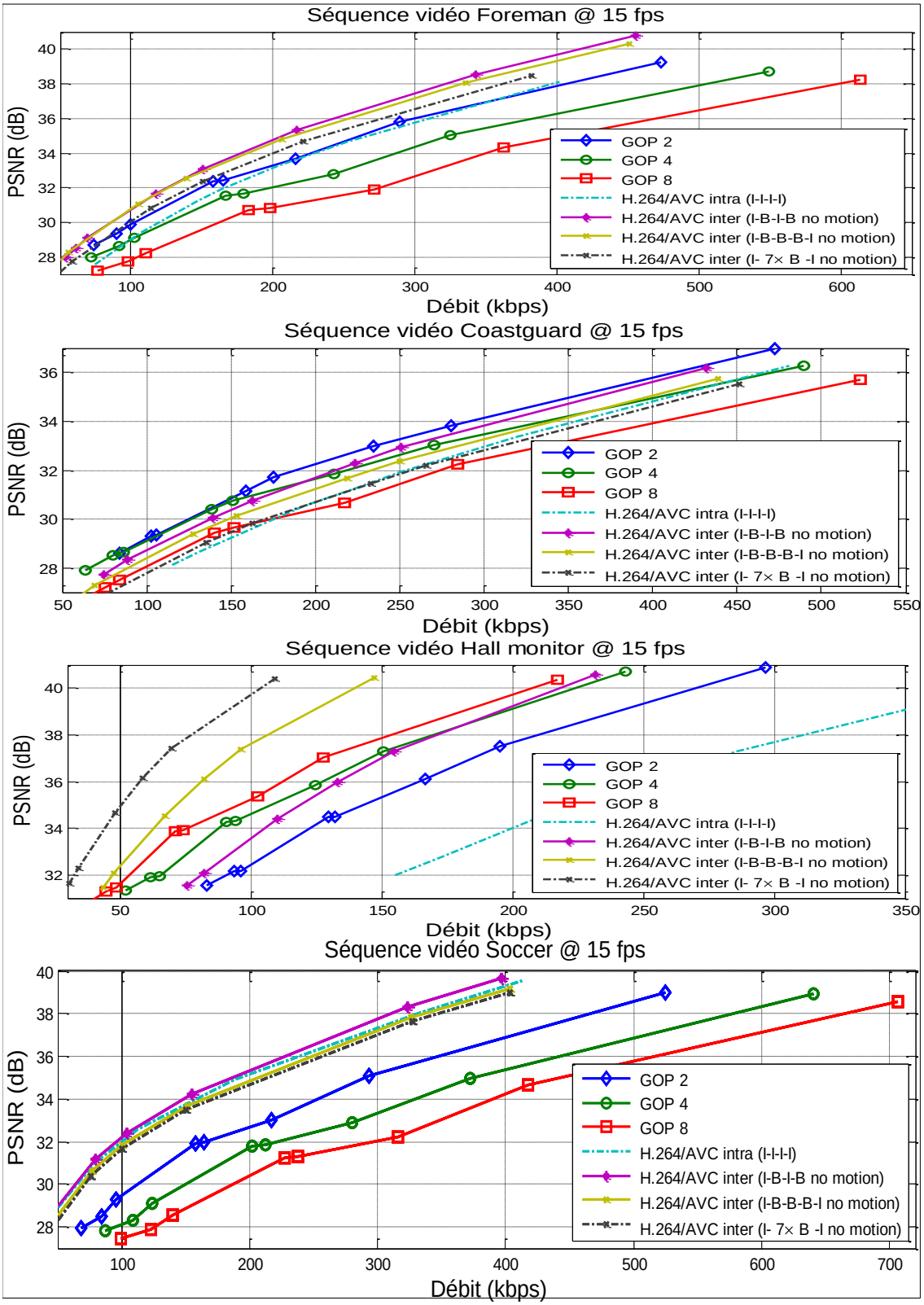

FIGURE 7.21: *Comparaison des performances du codec DVC pour GOP =2, 4 et 8.*

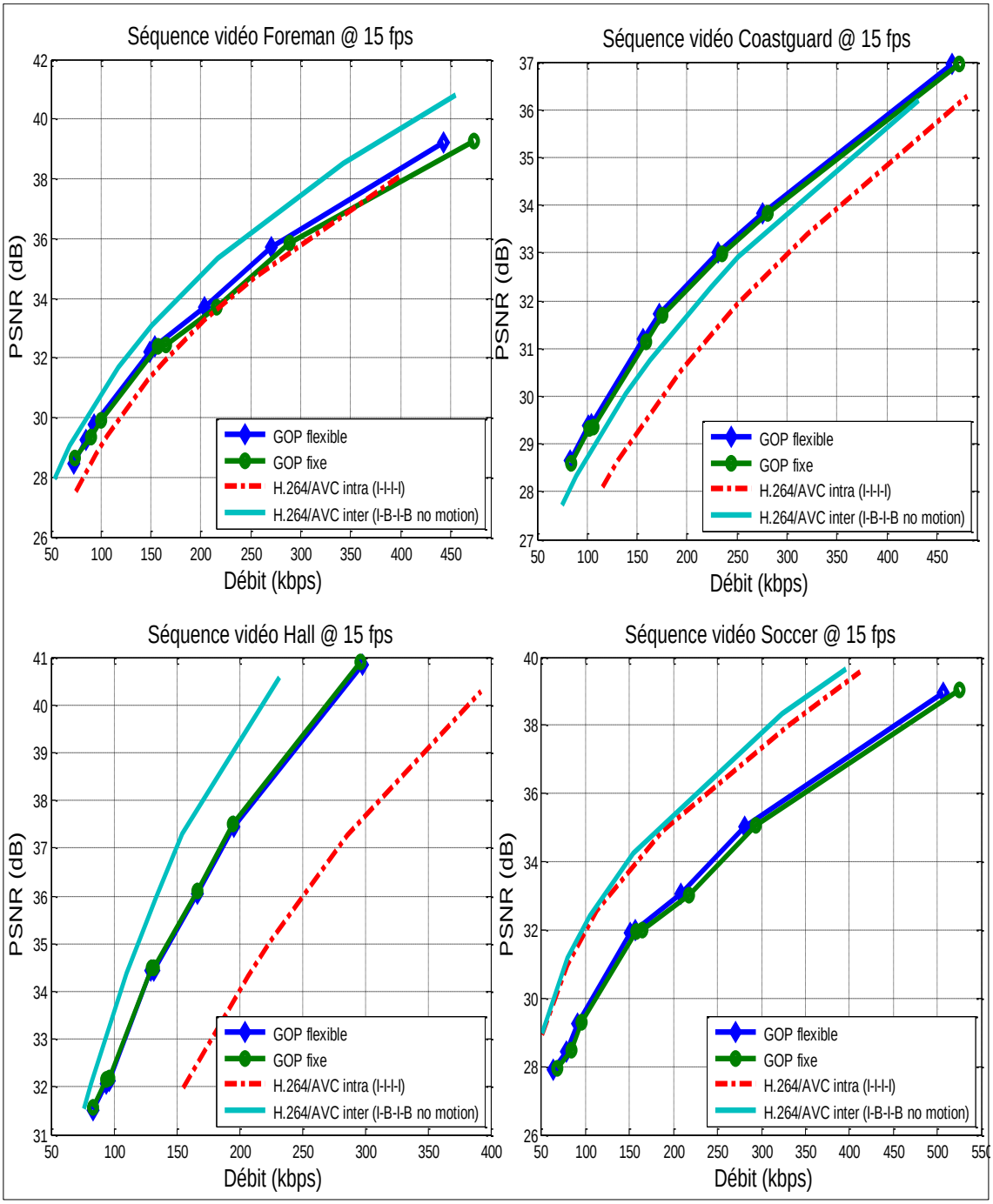

FIGURE 7.22: *Comparaison des performances de l'architecture DVC avec un GOP fixe et un GOP flexible.*

corporer d'autres techniques et d'autres concepts. Même si le système Discover a misé énormément sur les performances, il faut noter qu'il reste encore un écart à rétrécir entre les résultats obtenus et les limites théoriques que peut atteindre le système.

Dans le chapitre suivant on propose d'explorer la corrélation spatiale [2] des trames Wyner-Ziv en transmettant ces dernières progressivement et en raffinant la trame interpolée au fur et à mesure. Cette nouvelle architecture progressive va être incorporée au codec Discover étudié et implémenté précédemment afin d'évaluer son apport de point de vue performance.

2. L'exploration de corrélation temporelle se fait au niveau du bloc d'interpolation générant l'information latérale.

Chapitre 8

Codage vidéo distribué progressif

8.1 Introduction

L'architecture de codage vidéo distribué proposée par Aaron *et al.* en 2002 a été étudiée et améliorée par plusieurs groupes de recherche. On retrouve dans le codec DVC Discover un ensemble de techniques qui ont été mises en place pour améliorer les performances de débit distorsion (R-D). Le codec DVC Discover a été mis à la disposition de la communauté en 2007 [5] comme référence de test et de benchmark. Contrairement au standard H264/AVC (4×4 et 16×16 modes Intra), le codec DVC dans la littérature n'exploite pas de façon efficace la corrélation spatiale. Pour le standard H264/AVC Intra, l'exploitation de la corrélation spatiale entraine une certaine complexité calculatoire au niveau de l'encodeur ce qui n'est pas souhaitable pour l'efficacité d'une architecture DVC avec des encodeurs légers. Dans ce chapitre, on propose une nouvelle architecture DVC de codage et de décodage progressifs qui permet d'exploiter la corrélation spatiale pour le raffinement de l'information latérale (*Side Information Refinement (SIR)*) sans toutefois occasionner une complexité calculatoire supplémentaire au niveau de l'encodeur.

Une trame WZ est subdivisée en blocs qui sont regroupés en un certain nombre de groupes de blocs spatialement corrélés. Le premier groupe de blocs est envoyé au récepteur et décodé en utilisant l'information latérale habituelle. Après le décodage du premier groupe, l'information latérale du second groupe est mise à jour. La subdivision de la trame WZ doit résulter en des groupes présentant une corrélation spatiale. Ainsi, après le décodage du premier groupe, l'information latérale du second groupe peut être raffinée (SIR) entrainant ainsi une réduction du débit. Après avoir décodé le premier et le second groupes de blocs, un raffinement de l'information du troisième groupe est effectuée et ainsi de suite.

Dans ce chapitre on présente les principales techniques de SIR se trouvant dans la littérature. On explique par la suite les motivations du développement de l'architecture progressive et on présente en détails son fonctionnement. Plusieurs dispositions pour la subdivision de la trame WZ sont considérées en 2, 3 et 4 groupes de blocs. On rapporte finalement dans les simulations les améliorations des

performances de débit distorsion (R-D) obtenues par l'architecture DVC progressive en comparaison avec le codec DVC Discover et l'une des plus récentes techniques de raffinement de l'information latérale de la littérature.

8.2 Raffinement de l'information latérale dans la littérature (SIR)

L'architecture DVC n'a pas cessé d'évoluer pour assurer une compétitivité de plus en plus élevée en termes de débit distorsion (R-D). Cependant, un écart de performances significatif persiste entre l'architecture DVC en comparaison avec les standards de codage vidéo prédictifs (i.e. H264/AVC). Ceci est principalement dû à l'exploitation non-idéale des propriétés de la corrélation temporelle lors de la génération de l'information latérale (SI). En effet, l'estimation de mouvement du côté du décodeur fournit uniquement une approximation du mouvement effectif basée sur l'hypothèse de la linéarité de ce dernier. Ainsi la génération de l'information latérale dans le schéma DVC (interpolation ou extrapolation) souffre de l'absence de toute information concernant la trame WZ originale. Plusieurs travaux de recherche ont été menés pour résoudre les problèmes inhérents à l'inefficacité de la génération *aveugle* de la SI, en utilisant certaines données sur la trame courante. Dans ce qui suit on rapporte les principaux travaux de raffinement de l'information latérale (SIR) présentés par la communauté DVC.

8.2.1 Information auxiliaire générée à l'encodeur pour un schéma DVC à faible délai

Pour le codage vidéo WZ à faible délai, l'information latérale est habituellement générée par extrapolation des deux trames précédemment décodées sans l'utilisation d'aucune trame future. Le processus d'estimation de mouvement devient, alors, assez problématique. De plus, l'extrapolation de mouvement est basée sur des trames WZ potentiellement mal reconstruites, ce qui peut conduire à une qualité d'information latérale de plus en plus faible. Pour remédier à cette situation comparable au phénomène de propagation d'erreurs, les auteurs de [67] ont proposé une méthode de génération de l'information latérale basée sur un partitionnement hybride de la trame en 8×8 macroblocs clés et macroblocs WZ (*Key/WZ macroblocks*), selon une structure d'*échiquier*. Les macroblocs clés sont intra-codés et après reconstruction, ils servent au raffinement de l'information latérale pour les macroblocs WZ. En effet, chaque macrobloc WZ est entouré par 4 macroblocs clés qui sont utilisés au cours de l'estimation de mouvement.

Dans [68], une information auxiliaire est générée en considérant la propriété de compactage de la transformée en ondelettes discrète (DWT). Cette information consiste en la sous-bande LL (*low-low*) de la DWT de l'image WZ : la sous-bande LL est un quart de la taille de la trame d'origine. Au niveau du décodeur, l'information auxiliaire est suréchantillonnée par la transformée en ondelettes discrète inverse (IDWT) et utilisée lors de l'estimation de mouvement avec l'image précédemment reconstruite.

Dans [69], l'extrapolation est assistée par l'utilisation des mots de code de hachage robustes consistant en une version sous-échantillonnée et "grossièrement" quantifiée de chaque bloc de la trame WZ. La

distance de *Hamming* entre le mot de code de hachage pour un bloc donné dans la trame WZ et le mot de code de hachage correspondant dans la trame précédente est calculée à l'encodeur. Selon cette distance, l'encodeur décide si le mot de code de hachage doit être envoyé pour aider l'estimation de mouvement au décodeur. Même si le calcul de la distance entre les mots code de hachage enfreint le principe de codage de source distribué, en utilisant des mots de code de faibles tailles, la complexité du codeur n'est pas grandement affectée.

8.2.2 Exploitation de la redondance spatiale au décodeur

Afin de garder l'encodeur DVC aussi léger que possible, les transformations à la source, par exemple la transformée en cosinus discrète (DCT) ou la transformée en ondelettes discrète (DWT), peuvent être évitées. Dans ce cas, la redondance spatiale est exploitée du côté du décodeur tel qu'il a été proposé par [70] et [71]. Dans [70], chaque trame WZ est divisée en deux ensembles selon un motif en damier. L'information latérale du premier sous-ensemble est générée par extrapolation. Le deuxième sous-ensemble a accès à deux composantes d'information latérale (SI) : la première composante SI est générée comme précédemment par *extrapolation temporelle* tandis que la deuxième composante SI est obtenue par *interpolation spatiale* du premier ensemble décodé. En fonction de la différence locale entre la SI extrapolée et le premier ensemble décodé, le décodeur évalue la qualité de la SI temporelle (obtenue par extrapolation temporelle). Si la qualité de la SI temporelle est faible, le décodeur utilise alors la SI spatiale (obtenue par interpolation spatiale). Dans [71], au lieu de sélectionner une seule composante SI, le décodeur utilise plusieurs composantes au cours du processus de décodage turbo. Le calcul des valeurs de vraisemblance turbo du canal virtuel est étendu pour comporter une composante SI temporelle et une composante SI spatiale. Tout comme la corrélation temporelle, la corrélation spatiale entre le pixel courant et ses voisins est également modélisée par une distribution de Laplace lors du calcul des valeurs de vraisemblance canal provenant de la SI spatiale.

8.2.3 Raffinement itératif de l'information latérale dans le domaine des pixels

Les méthodes itératives ont été étudiées pour explorer les propriétés de corrélation spatiale au cours du processus de génération du SI. Dans [72], le raffinement de l'information latérale est basé sur un processus de décodage itératif plan de bits par plan de bits. Le plan de bits décodé apporte une information supplémentaire qui peut être utilisée pour détecter des blocs SI non fiables et ainsi affiner le processus de recherche de mouvement pour ces blocs. Cela conduit à une meilleure SI pour les plans de bits les moins significatifs (*LSB*).

Dans [73], les deux processus d'interpolation temporelle à compensation de mouvement (MCTI) et de décodage turbo sont exécutés de façon itérative. La sortie de la première étape de MCTI est appliquée au décodeur turbo résultant en une trame WZ partiellement décodé (PDWZ) contenant davantage d'informations se rapportant à la trame d'origine. La trame PDWZ peut alors être utilisée pour affiner l'estimation de mouvement afin d'obtenir une information latérale améliorée. Cette opération est appelée *restauration de la compensation de mouvement* (MCR). Le résultat de la MCR est appliqué à

un second lancement du décodage turbo. L'ensemble du processus (MCTI / MCR et décodage turbo) est réitéré un certain nombre de fois. Bien que la majeure partie de l'amélioration de l'information latérale soit obtenue après une seule itération de cet ensemble de processus, la complexité de décodage turbo reste toute de même doublée.

L'approche itérative proposée dans [27] consiste en une technique de raffinement de mouvement visant à corriger les vecteurs suspects et en des outils de correction d'erreurs visant à interpoler les données corrompues à partir des informations décodées avec fiabilité. Le raffinement de mouvement fonctionne comme suit. Une information latérale initiale (ISI) est générée par MCTI et appliquée au décodeur turbo puis au module de reconstruction, résultant en une trame PDWZ. Cette trame PDWZ est envoyée à un module de raffinement pour obtenir une SI finale (FSI) avec une qualité améliorée. Enfin, le module de reconstruction est exécuté une deuxième fois en appliquant à son entrée la FSI à la place de l'ISI. Cela conduit à une augmentation de 1 dB du PSNR des trames reconstruites. L'inconvénient principal de cette méthode est que l'information latérale améliorée est utilisée uniquement pour améliorer la reconstruction et non pas lors du décodage turbo. Ainsi, cette approche ne conduit pas à une réduction du débit binaire global.

8.2.4 Raffinement itératif de l'information latérale dans le domaine des transformées

Dans [74], un mécanisme itératif de raffinement de l'information latérale est intégré dans une architecture DVC dans le domaine de la transformation DCT. La composante DC (fréquence nulle) est décodée en premier à l'aide de l'information latérale initiale générée par interpolation temporelle. Pour les composantes DCT suivantes, l'information latérale est affinée à l'aide de la trame DC (coefficients DC) déjà décodée. La transformation DCT est appliquée aux trames clés voisines pour extraire les trames DC en avant et en arrière. L'estimation de mouvement est effectuée alors entre la trame DC courante et les trames DC voisines : ainsi l'estimation de mouvement est réalisée dans le domaine DC. La compensation de mouvement applique les vecteurs calculés dans le domaine des transformées aux images clés dans le domaine des pixels (c'est-à-dire à la pleine résolution). Après le décodage d'un certain nombre de bandes DCT, une trame WZ partiellement décodée (PDWZ) est obtenue et peut ensuite être utilisée pour affiner les vecteurs de mouvement pour les bandes DCT restantes. La recherche de mouvement est effectuée entre la trame PDWZ et les trames clés voisines.

Dans [75], le raffinement de l'information latérale est également effectué dans le domaine des transformées. Cependant la recherche de mouvement est effectuée dans le domaine des pixels. Plus précisément, après le décodage et la reconstruction de chaque bande DCT, on regagne le domaine des pixels par l'application de l'IDCT sur les bandes décodées à ce point. Chaque bloc dans l'image reconstruite est ensuite comparé au bloc correspondant dans la trame interpolée initiale dont la transformée DCT constitue l'information latérale initiale. Si la somme des erreurs quadratiques dépasse un certain seuil, le bloc est sélectionné pour raffinement. La recherche de mouvement pour ces blocs sélectionnés est effectuée dans la trame interpolée initiale uniquement, sans tenir compte des trames clés réduisant, ainsi, la complexité au décodeur. Les résultats présentés dans [75] seront par la suite considérés à des

fins de comparaison des performances avec l'architecture progressive proposée.

Dans [76] une technique SIR itérative est considérée pour une architecture DVC sans canal de retour. Pour ce schéma avec un contrôle du débit à l'encodeur, le décodeur de canal peut échouer en utilisant le nombre de bits de parité estimés par l'encodeur. Comme le canal de rétroaction est supprimé, le décodeur effectue une nouvelle tentative de décodage avec le même nombre de bits de parité, mais en utilisant une information latérale (SI) amélioré. La SI est améliorée de manière itérative à travers des niveaux de raffinement successifs (RL : *Refinement level*). Pour la première RL (RL_0), les coefficients DC sont reconstruits et utilisés pour affiner le SI . Puis, en RL_1, deux autres bandes DCT sont décodés avec les coefficients DC si le décodage échoue à RL_0. Ainsi, la bande DC de l'information latérale peut être améliorée en utilisant l'information apportée par les bandes suivantes. Ce processus est répété dans le prochain niveau de raffinement RL_2 où une trame WZ entièrement décodée et reconstruite est produite. Si le décodage de certains plans binaires persiste à échouer, des niveaux de raffinement (RL) complémentaire sont considérés. Une technique de SIR similaire avec 4 et 6 RLs est considérée dans [77] pour une architecture avec un canal de retour. Cette dernière technique est également considérée aux fins de comparaison avec la technique progressive proposée en utilisant les métriques Bjøntegaard de variation du débit et du PSNR [6].

8.3 Architecture progressive

8.3.1 Contexte et motivation

Dans l'architecture de codage vidéo distribué conventionnelle, l'estimation des vecteurs de mouvement n'utilise pas la trame WZ courante comme référence, mais utilise plutôt les trames voisines dans une tentative d'estimer la trame courante. L'interpolation, basée sur les blocs dans le contexte Wyner-Ziv, est justifiée par l'hypothèse que chaque bloc est censé avoir un mouvement linéaire allant de la trame précédente à $t-1$ vers la trame suivante à $t+1$ et passant par t. Cette hypothèse tient dans la plupart des cas. Cependant, quand il y a un mouvement rapide, irrégulier ou complexe, l'interpolation, sans aucune connaissance de la trame actuelle peut échouer. Un exemple illustratif de l'échec d'interpolation basée sur les blocs est donné dans la figure 8.1 où le mouvement des objets allant de la trame précédente à $t-1$ vers la trame suivante à $t+1$ n'est pas uniforme. La position de l'objet estimé à l'instant t est généralement à mi-chemin entre les positions de l'objet à $t-1$ et $t+1$. Cependant, cette estimation de la position de l'objet réel à l'instant t peut être erronée, ce qui conduit à des erreurs d'interpolation qui nécessite un nombre important de bits de parité. Pour éviter cela, certains blocs de la trame de référence (courante) doivent être envoyés en premier au décodeur pour ajuster le processus d'interpolation pour les blocs restants de la trame. Ceci constitue le principe fondamental de l'architecture progressive proposée qu'on détaille dans la section suivante.

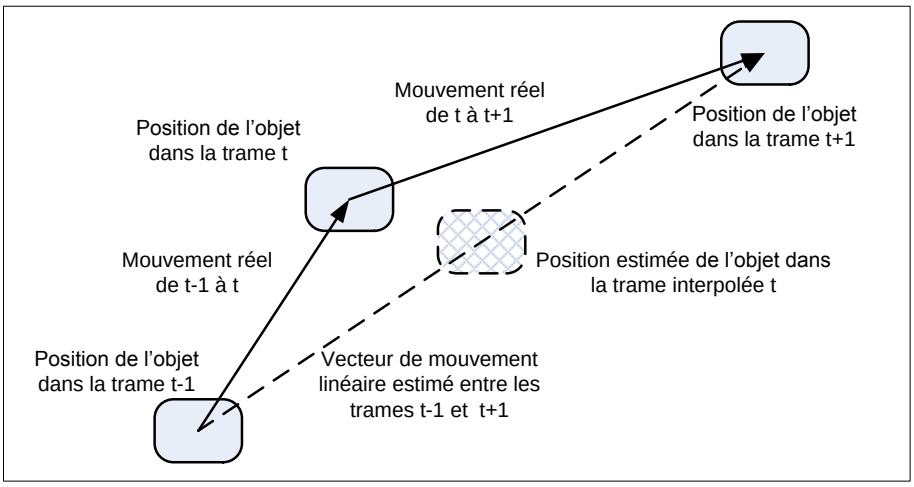

FIGURE 8.1: *Échec d'interpolation pour la détection du mouvement réel dans le cas d'un mouvement non-uniforme.*

8.3.2 Codec progressif avec mise à jour de l'information latérale

L'architecture du codec progressif proposé est représentée à la figure 8.2. Le principe du codage progressif consiste à subdiviser la trame Wyner-Ziv en blocs et ensuite regrouper les blocs en plusieurs ensembles de blocs. Ces ensembles de blocs sont codés progressivement les uns après les autres en utilisant le procédé de codage DVC conventionnel et ils sont transmis les uns après les autres au récepteur. Au niveau du récepteur, ces ensembles de blocs sont progressivement décodés un à la fois. Les ensembles de blocs précédemment décodés sont utilisés pour améliorer la qualité de l'information latérale qui est ensuite utilisée pour le décodage des ensembles de blocs courants et suivants.

Pour assurer l'efficacité de l'amélioration progressive de l'information latérale, les ensembles successifs de blocs doivent être spatialement corrélés. Ainsi, après la reconstruction du premier ensemble, le décodeur peut améliorer la qualité de l'information latérale relative au deuxième ensemble de blocs en détectant et corrigeant l'interpolation des blocs où le mouvement n'est pas uniforme.

Différents schémas de subdivision de la trame WZ peuvent être considérés tant que la corrélation spatiale est maintenue entre les ensembles. Dans ce travail, on étudie en premier lieu le regroupement des blocs de la trame Wyner-Ziv en deux ensembles selon une *structure d'échiquier*, démontrant une corrélation verticale et horizontale entre les deux ensembles.

8.3.3 Architecture progressive avec 2 groupes de blocs selon la structure d'échiquier

Chaque trame Wyner-Ziv est divisée en deux ensembles comme le montre la figure 8.3. Le premier ensemble comporte les " blocs" noirs et le second comporte les blocs " blancs". Le processus d'encodage et de décodage est décrit comme suit. L'ensemble des blocs noirs est d'abord codé et transmis en utilisant l'architecture DVC conventionnelle. Au récepteur, le décodeur génère l'information laté-

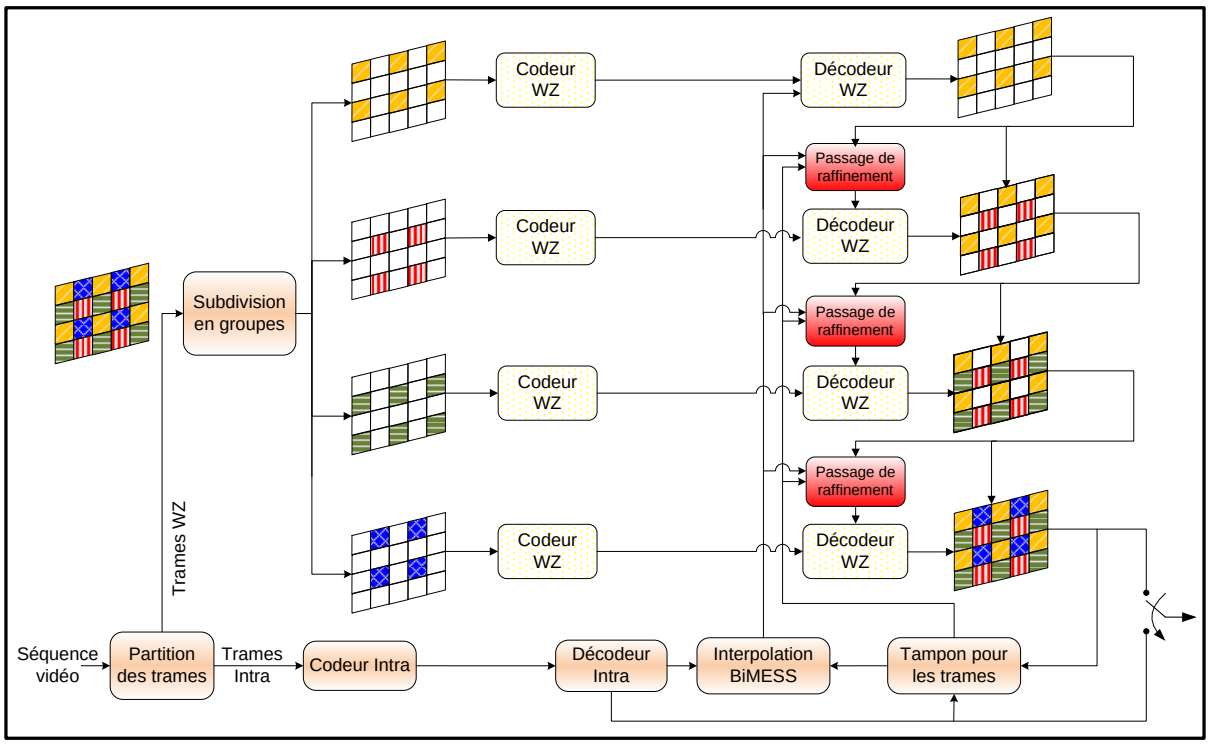

FIGURE 8.2: *Codage et décodage progressifs de la trame WZ groupe par groupe.*

rale pour les trames Wyner-Ziv utilisant la méthode d'interpolation BiMESS. L'ensemble des blocs noirs est décodé en utilisant cette trame interpolée comme information latérale. Ce premier ensemble de blocs décodés (les blocs noirs) est utilisé pour améliorer la qualité de l'information latérale pour le second ensemble formé par les blocs blancs. L'encodeur transmet l'ensemble des blocs blancs en utilisant la même méthode de codage et le récepteur décode les blocs blancs en utilisant l'information latérale améliorée. Comme l'information latérale pour les blocs blancs a été améliorée, ces blocs devraient avoir besoin de moins de bits parité pour la convergence du décodeur turbo : il en résulte alors une réduction du débit total.

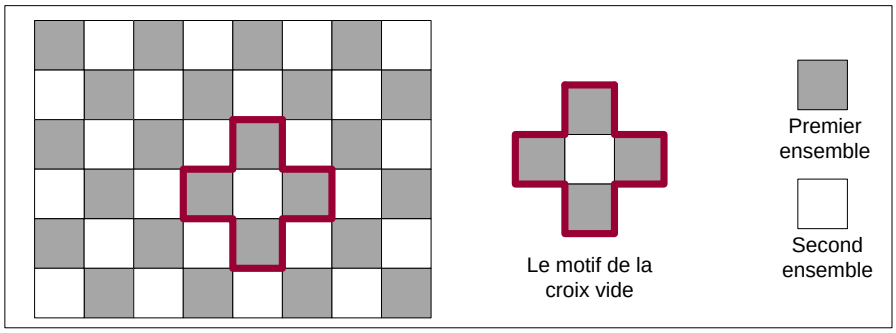

FIGURE 8.3: *Division des blocs en deux ensembles résultant en un motif de croix vide.*

Mise à jour de la trame interpolée en utilisant les blocs décodés

Il y a un certain nombre d'approches possibles pour améliorer la qualité de l'information latérale du deuxième groupe formé par les blocs noirs en utilisant les blocs blancs déjà décodés du premier groupe. On constate à partir de la décomposition en *structure d'échiquier* que chaque bloc blanc (non encore reçu) est entouré par quatre blocs noirs (déjà décodés). Ces quatre blocs noirs forment une *croix vide*, comme indiqué à la figure 8.3. Nous prenons d'abord la trame clé précédemment décodée comme trame de référence et on cherche la meilleure correspondance de chaque motif de croix vide dans cette trame de référence balayant une zone de recherche de 44×44 pixels. Le critère de correspondance considéré est la moyenne des différences absolues (*Mean of Absolute Differences* (MAD)) : ce critère est calculé à partir des quatre blocs entourant le bloc central. Une fois la meilleure correspondance de la croix vide localisée dans l'image précédente, le bloc central est considéré comme une première estimation du bloc blanc. La même approche est ensuite appliquée en faisant la recherche de la meilleure correspondance de la croix vide dans la trame clé suivante. En ce qui concerne la trame interpolée, on prend directement le bloc central du motif de la croix vide sans effectuer aucune estimation de mouvement. En effet, la génération de la trame interpolée met en œuvre une procédure sophistiquée d'estimation de mouvement (BiMESS). Si le résultat de cette interpolation temporelle est de bonne qualité il pourra contribuer lors de la compensation de mouvement dans l'architecture progressive moyennant un facteur de pondération (voir l'équation (8.1)).

La procédure de raffinement du mouvement est illustrée davantage à la figure 8.4. À la fin de cette procédure, il y a trois candidats pour le bloc blanc se trouvant dans les trois différentes trames de référence (trame clé précédente, trame clé suivante et trame interpolée). En se basant sur ces trois blocs candidats, la compensation de mouvement pour le bloc blanc est effectuée. On présente deux méthodes de compensation de mouvement progressive dans la section suivante.

Compensation de mouvement lors de la mise à jour de l'information latérale

Pour la mise à jour de l'information latérale, la compensation de mouvement, consiste à remplir le bloc vide de la croix (bloc blanc) selon les trois meilleures correspondances trouvées dans l'étape précédente. Ici, deux méthodes de compensation de mouvement sont considérées pour le raffinement de l'information latérale de l'architecture progressive.

1. Première méthode : en premier lieu, les deux meilleures correspondances de la croix vide dans les trames précédente et suivante sont trouvées. La moyenne des différences absolues entre chacunes des meilleures correspondance et la croix vide du premier ensemble décodé est calculée. Les valeurs obtenues sont dénotées MAD_{prec} et MAD_{suiv}. En second lieu la moyenne des différences absolues (MAD) entre la croix vide du premier ensemble décodé et son homologue co-localisé dans la trame interpolée est calculée. La valeur obtenue est dénotée MAD_{inter}. Ensuite, le bloc central de la croix avec la valeur MAD minimale est sélectionné. Ce bloc central va servir d'information latérale pour le décodage du bloc dans le second ensemble. Toutefois, cette méthode n'est pas statistiquement optimale et ses performances sont affectées par les er-

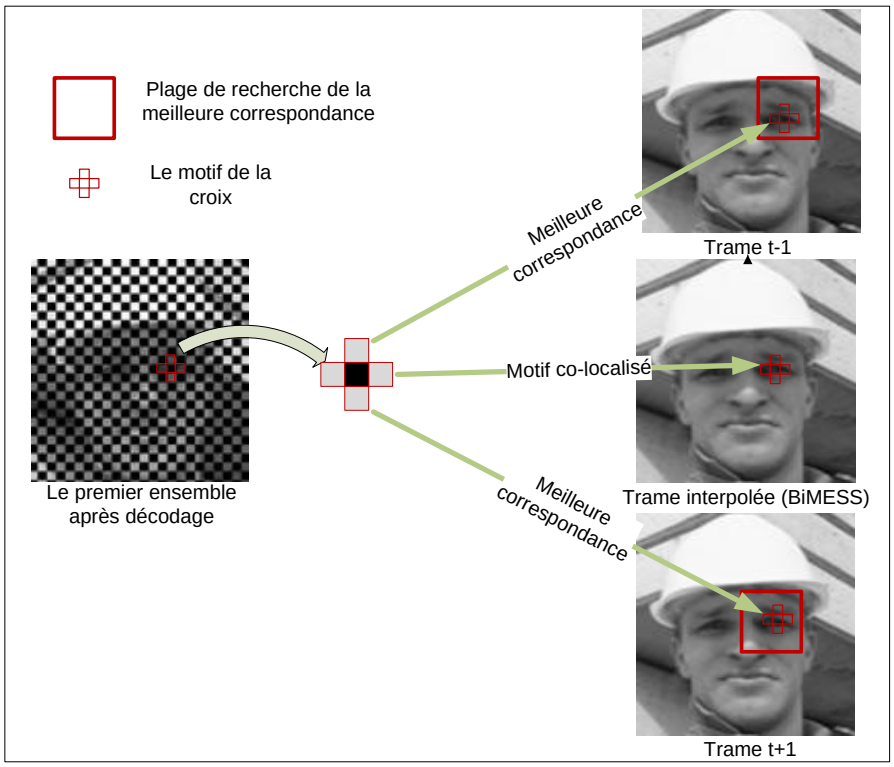

FIGURE 8.4: *Algorithme de mise à jour de l'information latérale.*

reurs qui peuvent survenir à l'étape précédente de recherche de la meilleure correspondance. En effet, le décodage du premier ensemble n'est pas sans erreur et la plage de recherche affiche une grande redondance spatiale. La recherche du bloc correspondant le mieux à la croix vide peut ainsi induire en erreur. Une deuxième méthode utilisant les trois différents candidats pour la compensation du bloc central de la croix vide peut être considérée.

2. Deuxième méthode : elle consiste en l'addition des trois blocs estimés (obtenus de la trame clé précédente, de la trame clé suivante et de la trame interpolée) à l'étape précédente pondérés par des coefficients calculés à partir du critère de correspondance. Bien que ce ne soit pas la meilleure correspondance qui est sélectionnée, d'un point de vue statistique, cette méthode de compensation de mouvement est plus performante que la première méthode de compensation de mouvement. Le bloc compensé $Bloc_{comp}$, est donné par :

$$Bloc_{comp} = \frac{\frac{1}{MAD_{prec}} Bloc_{prec} + \frac{1}{MAD_{suiv}} Bloc_{suiv} + \frac{1}{MAD_{inter}} Bloc_{inter}}{\frac{1}{MAD_{prec}} + \frac{1}{MAD_{suiv}} + \frac{1}{MAD_{inter}}} \tag{8.1}$$

Ici, MAD_X désigne la moyenne des différences absolues entre le motif de la croix vide obtenu du premier ensemble de blocs décodés et la meilleure correspondance dans les trames précédente (*prec*), suivante (*suiv*) ou interpolée (*inter*). $Bloc_X$ est ainsi le bloc à l'intérieur de la croix vide. Les facteurs $\frac{1}{MAD_X}$ ($X = prec$, $suiv$ ou $inter$) dans l'équation (8.1) indique le niveau de

correspondance de chaque bloc central de la croix vide dans la trame précédente, suivante ou interpolée. Une valeur élevée MAD_{prev} signifie que le bloc central de la croix vide obtenu à partir de la trame précédente contribuera à peine dans le bloc compensé.

8.3.4 Architecture progressive avec 3 groupes de blocs

Dans la méthode progressive présentée précédemment, uniquement le deuxième groupe de blocs bénéficie de l'amélioration de l'information latérale. Ainsi, le premier groupe de blocs, équivalent à la moitié de la trame, est décodé en utilisant la trame interpolée par l'algorithme BiMESS. L'aspect progressif est alors restreint à seulement la moitié de la trame. À la lumière de cela, on peut penser à diviser la trame WZ en plusieurs groupes (3 ou 4 groupes). Néanmoins, l'efficacité de la méthode progressive est étroitement liée à la corrélation entre les groupes de blocs déjà décodés et le groupe qui s'en suit. D'où l'existence d'un compromis quant au choix du nombre de groupes. D'une part, il est préférable de choisir un grand nombre de groupes pour faire bénéficier une grande partie de la trame WZ de l'architecture progressive (et non pas uniquement la moitié de la trame WZ). D'autre part, l'augmentation du nombre de groupes peut affecter la corrélation entre ces derniers, et par conséquent, affecter l'efficacité du raffinement progressif de l'information latérale.

Dans cette section, on considère une architecture progressive avec la subdivision de la trame Wyner-Ziv en trois groupes de blocs tel que le montre la figure 8.5. Ainsi la mise à jour de l'information latérale est effectuée en deux temps. En un premier temps, le premier groupe de blocs décodé servira à l'amélioration de l'information latérale relative au deuxième groupe de blocs. Ensuite, les premier et deuxième groupes de blocs décodés serviront à l'amélioration de l'information latérale relative au troisième groupe de blocs.

Tout d'abord le premier ensemble de blocs est encodé et décodé selon un schéma DVC conventionnel c'est-à-dire avec une information latérale générée par l'algorithme BiMESS. Par la suite, en utilisant ces blocs décodés du premier ensemble, un premier passage de raffinement de l'information latérale relative au second ensemble de blocs est effectué. Ce passage de raffinement considère un motif composé des quatre blocs voisins diagonaux. Finalement, le résultat de décodage des deux premiers ensembles blocs est utilisé pour raffiner la qualité de l'information latérale relative au troisième ensemble de blocs. Dans ce deuxième passage de raffinement, on retrouve le motif de la croix vide considéré dans l'architecture progressive avec 2 groupes de blocs étudiée précédemment. Les détails sur les deux passages de raffinement de l'information latérale sont illustrés à la figure 8.5. Les techniques d'estimation et de compensation de mouvement sont identiques à celles décrites pour le schéma progressif avec deux ensembles de blocs.

8.3.5 Architecture progressive avec 4 groupes de blocs

Pour exploiter davantage l'architecture progressive, on considère une subdivision de la trame Wyner-Ziv en quatre ensembles de blocs résultant en trois passages de raffinement de l'information latérale comme le montre la figure 8.6.

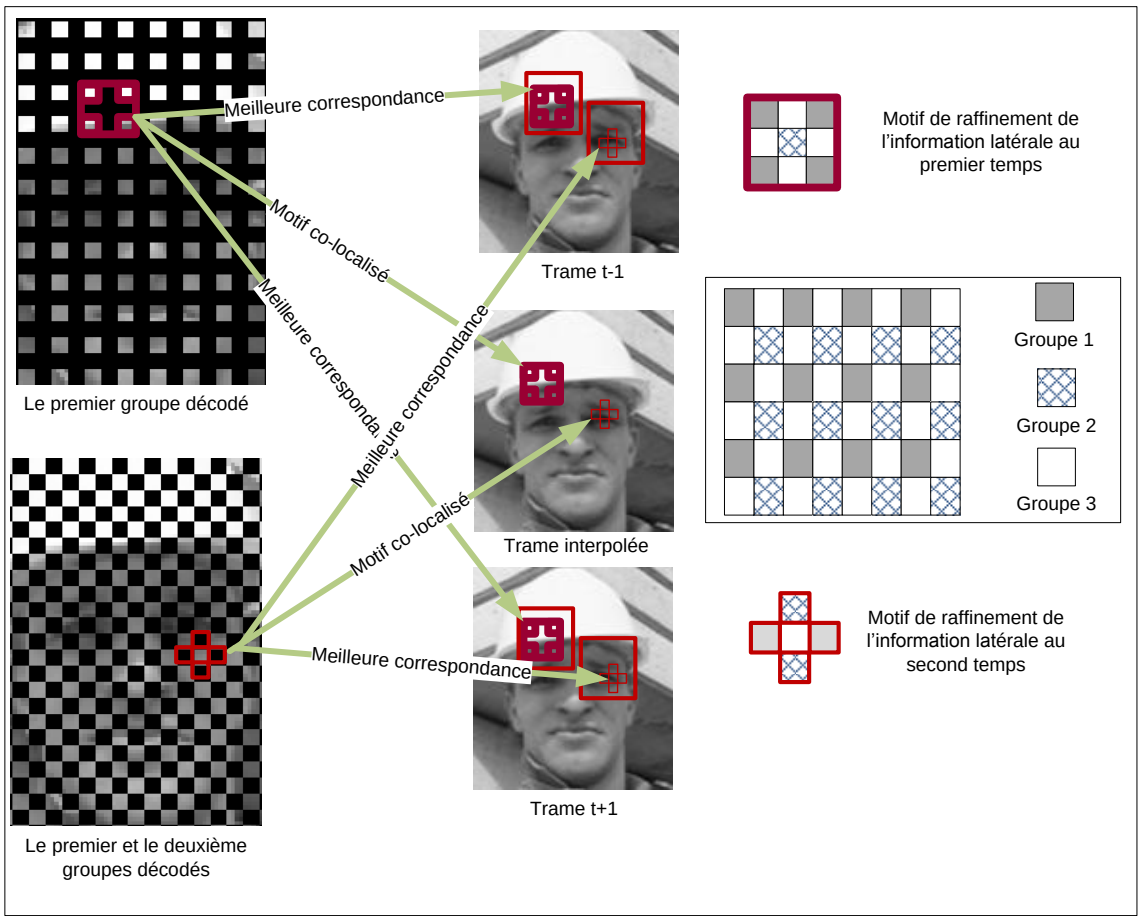

FIGURE 8.5: *Schéma du codage progressif avec une subdivision de la trame WZ en trois groupes de blocs et deux passages de raffinement l'information latérale.*

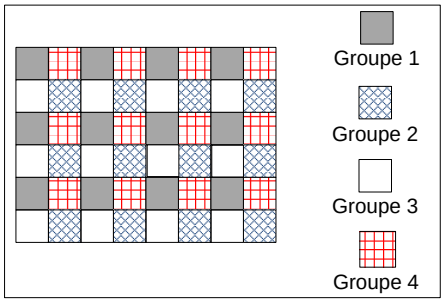

FIGURE 8.6: *Subdivision de la trame WZ en quatre groupes de bloc.*

Les deux premiers passages de raffinement de l'information latérale sont identiques aux deux passages de l'architecture progressive avec trois groupes de blocs. Pour le troisième passage, on considère un motif de raffinement comprenant tous les huit blocs voisins du bloc en question. Ces trois passages de raffinement et les motifs correspondants sont illustrés à la figure 8.7.

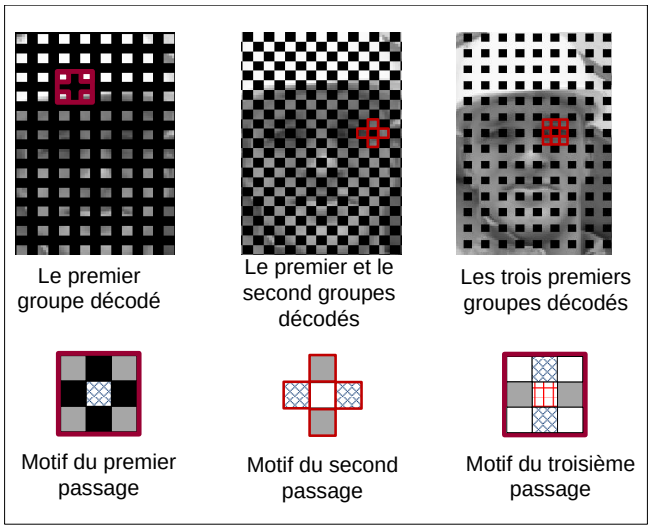

FIGURE 8.7: *Schéma progressif avec une subdivision de la trame WZ en quatre groupes de blocs et trois passage de raffinement l'information latérale.*

8.3.6 Facteur de corrélation spatio-temporelle des différents motifs de raffinement

Les motifs utilisés le long des différents passages de raffinement sont dénommés comme suit :

Motif de la croix vide / *Empty cross template (EC)* : Ce motif contient 4 blocs de 16 pixels $= 64$ pixels. Il est utilisé dans l'unique passage de raffinement du schéma progressif avec 2 groupes, dans le second passage de raffinement du schéma progressif avec 3 groupes et dans le troisième passage de raffinement du schéma progressif avec 4 groupes.

Motif des 4 blocs diagonals / *Four diagonal blocks template (4D)* : Ce motif contient 4 blocs de 16 pixels $= 64$ pixels. Il est utilisé dans le premier passage de raffinement du schéma progressif avec 3 groupes et dans le premier passage de raffinement du schéma progressif avec 4 groupes.

Motif avec tous les blocs voisins / *All neighboring blocks template (AN)* : Ce modèle contient 8 blocs de 16 pixels $= 128$ pixels. Il est utilisé dans le troisième passage de raffinement du schéma progressif avec 4 groupes.

La pertinence du choix des motifs de raffinement (modèles) mentionnés ci-dessus est déterminée par le calcul d'un facteur de corrélation spatio-temporel. Ce facteur vise à évaluer dans quelle proportion les modèles proposés sont en mesure d'affiner le mouvement en utilisant les trames voisines. Le calcul de ce facteur est basé sur des mesures hors ligne, envisageant les 5 séquences vidéo : `Foreman`, `Coastgaurd`, `Hall monitor`, `Soccer` et `Carphone`. Chaque trame est divisée en blocs de 4×4

pixels. En considérant, ensuite, un modèle de raffinement autour de chaque bloc, B, la meilleure correspondance est recherchée dans les trames voisines précédente et suivante en balayant une zone de recherche de 44 $\times$ 44 pixels. Les blocs centraux dans la trame précédente, B_P, et dans la trame suivante, B_N, sont utilisés pour calculer le facteur de corrélation comme suit :

$$\rho = \frac{\frac{1}{MAD_P}\frac{cov(B,B_P)}{\sigma_B \sigma_{B_P}} + \frac{1}{MAD_N}\frac{cov(B,B_N)}{\sigma_B \sigma_{B_N}}}{\frac{1}{MAD_P} + \frac{1}{MAD_N}} \tag{8.2}$$

où $cov(X,Y) = E[XY] - E[X]E[Y]$ désigne la covariance de X et Y, et $\sigma_X = \sqrt{E[X^2] - E[X]^2}$ désigne l'écart type de X. $E[X]$ dénote la valeur moyenne de X.

Le tableau 8.1 montre les facteurs de corrélation calculés à partir des simulations considérant toutes les trames des 4 séquences vidéo. Tel qu'attendu, les modèles *EC* et *AN* présentent plus de corrélation que le modèle 4*D*. Toutefois, le modèle *EC* est légèrement meilleur que le modèle *AN*, même si ce dernier contient plus de blocs. En fait, le motif *EC* considérant uniquement les blocs horizontaux et verticaux est perd de la corrélation en ajoutant les blocs diagonaux. Notez que cette étude est menée pour évaluer l'efficacité de chaque modèle et qu'elle est basée sur les trames originales sans distorsion (sans quantification). Le codage DVC progressif, toutefois, applique le raffinement de mouvement sur des images reconstruites avec une certaine quantité de distorsion. En outre, il considère également la trame interpolée lors de la compensation du mouvement et non seulement les trames voisines.

	4D	*EC*	*AN*
`Foreman`	0.7775	0.8100	0.8052
`Coastguard`	0.8404	0.8582	0.8553
`Hall monitor`	0.8891	0.9035	0.9022
`Soccer`	0.6344	0.7134	0.6913
`Carphone`	0.9659	0.9716	0.9705
Average	0.8215	0.8513	0.8449

Tableau 8.1: *Facteur de corrélation spatio-temporel ρ des motifs de raffinement.*

8.4 Résultat de simulation

Dans cette section on s'intéresse à l'implémentation des différentes architectures progressives présentées dans les sections précédentes. Pour évaluer l'apport de la technique progressive on l'incorpore dans deux architectures DVC conventionnelles :

1. Architecture de *Brites et al.* [11] : cette architecture considère que les trame clés sont parfaitement reconstruites au niveau du décodeur et elle s'intéresse uniquement aux performances R-D relatives aux trames WZ.
2. Architecture de Discover [5] : cette architecture considère que les trames clés envoyées par le standard H264/AVC constituant ainsi un scénario plus réaliste. Les trames clés reconstruites

au décodeur présentent alors une certaine distorsion qui peut influer lors du raffinement de l'information latérale. En effet, la recherche des motifs des passages de raffinement est effectuée dans ces trames clés.

8.4.1 Incorporation de la technique progressive sur l'architecture de *Brites et al.* [11]

Dans cette section, on implémente les trois techniques proposées de codage vidéo distribué progressif afin de présenter les améliorations obtenues en comparaison avec l'un des meilleurs résultats trouvés dans la littérature publié par *Brites et al.* [11]. Nous avons d'abord reproduit les résultats présentés par *Brites et al.* qui supposent que les images clés sont parfaitement connues au décodeur. Ensuite, seule la fonction débit-distorsion des trames WZ est reportée. Les résultats des simulations considèrent quatre séquences vidéo QCIF @ 30 trames par seconde : *Foreman*, *Mother and Daugther*, *Salesman* et *Carphone*. Ces séquences sont téléchargées à partir du site Web du centre de l'Université de Stanford pour l'ingénierie des systèmes et des images [53]. Pareillement à [11], seules les 101 premières images sont considérées et la génération de l'information latérale est basée sur la technique d'estimation de mouvement bidirectionnelle avec lissage spatial (BiMESS).

Qualité de l'information latérale pour les différentes architectures progressives

Avant de rapporter les performances débit-distorsion de l'architecture progressive, on s'intéresse à la qualité de l'information latérale lors des passages de raffinement. On rapporte dans les tableaux 8.2, 8.3 et 8.4, le PSNR des trames interpolées pour les différents groupes de blocs des 3 architectures progressives proposées. Ces trois tableaux montrent l'amélioration de la qualité de l'information latérale de chaque groupe de blocs en effectuant la moyenne sur les différentes trames WZ des différentes séquences.

	Groupe 1	Groupe 2
`Foreman`	36.7823	38.4369
`Carphone`	35.0049	37.9426
`Salesmman`	44.2555	44.8114
`Mother and Daughter`	42.3797	43.7461

Tableau 8.2: *PSNR (en dB) des 2 groupes de blocs de la trame interpolée de l'architecture progressive avec un seul passage de raffinement.*

	Groupe 1	Groupe 2	Groupe 3
`Foreman`	36.8150	38.0156	38.7330
`Carphone`	34.9413	37.0005	38.2244
`Salesmman`	44.1643	44.7052	44.8943
`Mother and Daughter`	42.4762	43.1938	43.8350

Tableau 8.3: *PSNR (en dB) des 3 groupes de blocs de la trame interpolée de l'architecture progressive avec 2 passages de raffinement.*

	Groupe 1	Groupe 2	Groupe 3	Groupe 4
Foreman	36.8150	38.0156	38.5075	38.9672
Carphone	34.9413	37.0005	38.1301	38.4996
Salesmman	44.1643	44.7052	45.0079	44.9028
Mother and Daughter	42.4762	43.1938	44.0708	43.6568

Tableau 8.4: *PSNR (en dB) des 4 groupes de blocs de la trame interpolée de l'architecture progressive avec 3 passages de raffinement.*

Le tableau 8.5 présente la qualité moyenne de l'information latérale générée par la technique d'interpolation BiMESS utilisé par *Brites et al.* et les différentes architectures progressives. On remarque que la qualité de l'information latérale est supérieure lorsque le nombre des passages de raffinement augmente.

	BiMESS	Progressive avec 1 passage	Progressive avec 2 passages	Progressive avec 3 passages
Foreman	36.73	37.6096	37.8545	38.0763
Carphone	35.07	36.4737	36.7221	37.1429
Salesmman	43.95	44.5335	44.5879	44.6950
Mother and Daughter	42.03	43.0629	43.1683	43.3494
Moyenne	39.4450	40.4199	40.5832	40.8159

Tableau 8.5: *Récapitulatif de la qualité de l'information latérale des diffèrentes architectures DVC.*

Comparaison entre les performances des différentes architectures progressives

On présente à la figure 8.8 les performances de l'architecture progressive en comparaison avec l'architecture DVC conventionnelle de *Brites*. Le niveau de distorsion est calculé en utilisant deux mesures de distorsion : le PSNR qui est une mesure couramment utilisée pour l'évaluation objective en traitement d'images et le SSIM qui est un indice de similarité structurelle prenant en compte les cohérences spatiales dans l'évaluation de distorsion.

La performance de la technique progressive avec deux passages de raffinement dépasse la performance de celle obtenue avec un seul passage par jusqu'à 0,4 dB. Les performances de la technique progressive avec trois passages sont légèrement plus élevées que celles résultant de deux passages. Cette augmentation très faible, en dépit de l'amélioration de la qualité de la trame interpolée (voir le tableau 8.5), s'explique d'un point de vue de codage de canal. En effet, lorsque la trame est divisée en plusieurs ensembles, la longueur du code turbo et de son entrelaceur est plus faible. Ainsi, sa capacité de correction d'erreurs diminue et, par conséquent, un nombre supérieur de bits de parité est nécessaire pour la convergence du processus de décodage turbo. En définitive, l'architecture progressive avec trois passages de raffinement résulte en une amélioration allant jusqu'à 2 dB par rapport à l'architecture conventionnelle comme on peut le voir dans le cas de la séquence vidéo *Carphone*.

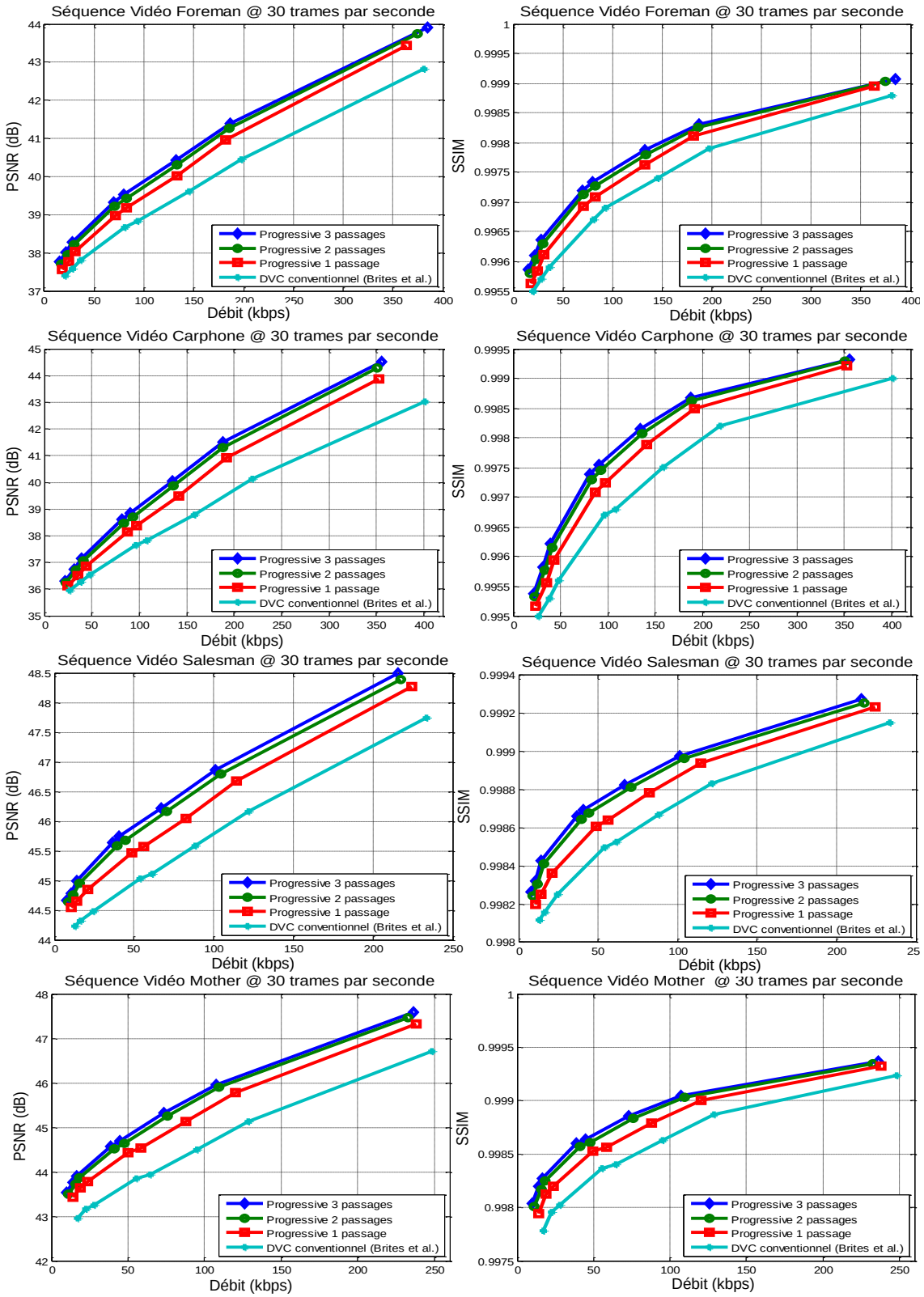

FIGURE 8.8: *Comparaison entre les architectures progressives et conventionnelle.*

8.4.2 Incorporation de la technique progressive dans l'architecture de Discover pour différentes valeurs du GOP

Dans la sous-section précédente, on a montré l'apport de la technique progressive pour une architecture qui considère les trames clés parfaitement reconstruites au décodeur et pour un groupe d'images $GOP = 2$. Dans cette sous-section, on étend les simulations relatives à la technique progressive en considérant l'architecture de Discover comme architecture de base. Ainsi on peut étudier le comportement et l'efficacité du schéma progressif lorsque les trames clés présentent une certaine distorsion. Par ailleurs, on peut étudier la contribution du schéma progressif pour différentes valeurs du groupe d'images $GOP = 2$, 4 et 8. Lorsque le GOP augmente la corrélation temporelle diminue et il est intéressant de compenser cette diminution par l'exploration de la corrélation spatiale grâce à la technique progressive.

Conditions des tests de simulation

Outre la comparaison du schéma DVC progressif avec la référence DVC conventionnelle (benchmark Discover), les performances de la technique SIR (*Side information refinement*) proposée basée sur la corrélation spatiale est comparée avec la technique SIR présentée dans [75]. Pour une comparaison appropriée, les mêmes conditions de simulations sont considérées. Ces conditions sont d'ailleurs prises du site web d'évaluation du codec Discover [5] (voir le chapitre 7). En ce qui concerne les paramètres du schéma progressif proposé, on énumère les points suivants. La décomposition des trames WZ résulte en des blocs de 4×4 pixels. La zone de recherche de la meilleure correspondance du motif s'étend sur 28×28 pixels. Comme chacun des trois motifs (motif diagonal, motif de la croix vide et motif de tous les blocs voisins) couvre une superficie de 12×12 pixels, une zone de recherche de 28×28 pixels permet un déplacement de 8 pixels dans chaque direction (vers la droite, la gauche, le haut et le bas). La zone de recherche doit être suffisamment large pour capturer les mouvements rapides en particulier pour des valeurs élevées du GOP. Toutefois, pour éviter de capturer le bruit au lieu de vrai mouvement, la taille de la zone de recherche ne peut pas être indéfiniment augmentée.

Avant d'analyser les performances globales de débit distorsion de l'architecture progressive proposée, la génération de l'information latérale est examinée à part. En effet, la qualité de la trame interpolée a un impact direct sur la réduction du débit réalisable et sur la qualité de reconstruction de la séquence vidéo. L'efficacité du schéma progressif, interprétée par l'amélioration de la qualité de l'information latérale, est rapportée pour trois tailles du GOP impliquant différentes conditions de corrélation temporelle. On commence alors par évaluer la qualité de la SI obtenue par la technique proposée, en termes de PSNR, au point de quantification à haut débit $Q_i = 8$ (voir la figure 5.6).

Les tableaux 8.6, 8.7 et 8.8 fournissent les valeurs du PSNR de l'information latérale relatives à chacun des ensembles de blocs (ou groupes de blocs) des architectures progressives avec un, deux et trois passages de raffinement. Le PSNR de chaque groupe donné dans ces tableaux est la moyenne du PSNR de ce groupe sur toutes les trames WZ dans la séquence vidéo. La comparaison de chaque

groupe avant et après le raffinement aurait également pu être réalisée. Toutefois, c'est pratiquement la même que la comparaison de la qualité de la SI entre chaque groupe. En fait, les différents groupes de blocs représentent pratiquement le même contenu vu qu'ils sont uniformément répartis sur la trame WZ. La moyenne est calculée sur l'ensemble des PSNRs de toutes les trames WZ des quatre séquences vidéo afin de permettre une comparaison statistique globale de la qualité de l'information latérale pour les séquences vidéo agrégées. L'amélioration obtenue par le raffinement de la SI est évaluée par la différence de PSNR, $\Delta PSNR_{1g}^{motif}$, entre le premier groupe (sans raffinement) et le deuxième, troisième ou quatrième groupe ($g = 2$, 3 ou 4) après raffinement en utilisant l'un des motifs susmentionnés (motif : *EC*, *4D* ou *AN*).

Qualité de l'information latérale de l'architecture progressive avec un seul passage de raffinement

Pour déterminer l'amélioration de la qualité de l'information latérale au cours des différents passages de raffinement, nous rapportons, dans le tableau 8.6, le PSNR des trames interpolées pour les deux groupes de blocs avec un $GOP = 2$, 4 et 8. Ce tableau montre l'amélioration de la qualité de l'information latérale du deuxième groupe (après raffinement utilisant le modèle *EC*) par rapport au premier groupe (sans raffinement) en considérant la moyenne des trames entières des différentes séquences. Les valeurs de PSNR démontrent que le motif de raffinement de la croix vide, *EC*, est efficace dans l'amélioration de l'estimation du mouvement. Ceci revient à la corrélation spatiale entre la croix vide et le bloc à l'intérieur.

En outre, ce tableau montre que l'apport de l'architecture progressive est plus important lorsque la corrélation temporelle diminue, ou lorsque la taille du *GOP* augmente. L'amélioration obtenue par le raffinement de mouvement en exploitant la corrélation spatiale, est plus prononcée lorsque la corrélation temporelle est faible. En fait, lorsque le *GOP* augmente, l'interpolation temporelle (MCTI) estime le mouvement entre deux images clés temporellement éloignées. Le véritable mouvement n'est pas facile à détecter et le schéma progressif s'avère particulièrement utile pour de grandes valeurs du *GOP*.

Qualité de l'information latérale de l'architecture progressive avec deux passages de raffinement

Les performances d'interpolation du schéma progressif avec deux passages de raffinement sont rapportés dans le tableau 8.7 pour les 3 valeurs différentes du GOP. Comme mentionné précédemment, le premier passage de raffinement est basé sur le motif *4D* formé des 4 blocs diagonaux. Ce motif présente une corrélation spatiale avec le bloc central plus faible que la corrélation du motif *EC* de la croix vide utilisé dans le second passage de raffinement. Par conséquent, le deuxième passage de raffinement résulte en un PSNR (troisième groupe) plus élevé, en moyenne, que celui obtenu avec le premier passage (deuxième groupe). Pour les séquences à faible mouvement, cependant, le motif de raffinement *4D* du premier passage n'est pas assez précis pour augmenter la précision de l'esti-

GOP = 2			
	Groupe 1	Groupe 2	$\Delta PSNR_{12}^{EC}$
Foreman	29.558	33.282	**3.724**
Coastguard	31.761	32.858	**1.097**
Hall monitor	36.084	36.794	**0.710**
Soccer	21.678	27.51	**5.832**
Carphone	29.243	32.560	**3.3170**
Moyenne	29.664	32.601	**2.9370**
GOP = 4			
	Groupe 1	Groupe 2	$\Delta PSNR_{12}^{EC}$
Foreman	27.78	32.399	**4.619**
Coastguard	30.144	31.714	**1.570**
Hall monitor	34.902	35.935	**1.033**
Soccer	20.509	26.312	**5.803**
Carphone	28.703	32.198	**3.495**
Moyenne	28.407	31.711	**3.304**
GOP = 8			
	Groupe 1	Groupe 2	$\Delta PSNR_{12}^{EC}$
Foreman	26.407	31.332	**4.925**
Coastguard	29.082	30.93	**1.848**
Hall monitor	34.053	35.411	**1.358**
Soccer	19.908	26.285	**6.377**
Carphone	28.061	31.841	**3.780**
Moyenne	27.502	31.160	**3.658**

Tableau 8.6: *PSNR (en dB) des 2 groupes de blocs de la trame interpolée de l'architecture progressive avec un seul passage de raffinement.* $\Delta PSNR_{12}^{EC}$ *: différence du PSNR entre le groupe 2 après raffinement utilisant le motif* EC *et le groupe 1 sans raffinement.*

mation de mouvement. En d'autres termes, l'estimation de mouvement fonctionne déjà bien pour les séquences lentes et le raffinement de mouvement avec un modèle non précis est plus susceptible de capturer le bruit que le vrai mouvement. Pour les séquences rapides, l'interpolation temporelle, MCTI, est très imprécise, et donc, même un raffinement basé sur un motif non précis est capable d'améliorer la qualité d'interpolation. En effet, le premier passage de raffinement utilisant le modèle *4D* donne une amélioration significative pour les séquences vidéo `Soccer` et `Foreman`.

GOP = 2					
	Groupe 1	Groupe 2	$\Delta PSNR_{12}^{4D}$	Groupe 3	$\Delta PSNR_{13}^{EC}$
`Foreman`	29.558	31.917	**2.359**	33.357	**3.799**
`Coastguard`	31.761	29.306	**-2.455**	32.882	**1.121**
`Hall monitor`	36.084	35.82	**-0.264**	36.792	**0.708**
`Soccer`	21.678	25.167	**3.489**	27.533	**5.855**
`Carphone`	29.353	31.223	**1.870**	32.637	**3.284**
Moyenne	29.686	30.686	**1.00**	32.640	**2.954**
GOP = 4					
	Groupe 1	Groupe 2	$\Delta PSNR_{12}^{4D}$	Group 3	$\Delta PSNR_{13}^{EC}$
`Foreman`	27.78	30.918	**3.138**	32.353	**4.573**
`Coastguard`	30.144	28.361	**- 1.783**	31.777	**1.633**
`Hall monitor`	34.902	34.751	**-0.151**	35.93	**1.028**
`Soccer`	20.509	24.024	**3.515**	26.338	**5.829**
`Carphone`	28.842	30.833	**1.991**	32.262	**3.420**
Moyenne	28.435	29.777	**1.342**	31.732	**3.297**
GOP = 8					
	Groupe 1	Groupe 2	$\Delta PSNR_{12}^{4D}$	Group 3	$\Delta PSNR_{13}^{EC}$
`Foreman`	26.407	30.033	**3.626**	31.523	**5.116**
`Coastguard`	29.082	27.782	**-1.30**	30.956	**1.874**
`Hall monitor`	34.053	34.106	**0.053**	35.398	**1.345**
`Soccer`	19.908	23.484	**3.576**	25.741	**5.833**
`Carphone`	28.202	30.455	**2.253**	31.920	**3.718**
Moyenne	27.530	29.172	**1.642**	31.107	**3.577**

Tableau 8.7: *PSNR (en dB) des 3 groupes de blocs de la trame interpolée de l'architecture progressive avec un deux passages de raffinement.* $\Delta PSNR_{12}^{4D}$ *: différence du PSNR entre le groupe 2 après raffinement utilisant le motif* 4D *et le groupe 1 sans raffinement.* $\Delta PSNR_{13}^{EC}$ *: différence du PSNR entre le groupe 3 après raffinement utilisant le motif* EC *et le groupe 1 sans raffinement.*

Qualité de l'information latérale de l'architecture progressive avec trois passages de raffinement

Pour l'architecture progressive avec trois passages de raffinement, les valeurs du PSNR des quatre groupes de blocs sont fournis dans le tableau 8.8. Les deux premiers passages de raffinement donnent des résultats similaires à ceux obtenus avec le schéma progressif précédent avec trois groupes. Vu que le troisième passage de raffinement utilise le motif *AN* composé de tous les blocs voisins, il est capable d'assurer un meilleur raffinement de mouvement, en particulier pour les séquences rapides. Cependant pour les séquences lentes, on constate que le troisième passage de raffinement n'améliore pas l'interpolation MCTI.

Performances débit-distorsion des différentes architectures progressives

Pour l'analyse débit-distorsion (*Rate-Distorsion* (RD)), le codec DVC Discover avec codage turbo, est *réimplémenté* respectant les mêmes conditions de simulation précisées dans [5]. Des résultats de simulation similaires sont obtenus comme on peut le voir aux figures 8.9,8.10 and 8.11. Les résultats de performances de Discover sont obtenus à partir de [5].

Le mécanisme progressif proposé est incorporé au niveau du codec Discover réimplémenté. Une récapitulation de la qualité de la trame interpolée obtenue avec les différents schémas DVC est donnée aux tableaux 8.9 pour un GOP de taille 2, 4 et 8. Le premier schéma, identifié comme BiMESS, réfère à l'interpolation utilisée dans Discover qui consiste en l'estimation de mouvement bidirectionnel avec lissage spatial (BiMESS). Les schémas restants se rapportent à la technique DVC progressive avec les 3 arrangements de blocs proposés. La moyenne du PSNR sur l'ensemble des trames interpolées avec ses différents groupes de blocs est évaluée. La qualité d'interpolation augmente au fur et à mesure que les passages de raffinement sont effectués, à l'exception du cas des séquences vidéo à faible vitesse (i.e. `Hall monitor`). L'interpolation dans les architectures DVC a un impact direct sur la fonction R-D comme on le voit aux figures 8.9, 8.10 et 8.11. Ces figures rapportent les performances des différents schémas DVC et de certains schémas vidéo conventionnels n'effectuant aucune estimation de mouvement du côté du codeur :

H264/AVC no motion I- (*GOP*-1) x B -I : ce schéma exploite, bidirectionnellement, la redondance spatiale inter-trames (entre la trame courante et les trames avoisinantes) sans effectuer aucune estimation de mouvement.

H264/AVC intra I-I-I : Chaque trame est encodée indépendamment des trames avoisinantes sans aucune exploitation de la redondance spatiale inter-trames. La corrélation spatiale intra-trame est, cependant, explorée (au niveau de la trame elle-même).

Discover (Code Turbo) [5] : Les valeurs sont tirées de la page web de Discover [5] dont le codec Slepian-Wolf est basé sur le codage turbo.

Discover (réimplémenté) : Réimplémentation du codec de référence Discover basé sur le codage turbo et tenant compte des mêmes conditions de simulation se trouvant dans [5].

GOP = 2							
	Groupe 1	Groupe 2	$\Delta PSNR_{12}^{4D}$	Groupe 3	$\Delta PSNR_{13}^{EC}$	Groupe 4	$\Delta PSNR_{14}^{AN}$
Foreman	29.558	31.917	**2.359**	33.402	**3.844**	33.61	**4.052**
Coastguard	31.761	29.306	**-2.455**	33.167	**1.406**	32.703	**0.942**
Hall monitor	36.084	35.82	**-0.264**	36.976	**0.892**	36.724	**0.640**
Soccer	21.678	25.167	**3.489**	27.517	**5.839**	27.782	**6.104**
Carphone	29.353	31.223	**1.870**	32.476	**3.123**	33.079	**3.726**
Moyenne	29.686	30.686	**1.00**	32.707	**3.021**	32.779	**3.093**
GOP = 4							
	Groupe 1	Groupe 2	$\Delta PSNR_{12}^{4D}$	Groupe 3	$\Delta PSNR_{13}^{EC}$	Groupe 4	$\Delta PSNR_{14}^{AN}$
Foreman	27.78	30.918	**3.138**	32.399	**4.619**	32.695	**4.915**
Coastguard	30.144	28.361	**-1.783**	32.12	**1.976**	31.586	**1.442**
Hall monitor	34.902	34.751	**-0.151**	36.055	**1.153**	35.93	**1.028**
Soccer	20.509	24.024	**3.515**	26.34	**5.831**	26.547	**6.038**
Carphone	28.837	30.838	**2.001**	32.103	**3.266**	32.733	**3.896**
Moyenne	28.434	29.778	**1.3440**	31.803	**3.3690**	31.898	**3.464**
GOP = 8							
	Groupe 1	Groupe 2	$\Delta PSNR_{12}^{4D}$	Groupe 3	$\Delta PSNR_{13}^{EC}$	Groupe 4	$\Delta PSNR_{14}^{AN}$
Foreman	26.407	30.033	**3.626**	31.551	**5.144**	31.921	**5.514**
Coastguard	29.082	27.782	**-1.30**	31.278	**2.196**	30.78	**1.698**
Hall monitor	34.053	34.106	**0.053**	35.204	**1.151**	35.15	**1.097**
Soccer	19.908	23.4844	**3.5764**	25.726	**5.818**	25.964	**6.056**
Carphone	28.203	30.449	**2.246**	31.8	**3.597**	32.375	**4.172**
Moyenne	27.531	29.170	**1.639**	31.112	**3.581**	31.238	**3.707**

Tableau 8.8: *PSNR (en dB) des 4 groupes de blocs de la trame interpolée de l'architecture progressive avec un trois passages de raffinement.* $\Delta PSNR_{12}^{4D}$ *: différence du PSNR entre le groupe 2 après raffinement utilisant le motif* 4D *et le groupe 1 sans raffinement.* $\Delta PSNR_{13}^{EC}$ *: différence du PSNR entre le groupe 3 après raffinement utilisant le motif* EC *et le groupe 1 sans raffinement.* $\Delta PSNR_{14}^{AN}$ *: différence du PSNR entre le groupe 4 après raffinement utilisant le motif* AN *et le groupe 1 sans raffinement.*

Martins *et al.* : Le système de raffinement de l'information latérale (SIR) proposé par Martins *et al.* [75].

Deligiannis *et al.* : Le schéma proposé par Deligiannis *et al.* [77]. La partie WZ de ce codec est basée sur les codes LDPC qui donne une meilleure compression que le décodeur SW basée sur les turbo-codes. Pour cette raison, la métrique Bjøntegaard de ce schéma est calculée en prenant en compte le codec DISCOVER basée sur les codes LDPC.

Proposé (2, 3 ou 4 groupes) : Le schéma progressif proposé avec 2, 3 et 4 groupes de blocs.

L'amélioration des performances R-D obtenue par l'architecture progressive est intimement liée à la qualité de l'interpolation. Pour les valeurs élevées du groupe d'images (GOP), l'amélioration obtenue par le raffinement de l'information latérale (SIR) est plus importante : l'impact de l'absence de corrélation temporelle lors de la génération de l'information latérale est atténué par l'exploitation de la corrélation spatiale. Le SIR basé sur le codage progressif s'est avérée plus efficace pour les séquences rapides, montrant une amélioration allant jusqu'à 3 dB, par exemple pour la séquence `Foreman` avec le GOP = 8 et le schéma progressif avec 4 groupes. Pour les séquences à mouvements faibles, comme la séquence `Hall monitor`, cependant, le système progressif n'apporte pas les améliorations de performance notable, même lorsque la taille du GOP augmente.

D'après le tableau 8.9, on peut vérifier que la qualité des images interpolées est légèrement améliorée par le schéma progressif pour la séquence `Hall monitor`. Cependant, cette amélioration légère ne se traduit pas par une amélioration dans les courbes de débit-distorsion présentées aux figures 8.9,8.10 et 8.11. Ceci peut être expliqué à partir d'un point de vue de codage de canal : lorsque la trame est subdivisée en plusieurs sous-ensembles, la longueur du turbo code et de son entrelaceur sont plus petits. Par conséquent, la capacité de correction d'erreurs du code turbo diminue.

La technique SIR du schéma progressif proposé a conduit à une amélioration par rapport à la technique SIR de Martins *et. al.* [75] allant jusqu'à 1.2 dB (`Foreman` avec GOP = 8). En effet le raffinement de mouvement dans l'architecture progressive est effectué en utilisant des motifs complètement reconstruits comprenant ainsi les différentes composantes DCT et pas seulement un sous-ensemble de ces bandes comme dans la technique SIR de Martins. Le concept du schéma progressif est inspiré du principe de la modulation différentielle par impulsion codée (*Differential Pulse Code Modulation* (DPCM)). Toutefois, étant donné que le codec WZ nécessite de longs codes, il n'est pas possible d'appliquer la prédiction linéaire, pixel par pixel, en utilisant les pixels voisins spatialement corrélés. La décomposition de la trame dans le schéma progressif doit alors résulter en un certain nombre de groupes non seulement spatialement corrélées mais aussi de longueur importante pour s'assurer de la capacité de correction du codage canal. Contrairement au DPCM, où la corrélation spatiale est utilisée pour prédire le pixel à venir, dans l'architecture DVC progressive proposée la corrélation spatiale est utilisée pour corriger les vecteurs de mouvement, conduisant ainsi à une meilleure exploitation des propriétés de la corrélation temporelle au niveau de la séquence vidéo. Dans le tableau 8.10, les performances globales de RD des différents systèmes SIR sont évalués à l'aide de la métrique Bjøntegaard pour la variation du PSNR et du débit binaire [6]. Ces indicateurs sont calculés en prenant comme

GOP = 2				
	BiMESS	Prog. DVC (2 groupes)	Prog. DVC (3 groupes)	Prog. DVC (4 groupes)
Foreman	29.549	31.393	32.047	32.122
Coastguard	31.675	32.315	31.708	31.734
Hall Monitor	36.138	36.455	36.372	36.401
Soccer	21.698	24.592	25.478	25.536
Carphone	29.271	30.90	31.46	31.53
Moyenne	**29.666**	**31.131**	**31.413**	**31.464**
GOP = 4				
	BiMESS	Prog. DVC (2 groupes)	Prog. DVC (3 groupes)	Prog. DVC (4 groupes)
Foreman	27.721	30.063	30.851	30.95
Coastguard	30.136	30.927	30.515	30.555
Hall Monitor	35.007	35.42	35.194	35.41
Soccer	20.501	23.405	24.302	24.353
Carphone	28.716	30.451	31.049	31.128
Moyenne	**28.416**	**30.053**	**30.382**	**30.479**
GOP = 8				
	BiMESS	Prog. DVC (2 groupes)	Prog. DVC (3 groupes)	Prog. DVC (4 groupes)
Foreman	26.37	28.674	29.499	29.981
Coastguard	29.038	30.004	28.901	29.731
Hall Monitor	34.227	34.729	34.739	34.484
Soccer	19.896	23.057	23.154	23.778
Carphone	28.068	29.951	30.624	30.707
Moyenne	**27.519**	**29.283**	**29.383**	**29.736**

Tableau 8.9: *Qualité d'interpolation en PSNR (dB) pour un GOP*$= 2, 4,$ *et 8.*

référence le système sans raffinement de l'information latérale, Discover. Les architectures DVC avec raffinement, considérées ici, utilisent le codage turbo et elles sont par conséquent comparées par rapport au système DISCOVER utilisant le codage turbo. Cependant, la technique de Deligiannis [77] utilise les codes LDPC et est alors comparée au système Discover utilisant les codes LDPC. En se basant sur les mesures de PSNR et de débit de Bjøntegaard [6], on observe que le codec progressif proposé réalise des gains de compression allant jusqu'à 31.3859 % de réduction du débit par rapport à Discover. En outre, le système progressif avec 4 groupes dépasse les performances de la technique SIR de Martins [75] pour toutes les différentes configurations . En comparaison avec la technique de raffinement de Deligiannis [77], le système progressif avec 4 groupes donne de meilleurs résultats pour la majorité des scénarios, à l'exception de la séquence QCIF `soccer`.

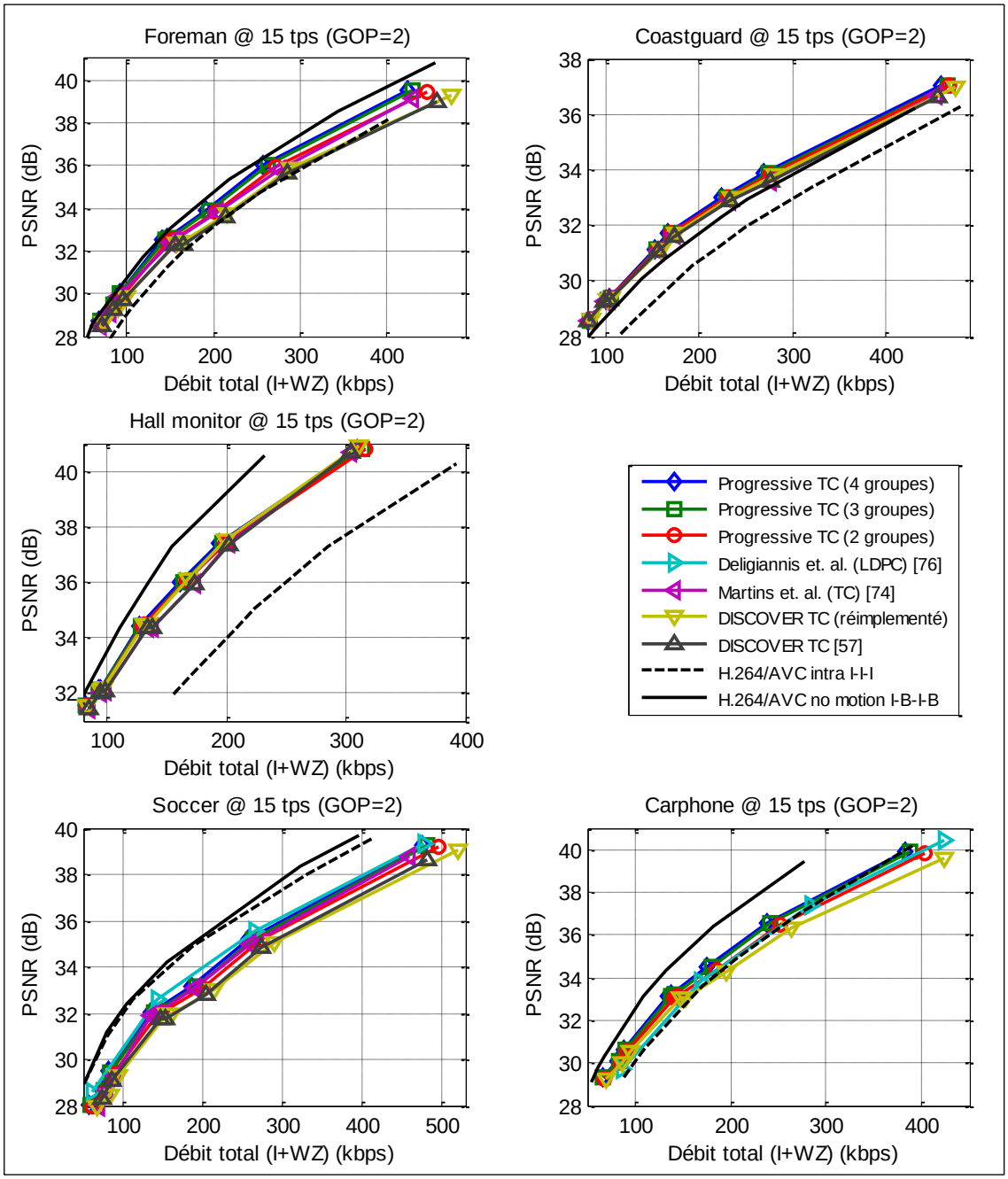

FIGURE 8.9: *Analyse débit-distorsion pour un GOP=2.*

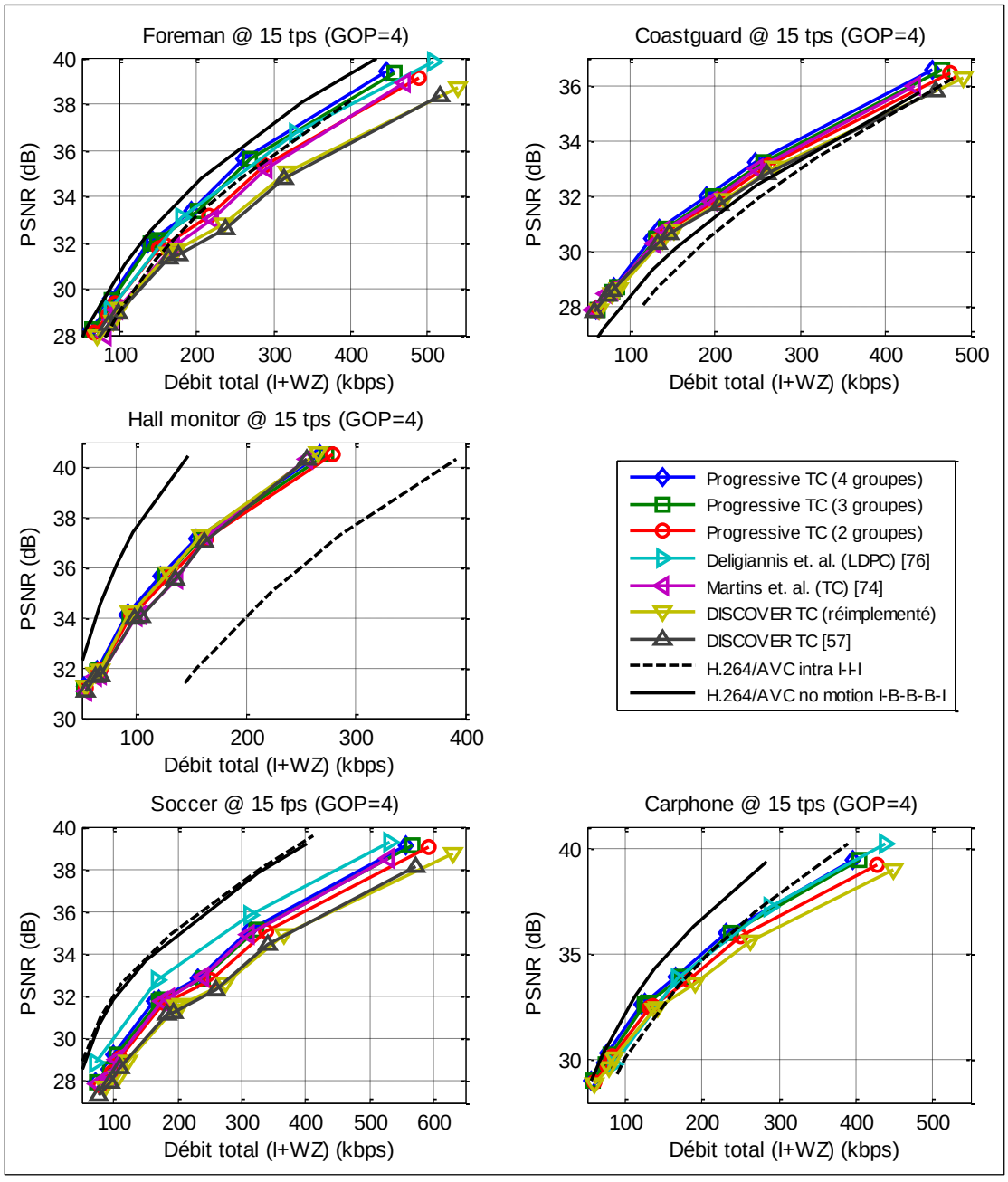

FIGURE 8.10: *Analyse débit-distorsion pour un GOP=4.*

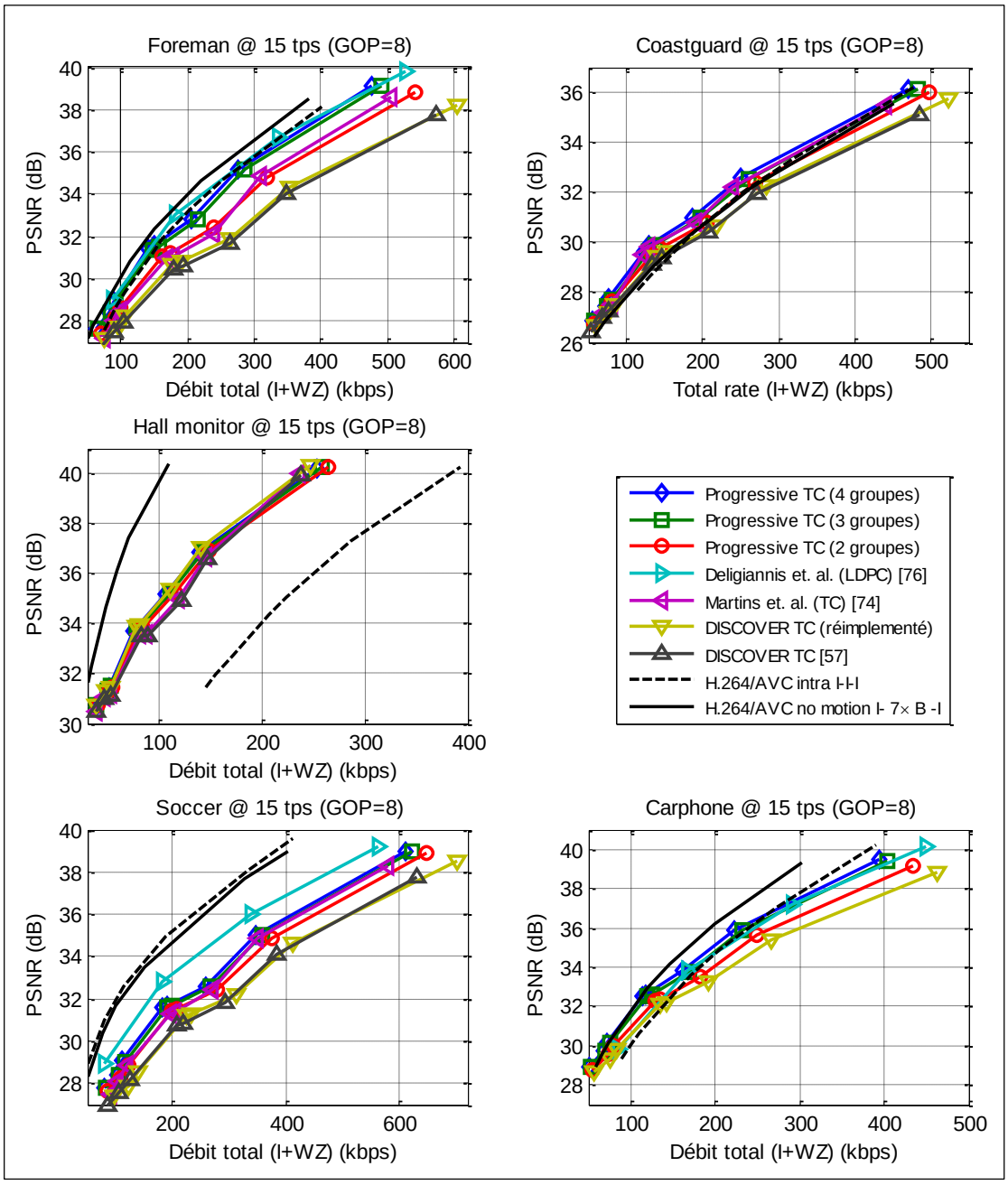

FIGURE 8.11: *Analyse débit-distorsion pour un GOP=8.*

		GOP 2		GOP 4		GOP 8	
Séquence	technique SIR	ΔR (%)	ΔPSNR (dB)	ΔR (%)	ΔPSNR (dB)	ΔR (%)	ΔPSNR (dB)
Foreman	Prog. 4 sets	**-13.1935**	**0.8249**	**-24.0237**	**1.4995**	-31.3859	2.0677
	Prog. 3 sets	-11.3440	0.6997	-21.0332	1.2848	-28.0933	1.8069
	Prog. 2 sets	-6.9061	0.4149	-12.7094	0.7371	-15.6575	0.9305
	Deligiannis et al.	-4.7693	0.3162	-20.8418	1.3473	**-34.1192**	**2.3699**
	Martins et al.	-5.6930	0.3644	-7.1264	0.4419	-14.3356	0.8285
Soccer	Prog. 4 sets	-14.1737	0.8376	-19.2194	1.2067	-22.0967	1.4351
	Prog. 3 sets	-11.9026	0.6945	-16.4910	1.0198	-19.3024	1.2328
	Prog. 2 sets	-7.4463	0.4279	-10.5923	0.6393	-13.9559	0.8687
	Deligiannis et al.	**-17.1781**	**0.9931**	**-30.2909**	**1.9246**	**-38.9008**	**2.6557**
	Martins et al.	-9.0854	0.5333	-15.2502	0.9139	-15.8318	0.9906
Coastguard	Prog. 4 sets	**-4.3803**	**0.2189**	**-10.8566**	**0.4840**	**-17.5782**	**0.8257**
	Prog. 3 sets	-2.5786	0.1271	-7.5031	0.3267	-13.4620	0.6154
	Prog. 2 sets	-2.1362	0.1060	-4.5434	0.1947	-7.8475	0.3476
	Martins et al.	0.3071	-0.0067	-7.3185	0.2890	-13.2512	0.6355
Hall monitor	Prog. 4 sets	**-1.9400**	**0.1437**	**-2.1744**	**0.1359**	**0.6280**	**-0.0233**
	Prog. 3 sets	-0.3869	0.0311	1.1726	-0.0605	2.7001	-0.1306
	Prog. 2 sets	0.6052	-0.0383	3.8789	-0.2132	6.6494	-0.3268
	Martins et al.	3.4073	-0.2491	5.5937	-0.3381	9.5248	-0.4840
Carphone	Prog. 4 sets	**-11.9046**	**0.7508**	**-16.6542**	**0.9418**	**-22.6768**	**1.2945**
	Prog. 3 sets	-10.0661	0.6258	-13.7520	0.7611	-18.9992	1.0578
	Prog. 2 sets	-5.8555	0.3546	-6.6457	0.3519	-9.4964	0.4989
	Deligiannis et al.	-2.2857	0.1941	-6.3553	0.4628	-16.3032	1.0443

Tableau 8.10: *Métriques de Bjøntegaard [6] pour les différentes technique de raffinement de l'information latérale (SIR) en considérant le système DVC de Discover comme référence.*

Analyse de la complexité calculatoire

Dans cette section, la complexité d'encodage et de décodage de l'architecture progressive proposée avec 4 groupes est étudiée. La complexité est évaluée en mesurant le temps d'exécution requis par l'encodeur et le décodeur à l'aide d'un ordinateur personnel avec un processeur Intel Core i7 CPU à 2.67 GHz et avec 12 GB de RAM.

Le tableau 8.11 montre le temps d'exécution d'encodage en secondes pour trois schémas de compression pour l'ensemble des trames des séquences Foreman et texttt Soccer pour $GOP = 2$ et 8 :

1. Standard conventionnel H264/AVC intra (T^{enc}_{Intra}) : Quatre valeurs du paramètre de quantification (QP) sont considérées. Le même encodeur génère les images clés utilisées par le système DVC progressive et le système DVC de DISCOVER.
2. L'encodeur DVC de l'état de l'art DISCOVER (T^{enc}_{DISC}) : Quatre valeurs de l'indice de quantification (Q_i) sont considérées. Le temps d'exécution est la somme du temps d'encodage de l'image clé et du temps d'encodage et de la trame WZ : $T^{enc}_{DISC} = T^{Key} + T^{WZ}_{DISC}$.
3. L'encodeur DVC progressif proposé avec 4 groupes : $T^{enc}_{Prog4} = T^{Key} + T^{WZ}_{Prog4}$.

	H264/AVC intra				DISCOVER			Prog. (4 groupes)			Rapports	
	QP	T^{enc}_{Intra}		Q_i	T^{Key}	T^{WZ}_{DISC}	T^{enc}_{DISC}	T^{WZ}_{Prog4}	T^{enc}_{Prog4}		$\frac{T^{enc}_{Prog4}}{T^{enc}_{DISC}}$	$\frac{T^{enc}_{Prog4}}{T^{enc}_{Intra}}$
Foreman GOP 2	40	27.941		1	14.659	1.14	**15.79**	2.36	**17.02**		**1.08**	**0.61**
	34	29.630		5	15.551	1.59	**17.14**	3.47	**19.02**		**1.11**	**0.64**
	29	31.263		7	16.451	1.81	**18.26**	4.10	**20.55**		**1.12**	**0.62**
	25	33.291		8	17.491	2.01	**19.50**	4.39	**21.88**		**1.12**	**0.65**
Foreman GOP 8	40	27.941		1	3.371	2.10	**5.47**	5.07	**8.44**		**1.54**	**0.30**
	34	29.630		5	3.581	2.82	**6.40**	6.14	**9.72**		**1.52**	**0.33**
	29	31.263		7	3.817	3.01	**6.83**	6.47	**10.29**		**1.51**	**0.33**
	25	33.291		8	4.105	3.52	**7.62**	8.03	**12.13**		**1.59**	**0.36**
Soccer GOP 2	44	27.971		1	14.281	1.46	**15.74**	3.31	**17.59**		**1.12**	**0.63**
	36	29.563		5	15.211	1.75	**16.96**	3.82	**19.03**		**1.12**	**0.64**
	31	31.251		7	16.121	1.90	**18.02**	4.31	**20.43**		**1.13**	**0.65**
	25	33.270		8	17.513	2.03	**19.54**	4.81	**22.32**		**1.14**	**0.67**
Soccer GOP 8	44	27.971		1	3.271	2.16	**5.43**	4.96	**8.23**		**1.51**	**0.29**
	36	29.563		5	3.471	3.08	**6.55**	6.84	**10.31**		**1.57**	**0.35**
	31	31.251		7	3.731	3.29	**7.02**	7.29	**11.02**		**1.57**	**0.35**
	25	33.270		8	4.091	3.61	**7.70**	8.30	**12.391**		**1.61**	**0.37**

Tableau 8.11: *Temps d'encodage en secondes de la technique progressive proposée avec 4 groupes (T^{enc}_{Prog4}) en comparaison avec DISCOVER (T^{enc}_{DISC}) et H264/AVC intra (T^{enc}_{Prog4}).*

Le rapport $\frac{T^{enc}_{prog4}}{T^{enc}_{DISC}}$ indique que la complexité de l'encodeur progressif dépasse la complexité du codeur DISCOVER d'environ 12 % pour GOP $= 2$ et environ 57 % pour GOP $= 8$. La complexité supplémentaire est principalement due à la subdivision de la trame en 4 parties et le lancement 4 fois du processus d'encodage avec les 2 codeurs convolutifs récursifs systématiques (RSC) formant l'encodeur turbo. Malgré cette augmentation de la complexité du codage, la technique progressive proposée respecte toujours l'objet du paradigme DVC, à savoir, la réduction de la complexité de codage par rapport à celle du standard H264/AVC intra : les rapports $\frac{T^{enc}_{Prog4}}{T^{enc}_{Intra}}$ dans le tableau 8.11 indiquent une réduction de la complexité de près de 40 % et 70 % pour GOP $= 2$ et 8, respectivement.

Quant à la complexité de décodage, le tableau 8.12 fournit les temps d'exécution de décodage pour les architectures de DISCOVER et progressive. Le temps global de décodage englobe le temps d'exécution de la génération de l'information latérale, T_{SIG}, et le temps d'exécution du décodeur Slepian-Wolf, T_{SW}, (décodage turbo) : $T^{dec} = T^{SIG} + T^{SW}$. Ce tableau montre que la complexité de calcul supplémentaire du décodeur proposé, en raison des multiples passages de raffinement de l'information latérale, est compensée par un décodage turbo plus rapide. Par exemple, pour le système progressif avec 4 groupes, le plan de bits est de longueur 1584 est divisé en quatre parties menant à quatre plans de bits de longueur 396 : le processus de décodage turbo du plan de bits de longueur 1584 est plus complexe que le turbo décodage des quatre plans de bits de longueur 396.

		DISCOVER			Prog. (4 groupes)			Rapports		
	Qi	T_{DISC}^{SIG}	T_{DISC}^{SW}	T_{DISC}^{dec}	T_{Prog4}^{SIG}	T_{Prog4}^{SW}	T_{Prog4}^{dec}	$\frac{T_{Prog4}^{SIG}}{T_{DISC}^{SIG}}$	$\frac{T_{Prog4}^{SW}}{T_{DISC}^{SW}}$	$\frac{T_{Prog4}^{dec}}{T_{DISC}^{dec}}$
Foreman GOP 2	1	53.54	339.0	**392.6**	217.79	403.1	**620.9**	4.07	1.19	**1.58**
	5	53.06	849.0	**902.1**	217.94	848.2	**1066.2**	4.11	0.99	**1.18**
	7	53.02	1702.0	**1755.1**	217.92	1595.5	**1813.4**	4.11	0.94	**1.03**
	8	53.15	2858.6	**2911.7**	217.03	2681.0	**2898.0**	4.11	0.94	**0.99**
Foreman GOP 8	1	88.73	829.4	**918.1**	358.68	977.1	**1335.8**	4.04	1.18	**1.45**
	5	88.65	2075.1	**2163.8**	358.75	2042.5	**2401.3**	4.05	0.98	**1.11**
	7	88.67	4115.5	**4204.2**	360.60	3617.1	**3977.7**	4.06	0.89	**0.94**
	8	88.73	6696.9	**6785.6**	358.93	5633.0	**5991.9**	4.04	0.84	**0.88**
Soccer GOP 2	1	53.79	490.38	**544.17**	218.18	532.59	**750.77**	4.05	1.09	**1.38**
	5	53.08	1108.59	**1161.67**	217.54	1165.73	**1383.27**	4.10	1.05	**1.19**
	7	53.64	2126.88	**2180.52**	217.23	1952.67	**2169.90**	4.05	0.92	**0.99**
	8	53.24	3357.65	**3410.89**	217.39	2945.31	**3162.70**	4.08	0.88	**0.93**
Soccer GOP 8	1	89.84	945.46	**1035.30**	364.35	989.26	**1353.61**	4.05	1.05	**1.31**
	5	89.86	2172.66	**2262.52**	362.52	2014.83	**2377.35**	4.03	0.93	**1.05**
	7	89.17	4330.37	**4419.54**	364.02	3756.27	**4120.29**	4.08	0.87	**0.93**
	8	89.76	6972.57	**7062.33**	365.24	5102.67	**5467.91**	4.07	0.73	**0.77**

Tableau 8.12: *Comparaison de la compléxité de décodage (en secondes) entre le schéma progressif proposé avec 4 groupes ($T_{Prog4}^{dec} = T_{Prog4}^{SIG} + T_{Prog4}^{SW}$) et DISCOVER ($T_{DISC}^{dec} = T_{DISC}^{SIG} + T_{DISC}^{SW}$).*

8.5 Conclusion

Une nouvelle architecture de codage vidéo distribué est présentée dans ce chapitre. Cette architecture est axée sur le codage progressif consistant à diviser la trame Wyner-Ziv en des groupes spatialement corrélées. Ces groupes sont ensuite envoyés progressivement et décodés progressivement. Chaque groupe décodé est spatialement corrélé avec les groupes subséquents et peut donc être exploité afin d'améliorer le processus de l'estimation de mouvement à travers des passages de raffinement. Cette méthode répond parfaitement au paradigme distribué et n'implique pas une complexité supplémentaire de codage. En outre, les passages de raffinement sont basés sur des motifs de bonne qualité qui sont en mesure d'améliorer progressivement l'estimation des vecteurs de mouvement. Pour démontrer la faisabilité de la méthode proposée, le schéma progressif est incorporé dans le codec DVC d'état de l'art Discover. Des améliorations significatives ont été obtenues, en particulier pour les séquences vidéo rapides et pour des valeurs de groupes d'images élevées. La qualité d'interpolation est grandement raffinée et une amélioration allant jusqu'à 3 dB est rapportée concernant les performances de débit distorsion.

Chapitre 9

Poinçonnage adaptatif dans le domaine des transformées et minimisation des délais de décodage

9.1 Introduction

Dans ce chapitre, deux directions indépendantes ont été explorées pour l'amélioration de l'architecture de codage vidéo distribué. La première direction se rapporte au mécanisme de poinçonnage dans le domaine des transformées et fait appel à deux nouvelles techniques :

1. La première technique vise à améliorer les performances R-D en réduisant les cas d'expansion par plan de bits. Elle consiste à utiliser efficacement des bits systématiques c'est-à-dire lorsque l'information latérale est fortement bruitée. Dans ces circonstances, l'utilisation de bits de parité uniquement n'est pas efficace pour la convergence du décodeur turbo et les bits systématiques peuvent être aussi envoyés. L'alternance adaptative entre les bits de parité et les bits systématiques est gérée par le changement de la matrice de poinçonnage en fonction du nombre de bits cumulatifs envoyés. Cette technique a été publiée dans [78].

2. La deuxième technique est une extension de la première et vise à diriger les bits de parité ainsi que les bits systématiques dans les régions les plus bruitées de chaque bande DCT. Cette technique fait l'objet d'un article de conférence [79].

Tandis que la première direction vise à améliorer les performances R-D, la deuxième direction, quant à elle, touche à un aspect plutôt pratique, à savoir la réduction des délais de décodage. Deux techniques d'estimation du débit sont développées afin de réduire le nombre de demandes de bits de parité envoyées par le canal de retour et, par conséquent, le nombre de lancements du processus de décodage turbo. Ces estimateurs présentent une précision supérieure en comparaison avec les estimateurs présentés dans la littérature réduisant de la sorte les situations indésirables de surestimation et de sous-estimation du nombre de fragments de bits de parité requis. Les techniques d'estimation propo-

sées apparaissent dans un article de conférence [80] qui a été développé ultérieurement dans un article de journal [81].

9.2 Utilisation adaptative des bits systématiques en variant la matrice de poinçonnage

Dans le système de codage vidéo distribué, l'information latérale est considérée comme une version bruitée des données originales : les techniques de codage de canal, comme les codes turbo, sont utilisées pour éliminer les erreurs. Le turbo code génère des bits de parité qui sont envoyés au décodeur afin de corriger les erreurs se trouvant dans l'information latérale (SI). Les bits systématiques, cependant, ne sont pas envoyés vu que le décodeur dispose déjà de l'information latérale qui peut être utilisée pour le calcul des rapports de vraisemblance du canal. Toutefois, lorsque l'information latérale est fortement bruitée, les rapports de vraisemblance canal calculés seront erronés entravant ainsi la convergence du décodage turbo. Ainsi un nombre élevé de bits de parité, dépassant même la longueur des données originales, est nécessaire pour renverser l'effet trompeur d'une information latérale fortement corrompue. Ceci conduit à un effet indésirable d'expansion où le nombre de bits de parité dépasse la longueur du plan de bits à envoyer. Dans cette section, nous proposons une méthode pour détecter de manière adaptative les situations où l'information latérale est de mauvaise qualité. Ensuite quatre matrices de poinçonnage différentes sont utilisées pour assurer l'envoi adaptatif des bits systématiques en plus des bits de parité. Des rapports de vraisemblance de canal fiables peuvent ainsi être calculés. Même si l'on ne rapporte qu'une amélioration de 0.5 dB pour les performances de débit-distorsion, dans certains plans de bits une réduction significative du débit est obtenue et l'effet d'expansion est évité.

9.2.1 Schéma de poinçonnage conventionnel dans les systèmes DVC

Le mécanisme de poinçonnage conventionnel dans l'architecture DVC est utilisé pour éliminer des bits systématiques et envoyer périodiquement les bits de parité relatifs aux deux encodeurs RSC. Un exemple d'allocation des bits de parité est présenté à la figure 9.1 où la longueur du plan de bits est de 1584 bits et la période de poinçonnage est de 48. Pour le premier RSC un nombre de $n_{p_1} = 4$ bits de parité sont envoyés sur une période de 48 bits et pour le second RSC, $n_{p_2} = 3$ bits de parité sont envoyés. Si le décodeur turbo ne converge pas, une requête est envoyée à travers le canal de retour vers la mémoire tampon de l'encodeur pour la transmission d'un bloc supplémentaire de 33 bits de parité. Ces bits sont destinés au second RSC de sorte que n_{p_2} passe de 3 à 4. La transmission des bits de parité est par la suite réitérée jusqu'à convergence du décodeur. Les bits systématiques sont éliminés et le décodeur turbo se base sur l'information latérale (SI) pour calculer les rapports de vraisemblance canal. Pour une mauvaise qualité de la SI, ce schéma de poinçonnage doit être modifié pour permettre l'envoi, en plus des bits de parité, de certains bits systématiques pour compenser la non fiabilité de la SI lors du calcul des rapports de vraisemblance canal.

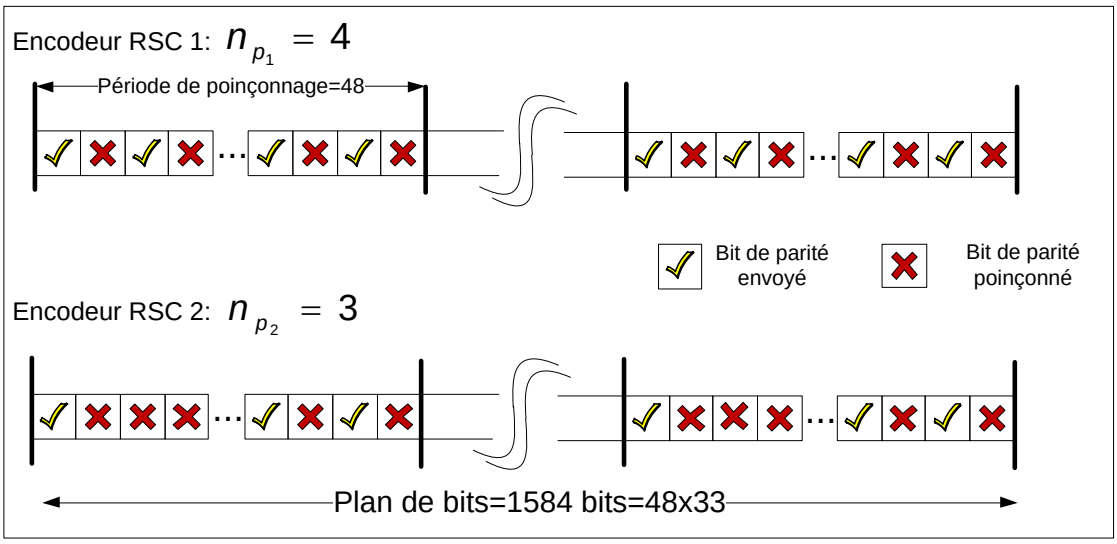

FIGURE 9.1: *Schéma de poinçonnage conventionnel dans les architectures DVC.*

9.2.2 Algorithme proposé pour le poinçonnage adaptatif

La figure 9.2 présente l'algorithme proposé pour effectuer une transmission adaptative des bits systématiques. À la première itération, le décodeur envoie R_{min} bits de parité [1]. Si le décodeur ne converge pas, une requête de bits supplémentaires est envoyée à l'encodeur. Lorsque le nombre cumulatif de bits (R_{min} + le nombre de bits ajoutés) dépasse un certain seuil N_1, l'encodeur change la matrice de poinçonnage pour permettre l'envoi de quelques bits systématiques. Ces bits systématiques, dispersés sur le treillis de décodage, aident au calcul fiable des ratios de vraisemblance, notamment lorsqu'ils coïncident à l'emplacement où l'information latérale est erronée. Cela est plus susceptible de se produire lorsqu'il y a un grand nombre d'erreurs. Dès que le nombre de bits demandé (de parité et systématiques) augmente et atteint le seuil N_2, la matrice de poinçonnage est encore changée pour favoriser davantage la transmission des bits systématiques jusqu'à un certain seuil (N_3) où seuls les bits systématiques sont envoyés.

En se basant sur les résultats de simulation, les bits systématiques sont envoyés de préférence aux mêmes endroits que les bits de parité de telle sorte que le nombre de transitions dans le trellis soit réduit d'un facteur 4. La figure 9.3 montre un exemple de la réception d'un bit systématique de valeur 1 et d'un bit de parité de valeur 1 dans un treillis à 8 états d'un code systématique récursif de taux 1/2. Dans le trellis réduit, seules les transitions où le bit systématique et le bit de parité sont égals à 1, sont maintenues. Si le bit systématique est envoyé ailleurs (où il n'y a pas de bit de parité), le nombre de transitions est divisé par deux, tout comme si un bit de parité avait été reçu.

1. R_{min} désigne une estimation du nombre minimal de bits de parité que l'encodeur envoie en un seul paquet au décodeur.

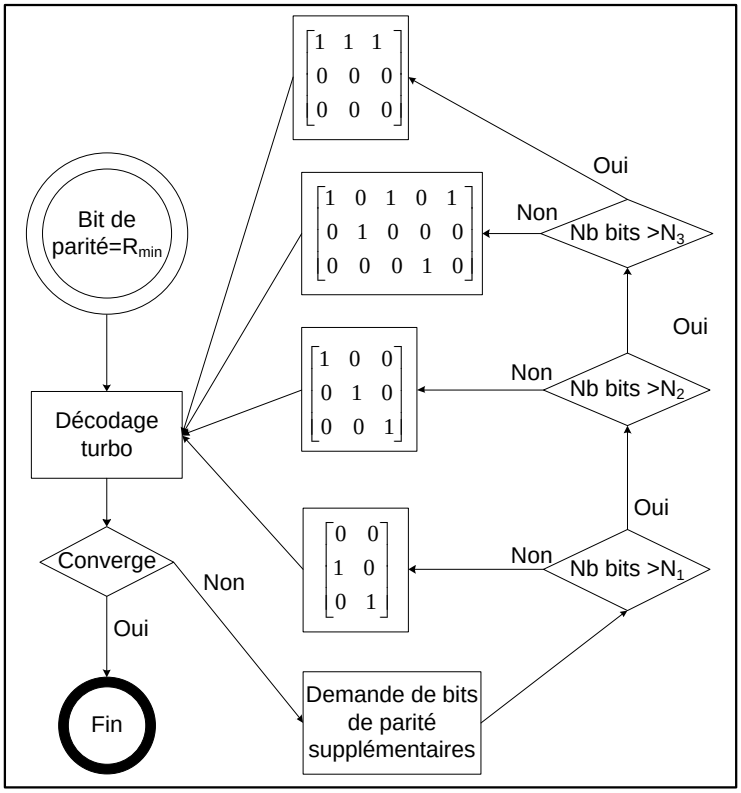

FIGURE 9.2: *Algorithme proposé pour la sélection adaptative des différentes matrices de poinçonnage.*

9.2.3 Implémentation de l'algorithme de variation adaptative de la matrice de poinçonnage

Dans cette sous-section, l'algorithme proposé avec de multiples matrices de poinçonnage est comparé à l'architecture DVC de Discover dont la partie Selpian-Wolf se base sur les codes turbo [82]. Deux séquences vidéo QCIF à 15 images par seconde sont considérées pour les simulations : `Foreman` et `Soccer`. Ces séquences sont téléchargées à partir du site web de Discover [5]. Toutes les 149 trames des séquences sont prises en compte. La taille de la trame est de 144 $\times$ 176 = 25344 pixels, menant à des plans de bits de longueur de 25344/16 = 1584 bits pour chaque composant DCT 4 $\times$ 4. Sur la base de tests préliminaires, les paramètres de l'algorithme (voir figure 9.2) sont fixés à : $N_1 = 825$, $N_2 = 990$ et $N_3 = 1320$. Les seuils N_i sont choisis en fonction d'une procédure d'étalonnage pour déterminer quand les bits de parité, à eux seuls, ne suffisent plus et l'utilisation de bits systématiques est recommandée.

Le Tableau 9.1 indique le nombre d'erreurs pour chacun des plans de bits de quelques composantes DCT de la quatrième trame de la séquence `soccer`. Il affiche aussi le nombre de bits requis pour les corriger. Pour mettre en évidence la compression obtenue par l'algorithme proposé, on considère un indice quantification élevé, $Qi = 8$, et un groupe d'image GOP = 8. Pour la composante de DC, le nombre d'erreurs dans l'information latérale est important. Par conséquent, le codec Discover néces-

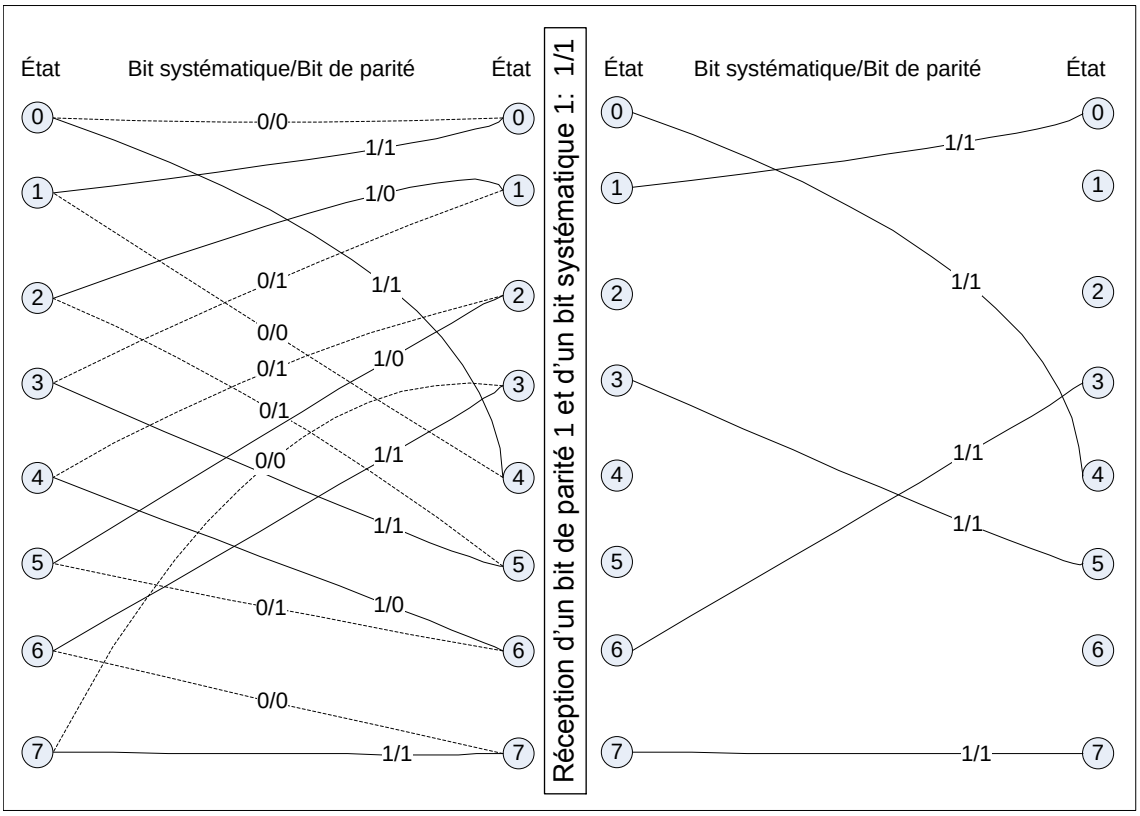

FIGURE 9.3: *Transitions entre les états d'un trellis à 8 états d'un code systématique récursif de taux 1/2.*

site un grand nombre de bits de parité (dépassant parfois la longueur du plan de bits : d'où le cas indésirable d'expansion) pour contrer le poids des rapports de vraisemblance canal non fiables calculés à partir d'une information latérale corrompue. L'algorithme proposé permet la transmission de certains bits systématiques si le nombre de bits cumulatifs N augmente. Lorsque N dépasse le seuil N_1, les bits systématiques sont envoyés à la même fréquence que chacun des bits de parité des encodeurs RSC. Ensuite, si N continue à augmenter et dépasse N_2, les bits systématiques seront envoyés à la même fréquence que les bits de parité de deux encodeurs RSC. Enfin, si N dépasse N_3, uniquement les bits systématiques sont utilisés et les bits de parité des deux encodeurs RSC sont poinçonnés.

Pour le second plan de bits de la composante AC1, le nombre d'erreurs est faible et donc seulement quelques bits de parité sont nécessaires ($N = 660$). Ce nombre ne dépasse pas le premier seuil $N_1 = 825$. Ainsi, le mécanisme d'envoi de bits systématiques n'est pas activé et l'algorithme proposé fonctionne de façon similaire à Discover. Pour le cinquième plan de bits de la composante AC2, on remarque que le nombre de bits requis par l'algorithme proposé est supérieur au nombre requis par Discover. Dans ce cas, les bits systématiques envoyés sont gaspillés parce que l'information latérale n'est pas fortement erronée et le décodeur est toujours en mesure d'assurer un décodage efficace avec

	Plan de bits	1	2	3	4	5	6	7
DC	# err.	282	168	372	290	478	534	662
Bits envoyés	Discover	1452	1221	1881	1617	2145	2442	2541
	proposé	1386	1155	1617	1452	1749	1749	1782
AC1	# err.	464	69	97	184	281	398	
Bits envoyés	Discover	2277	660	792	1089	1419	1683	
	proposé	1749	660	792	1089	1353	1518	
AC2	# err.	371	46	121	194	313		
Bits envoyés	Discover	2046	528	858	1056	1419		
	proposé	1683	528	858	1023	1485		
AC3	# err.	224	29	80	213			
Bits envoyés	Discover	1386	363	693	1122			
	proposé	1386	363	693	1089			
AC4	# err.	550	36	133	303	395	536	
Bits envoyés	Discover	2442	528	990	1584	1749	2079	
	proposé	1782	528	1023	1518	1617	1716	
AC5	# err.	348	19	71	190	313		
Bits envoyés	Discover	1848	297	594	1089	1551		
	proposé	1584	297	594	1089	1452		
AC8	# err.	393	19	99	202	369		
Bits envoyés	Discover	2013	363	759	1188	1650		
	proposé	1650	363	759	1254	1551		

Tableau 9.1: *Nombre d'erreurs (# err.) et nombre de bits nécessaires pour chaque plan de bits pour quelques composantes DCT. Algorithme proposé (débit* $= 770.1$ *kbps) vs Discover (débit* $= 838.96$ *kbps) pour GOP* $= 8$*. Trame numéro 4 de la séquence vidéo* `soccer`*. Matrice de quantification à haut débit (Qi* $= 8$*).*

seulement des bits de parité. Dans ce tableau, on rapporte uniquement les composantes DCT présentant les différences entre Discover et l'algorithme proposé. Le débit global pour cette trame est de 839 kbps pour Discover et seulement 770 kbps pour l'algorithme proposé, assurant ainsi une réduction du débit de 8%.

Les performances de débit-distorsion sont présentées à la figure 9.4 pour un $GOP = 2$ et un $GOP = 8$. L'algorithme proposé est comparé au codec DVC de Discover avec turbo-code [5]. Les performances du codec standard intra (H.264/AVC) sont également présentées. L'algorithme proposé affiche une réduction du débit en comparaison avec le codec Discover. Une réduction légère du débit binaire (pour le même PSNR) est observée pour les séquences vidéo `Foreman` vu que l'interpolation est capable de fournir une information latérale de bonne qualité et que les bits systématiques peuvent être éliminés. Toutefois, pour la séquence `Soccer` (en particulier avec $GOP = 8$), une amélioration plus avérée est observée vu que l'information latérale est peu fiable et le phénomène indésirable d'expansion des plans de bits se produit plus souvent. En définitive une amélioration de 0.5 dB est notée.

Même si l'amélioration globale est relativement modeste, on constate à partir du tableau 9.1 que la réduction du débit reste considérable pour les plans de bits les moins significatifs (LSB : least significant bits) où l'expansion est plus susceptible de se produire. En outre, pour les séquences vidéo extrêmement rapides, l'architecture DVC risque de ne pas parvenir à décoder les plans de bits LSB, même si

tous les bits de parité des deux encodeurs RSCs sont envoyés. Pour les codecs DVC pratiques, les bits systématiques doivent être utilisés judicieusement. Lorsque l'information latérale est correctement générée, les bits systématiques devraient être écartés pour maintenir des performances compétitives. L'algorithme proposé est principalement conçu pour les situations où la génération d'une information latérale propre est difficile et ce, lorsqu'il s'agit de vidéo à mouvements rapides ou lorsque la génération se fait par extrapolation plutôt que par interpolation.

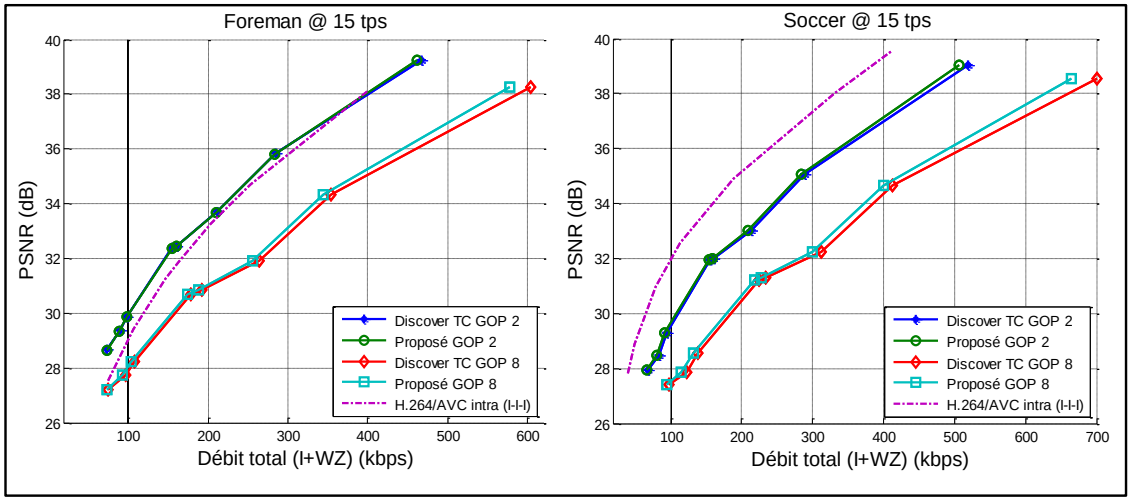

FIGURE 9.4: *Comparaison des performances de débit-distorsion entre le système de poinçonnage proposé et le système Discover TC (turbo coding) .*

9.3 Turbo code utilisant le poinçonnage adaptatif pour le codage vidéo distribué dans le domaine de transformée

Le schéma de poinçonnage conventionnel utilise un motif périodique qui assure la dispersion uniforme des bits de parité sur le treillis du code turbo. Le but des bits de parité est d'affiner l'information latérale générée par une interpolation temporelle avec compensation de mouvement (MCTI : *motion compensated temporal interpolation*). La trame interpolée présente généralement des erreurs dans les endroits où il y a du mouvement. Ainsi, les erreurs sont spatialement regroupées. Le but de ce travail est d'améliorer le mécanisme de poinçonnage par la détection de l'emplacement des erreurs dans la trame interpolée. Ensuite les bits de parité sont dirigés là où ils sont le plus utile au lieu de les disperser uniformément. Le même concept a été adopté par les auteurs [83] dans le domaine des pixels. Cette section étend cette méthode pour l'architecture DVC dans le domaine de la transformée DCT : au lieu de localiser les erreurs dans la trame interpolée, les erreurs sont plutôt localisées dans chaque bande de la DCT de la trame interpolée. Par ailleurs, il a été également démontré [78] que les bits systématiques peuvent être utiles pour supporter les bits de parité lorsque l'information latérale est fortement bruitée (voir la section précedente). Le mécanisme de poinçonnage assure la transmission adaptative des bits

de parité ainsi que des bits systématiques en les dirigeant vers les régions bruitée de l'information latérale.

9.3.1 Mécanisme de poinçonnage conventionnel pour le DVC dans le domaine transformé

Comme le montre la figure 7.1, l'encodeur turbo traite chaque bande DCT quantifiée des trames WZ plan de bits par plan de bits. Pour les séquences vidéo au format QCIF, la taille de la trame est de $144 \times 176 = 25344$ pixels. L'application de la DCT 4×4 conduit à une longueur du plan de bits de $25344/16 = 1584$. Le mécanisme de poinçonnage dans l'architecture DVC conventionnelle élimine des bits systématiques et considère uniquement les bits de parité. Ces bits sont alloués périodiquement aux deux encodeurs RSC. La période de poinçonnage est fixée à 48 et les bits de parité sont transmis selon un motif pseudo-aléatoire au sein de la période. Pour chaque demande via le canal de retour, l'encodeur envoie, alternativement, $1584/48 = 33$ bits (à raison de 1 bit par période) pour les deux encodeurs RSC. Ainsi, les bits de parité sont répartis uniformément à travers le treillis de longueur 1584. La figure 9.5 illustre le fonctionnement du poinçonnage uniforme : elle montre l'emplacement des bits de parité sur un plan de bits d'une bande DCT de taille $144/4 \times 176/4 = 1584$ bits. Le plan de bits est formé par la quantification des coefficients DCT avec une lecture selon l'ordre de balayage de trame indiqué par les flèches rouges (*raster scan*). Cette figure indique la position des bits de parité générés par les deux encodeurs RSC. Il est clair que ces bits sont répartis uniformément le long du treillis.

9.3.2 Algorithme de poinçonnage adaptatif

La figure 9.6 représente la différence entre les composantes de DCT 4×4 d'une trame WZ et des composantes DCT 4×4 de la trame interpolée. On remarque que les erreurs sont regroupées et concentrées dans certaines régions, tandis que les autres régions sont moins bruitées : la corrélation entre l'information originale et l'information latérale est plus élevée. Dans ce qui suit, on introduit un algorithme adaptatif pour une répartition efficace des bits de parité et des bits systématiques. Il consiste essentiellement à propager de façon adaptative les bits en fonction de la répartition des erreurs dans le domaine DCT.

Pour chaque bande de la trame DCT, une répartition différente des bits de parité/systématique est établie en fonction des emplacements des erreurs. La trame des résidus (différences) affichées à la figure 9.6, peut être estimée à partir de l'algorithme d'interpolation BiMESS (voir la section 6.2). Tout d'abord, la trame résiduelle R est générée à partir des versions compensées en mouvement de la trame vers l'arrière (*Backward frame*) X_B et de la trame vers l'avant (*Forward frame*) X_F comme suit :

$$R(x,y) = \frac{X_F(x+dx_F, x+dy_F) - X_B(x+dx_B, x+dy_B)}{2} \tag{9.1}$$

où $X_F(x+dx_F, x+dy_F)$ et $X_B(x+dx_B, x+dy_B)$ représentent la trame vers l'avant compensée en mouvement et la trame vers l'arrière compensée en mouvement, respectivement, et (x,y) désigne les

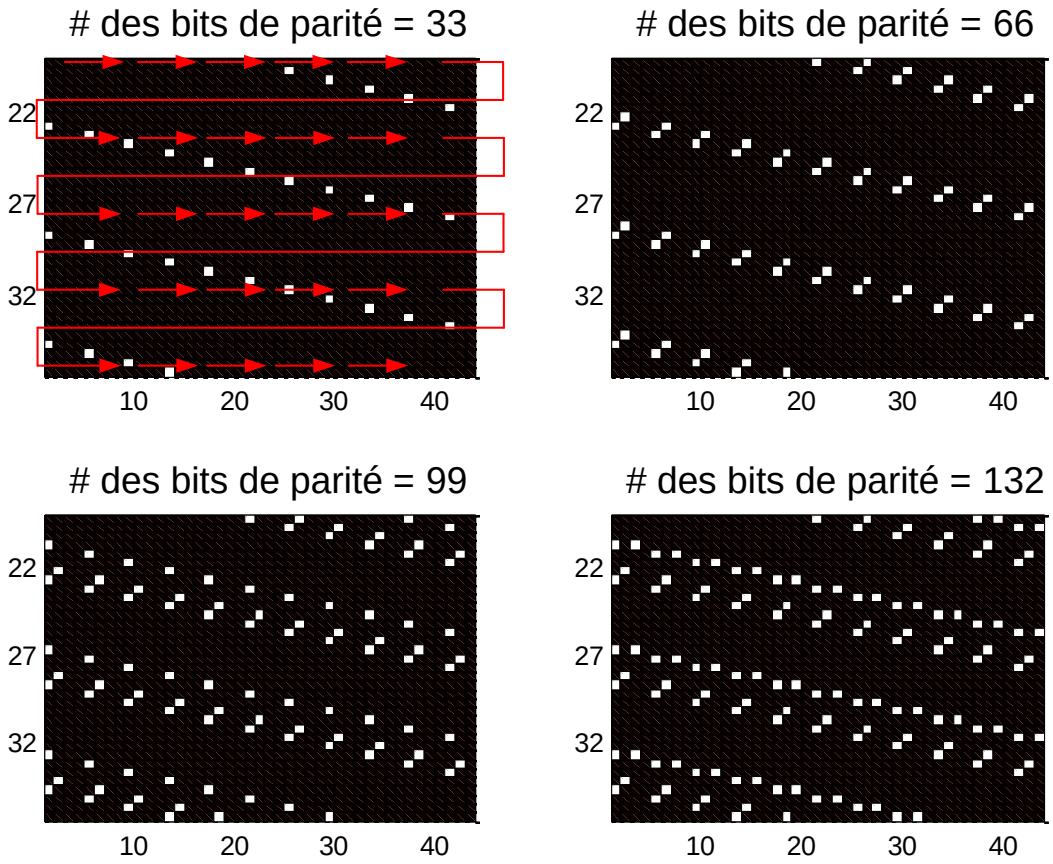

FIGURE 9.5: Poinçonnage périodique conduisant à une répartition uniforme des bits de parité à travers le plan de bits de la bande DCT .

coordonnées du pixel. (dx_B, dy_B) et (dx_F, dy_F) représentent les vecteurs de mouvement pour X_B et X_F respectivement. Cette méthode d'estimation de trame résiduelle est semblable à la méthode utilisée dans [60] pour la modélisation de la corrélation en ligne (voir l'équation 7.26). Ensuite on applique une transformation DCT 4×4 sur la trame résiduelle R, conduisant à une estimation T_b des 16 bandes DCT résiduelles où b indique la bande DCT.

La disposition des bits de parité et des bits systématiques doit être générée à la fois au niveau de l'encodeur et au niveau du décodeur de telle sorte que le turbo-décodeur soit capable de reconnaître l'emplacement de chaque bit reçu. Pour le mécanisme de poinçonnage périodique, par exemple, le décodeur peut déduire exactement l'emplacement de chacun des 33 bits de parité reçus à chaque demande via le canal de retour. En effet, une attribution d'un bit dans chacun des 33 périodes de taille 48 est effectuée selon un motif pseudo-aléatoire fixé à l'avance à l'encodeur et au décodeur. Cependant, en ce qui concerne le mécanisme de poinçonnage adaptatif, pour chaque bande DCT de chaque trame WZ, les bits de redondance (de parité et systématiques) sont envoyés selon une disposition particulière à travers le treillis de décodage turbo. Cet arrangement doit être généré de

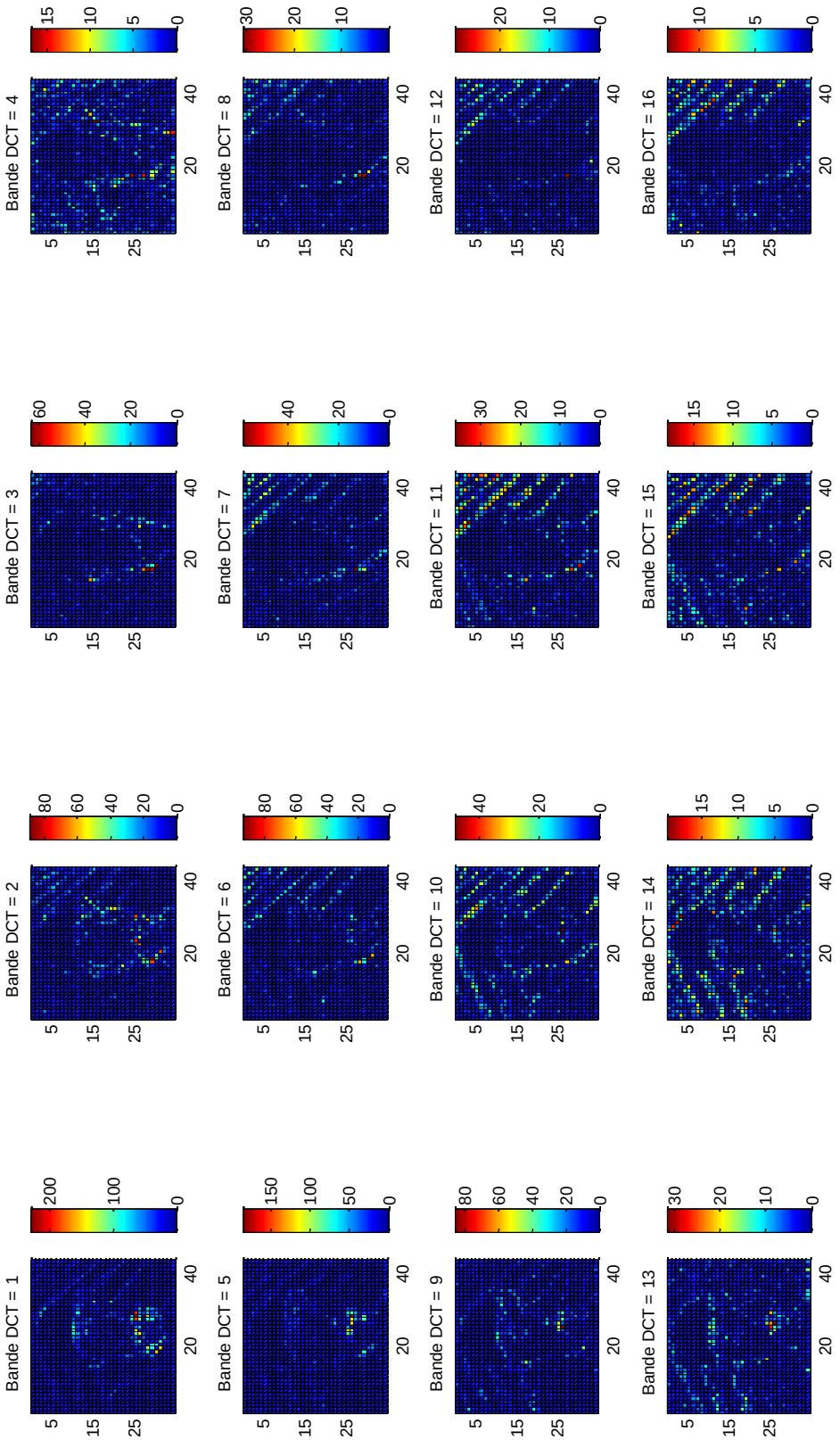

FIGURE 9.6: *Emplacement des erreurs de chaque bande DCT de l'information latérale. La disposition des erreurs varie d'une bande DCT à une autre.*

la même façon à l'émission et à la réception. En effet, il est calculé à partir de la bande résiduelle estimée T_b en respectant le concept de concentration des bits de redondance dans les régions les plus bruitées. La méthode d'allocation des bits proposée est appliquée à la fois au niveau de l'encodeur et du décodeur, selon les étapes suivantes :

- **Subdivision de chaque bande DCT en régions** : La bande DCT, de taille 36×44, est décomposée en 16 régions, chacune ayant une taille de 9×11 comme le montre la figure 9.7 affichant la bande DCT 1 de la figure 9.6 à part.
- **Classification des régions** : Le décodeur calcule ensuite la moyenne des différences absolues (*MAD* : mean of absolute difference) pour chaque région et il les classifie en 4 groupes (I, II, III et IV) où chaque groupe est composé de 4 régions. Le groupe I est composé des 4 régions ayant le *MAD* le plus élevé tandis que le groupe IV est composé des 4 régions ayant le plus petit *MAD*.
- **Information véhiculées du décodeur vers l'encodeur** : Après la classification, le décodeur informe, par l'intermédiaire du canal de retour, l'encodeur des 4 régions qui composent chaque groupe. Cela nécessite uniquement 12 bits envoyés sur le canal de retour pour chaque bande. En fait, le décodeur informe sur la composition des 3 régions (I, II, III) et l'encodeur déduit que les 4 régions restantes constituent le groupe IV. Par exemple, pour la première bande DCT présentée à la figure 9.7, la composition des groupes est la suivante :

$$\begin{array}{lcl} \text{Group I} & = & \{\, \text{Reg}_{10} \quad \text{Reg}_{11} \quad \text{Reg}_{14} \quad \text{Reg}_{15} \} \\ \text{Group II} & = & \{\, \text{Reg}_{4} \quad \text{Reg}_{6} \quad \text{Reg}_{7} \quad \text{Reg}_{8} \} \\ \text{Group III} & = & \{\, \text{Reg}_{1} \quad \text{Reg}_{3} \quad \text{Reg}_{5} \quad \text{Reg}_{16} \} \\ \text{Group IV} & = & \{\, \text{Reg}_{2} \quad \text{Reg}_{9} \quad \text{Reg}_{12} \quad \text{Reg}_{13} \} \end{array} \tag{9.2}$$

  avec Reg_k désigne la région numéro k.
- **Génération de la disposition des bits** : La composition de chaque groupe de régions est désormais connue à la fois au décodeur et à l'encodeur. En fait, la composition est générée au niveau du décodeur et renvoyé à l'encodeur. L'algorithme de répartition des bits de redondance est alors lancé à la fois du côté de l'émission et du côté de la réception. Cet algorithme prend comme paramètre d'entrée la composition des quatre groupes et permet de préciser conjointement l'endroit (à l'émission et à la réception) où les bits doivent être assignés.

L'algorithme de répartition des bits spécifie dans quel ordre et à quel emplacement les bits de parité et les bits systématiques doivent être envoyés. Rappelons que, lorsque le décodeur turbo ne parvient pas à converger, il demande à l'encodeur via le canal de retour, un fragment de 33 bits supplémentaires. Le principe de l'algorithme proposé est de diriger plus de bits pour les régions du groupe I et moins de bits pour les régions du groupe II, et ainsi de suite. La technique proposée de répartition des bits permet d'allouer itérativement les bits groupe par groupe selon un ordre ajusté par des essais empiriques et de simulations :

1. 4 bits (2 bits de parité pour chacun des encodeurs RSC 1 et RSC 2) sont dirigés vers chacune des 4 régions du groupe I.

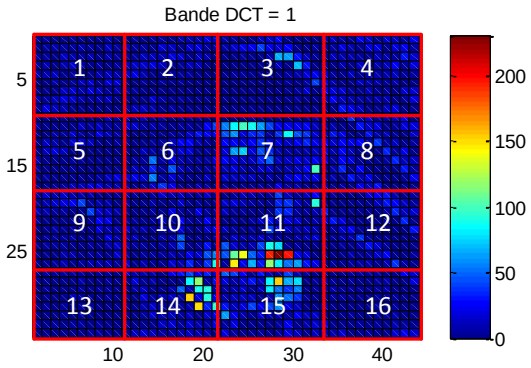

FIGURE 9.7: *Subdivision de la composante DC en régions et classification des régions en groupes donné par l'équation 9.2.*

2. 2 bits (1 bits de parité pour chacun des encodeurs RSC 1 et RSC 2) sont dirigés vers chacune des 4 régions du groupe II.
3. 1 bit (1 bit de parité pour RSC 1 et 1 bit de parité pour RSC 2 alternativement) est dirigé vers chacune des 4 régions du groupe III.
4. Aucun bits de parité n'est envoyé pour les régions du groupe IV.
5. Chaque fois que le nombre cumulatif des bits envoyés dépasse 1300, 1500 ou 2000, 50 bits systématiques sont uniformément dispersés dans les régions du groupe I.
6. Si le nombre total des bits envoyés est inférieur à 2640 bits, aller à l'étape 1.
7. Fin de l'algorithme de répartition des bits.

Pour une comparaison équitable avec le schéma de poinçonnage périodique, un fragment de 33 bits est envoyé, à chaque demande de bits supplémentaires via le canal de retour, selon l'ordre décrit par les étapes mentionnées ci-dessus.

9.3.3 Simulations et discussion

Dans cette sous-section, le mécanisme de poinçonnage proposé permettant une répartition adaptative des bits de parité et des bits systématiques est comparé au mécanisme de poinçonnage conventionnel dispersant de façon uniforme les bits de parité. Deux séquences vidéo QCIF à 15 trames par seconde sont considérées pour les tests de simulation : `Foreman`, `Soccer`. Ces séquences sont téléchargées à partir du site de Discover [5]. Toutes les 149 trames des séquences sont considérées.

La figure 9.8 affiche la bande DCT 1, T_1, de la trame résiduelle estimée $R(x,y)$ calculée à partir de l'équation (9.1) ainsi que la distribution des bits de parité/systématique selon les techniques de poinçonnage adaptative et périodique pour le sixième plan de bits de la composante DC de la première trame WZ de la séquence vidéo `Foreman`. La composition des quatre groupes de régions est donnée à l'équation (9.2). Pour le mécanisme de poinçonnage adaptatif, où les bits de parité et systématiques

sont dirigés vers les régions les plus corrompues, le nombre total de bits nécessaires pour la convergence du processus de décodage turbo est seulement de 1386 bits alors que le nombre de bits s'élève à 1815 pour le poinçonnage périodique. On voit, en effet, que les régions du groupe I (10, 11, 14 et 15) sont submergées par les bits (parité/ systématiques) alors qu'aucun bit n'est envoyé vers les régions du groupe IV (2, 9, 12 et 13). La barre de couleur latérale dans cette figure indique le nombre de bits (parité/systématiques) alloués dans le treillis du code turbo :

- 0 bit : ni les bits de parité ni le bit systématique ne sont alloués à cette position du treillis.
- 1 bit : un bit de parité est généré à partir de l'encodeur RSC 1 ou RSC 2 est affecté à cette position du treillis.
- 2 bits : les deux bits de parité des deux encodeurs RSC 1 et 2 sont attribués à cette position du treillis.
- 3 bits : les deux bits de parité et le bit systématique sont alloués à cette position du treillis.

Lors des simulations, quelques cas particuliers se sont révélés pour lesquels le poinçonnage adaptatif peut conduire à des performances beaucoup plus faibles (cas catastrophique) que le poinçonnage périodique. Cela amène à rectifier légèrement l'algorithme proposé lorsque de tels cas catastrophiques sont détectés. On s'explique à travers l'exemple de la figure 9.9 se rapportant au premier plan de bits de la neuvième bande de la troisième WZ frame de la séquence `soccer`. La composition des groupes pour la neuvième composante DCT T_9 de la trame résiduelle estimée est la suivante :

$$\begin{array}{lcl} \text{Group I} & = & \{\,\text{Reg}_5 \quad \text{Reg}_6 \quad \text{Reg}_7 \quad \text{Reg}_{10}\} \\ \text{Group II} & = & \{\,\text{Reg}_8 \quad \text{Reg}_9 \quad \text{Reg}_{13} \quad \text{Reg}_{16}\} \\ \text{Group III} & = & \{\,\text{Reg}_{11} \quad \text{Reg}_{12} \quad \text{Reg}_{14} \quad \text{Reg}_{15}\} \\ \text{Group IV} & = & \{\,\text{Reg}_1 \quad \text{Reg}_2 \quad \text{Reg}_3 \quad \text{Reg}_4\} \end{array} \tag{9.3}$$

On remarque que les régions du groupe IV se trouvent dans la même ligne. Étant donné que ce groupe ne reçoit pas de bits selon le schéma de poinçonnage adaptatif durant le processus de décodage, un quart du treillis reste sans aucun bit reçu. En effet $1584/4 = 394$ branches successives sur le treillis seront livrées à elles-mêmes sans bits de parité ni bits systématiques. Même si la partie restante du treillis est inondée par les bits, les erreurs dans la section sans bits du treillis ne peuvent être facilement corrigées. Ainsi, le schéma de poinçonnage périodique exige seulement 594 bits pour la convergence turbo alors que le schéma adaptatif proposé nécessite 2343 bits. Pour remédier à cette situation, l'algorithme doit détecter si les 4 régions du groupe IV se trouvent sur la même ligne. Si c'est le cas, le groupe IV doit être limité à seulement 3 régions. On propose dans ce cas une autre composition des groupes :

$$\begin{array}{lcl} \text{Group I} & = & \{\,\text{Reg}_5 \quad \text{Reg}_6\} \\ \text{Group II} & = & \{\} \\ \text{Group III} & = & \{\,\text{Reg}_4 \quad \text{Reg}_7 \quad \cdots \quad \text{Reg}_{16}\} \\ \text{Group IV} & = & \{\,\text{Reg}_1 \quad \text{Reg}_2 \quad \text{Reg}_3\} \end{array} \tag{9.4}$$

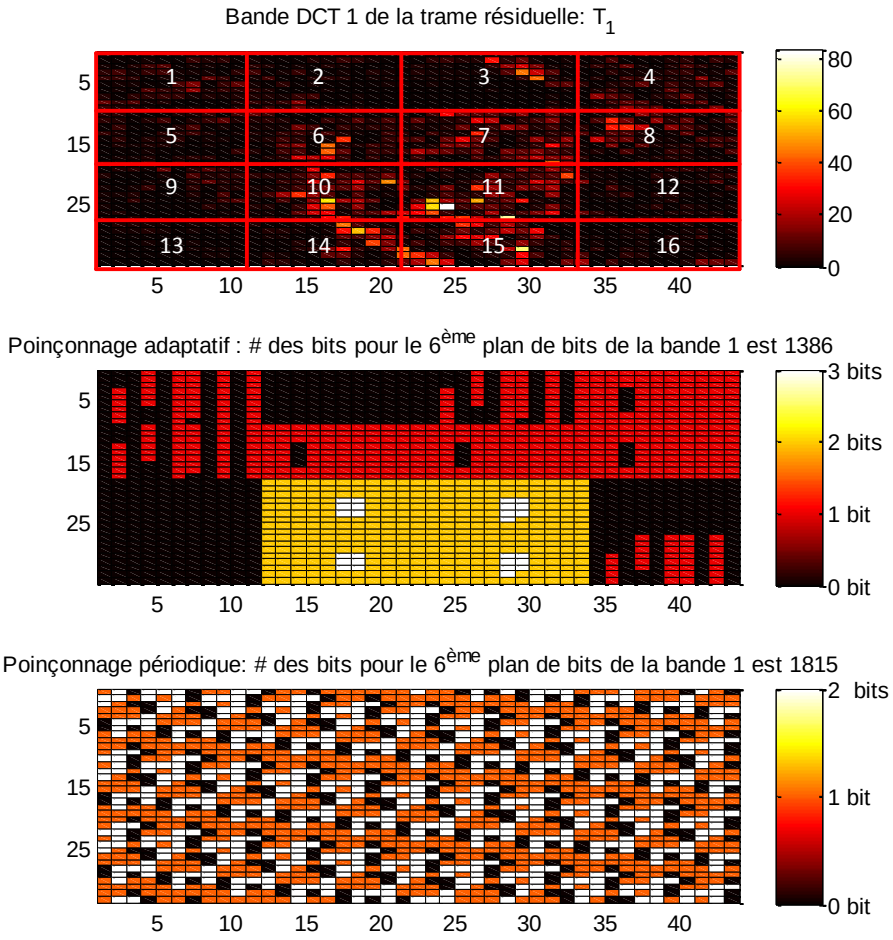

FIGURE 9.8: Distribution des bits de parité et des bits systématique pour les mécanismes poinçonnage adaptatif et périodique : 6ème plan de bits de la composante DC de la première trame WZ de `Foreman`, $GOP = 2$ et matrice de quantification à haut débit $Qi = 8$.

Cette simple modification est capable de briser la séquence de 394 branches consécutives ne recevant aucun bit. Ainsi, la distribution des bits devient tel que montré à la figure 9.10. Le nombre de bits requis est désormais 528 au lieu de 2343. Dans la suite de ce chapitre, la technique de poinçonnage adaptatif prend en compte ces cas particuliers.

Les performances en terme de débit-distorsion (trame WZ + trames clés) sont présentés à la figure 9.11 pour un groupe d'images $GOP = 2$, 4 et 8. Les trames clés sont encodées avec la norme vidéo H.264/AVC intra et après reconstitution, elles sont utilisées pour générer l'information latérale. L'algorithme de poinçonnage proposé est comparé au codec DVC Discover avec codage turbo [5] qui utilise le mécanisme de poinçonnage périodique. Les performances du codec standards H.264/AVC sont également présentées. Pour un $GOP = 2$, le codec DVC avec la technique de poinçonnage proposé conduit à une réduction du débit WZ pour le même PSNR de 15% en comparaison avec le codec de Discover pour la séquence `Foreman` et 18% pour la séquence `Soccer` . Cette dernière séquence contient des mouvements rapides et complexes de telle sorte que l'interpolation génère une informa-

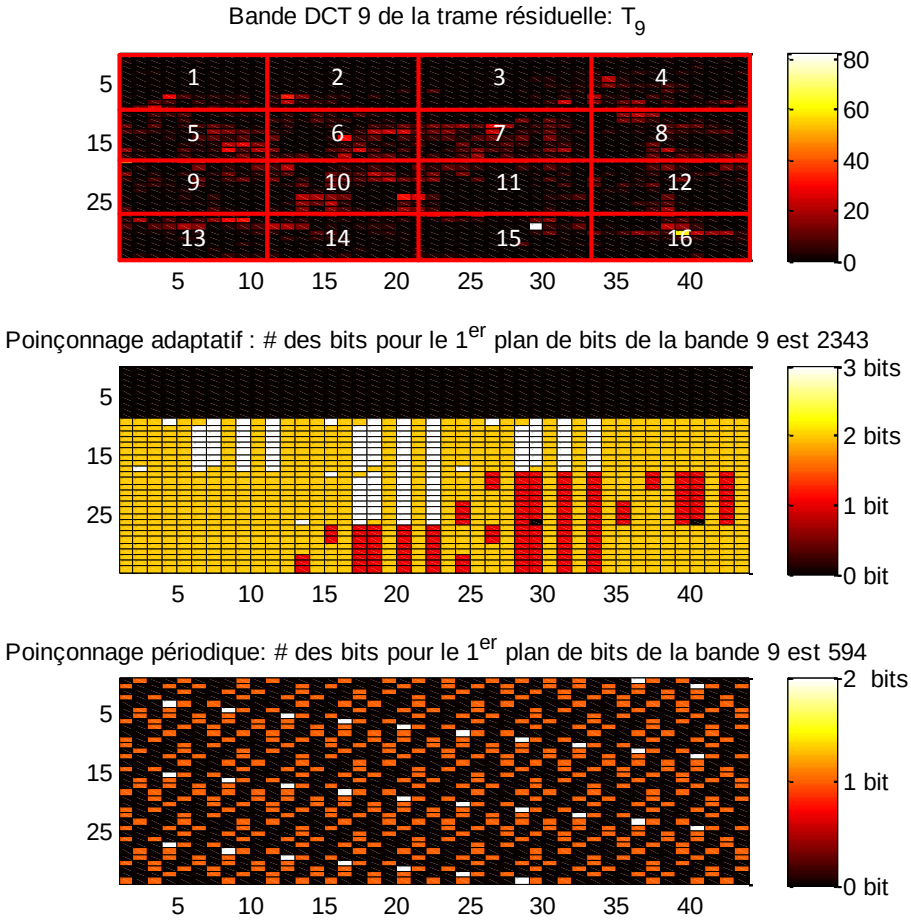

FIGURE 9.9: Distribution des bits de parité et des bits systématiques pour les mécanismes de poinçonnage adaptatif et périodique pour le cas particulier où toutes les régions du groupe IV se situent sur la même ligne : Premier plan de bits de la bande DCT 9 de la troisième trame WZ de la séquence vidéo `Soccer` et matrice de quantification $Qi = 7$.

tion latérale très bruitée. L'utilisation du poinçonnage adaptatif permet la concentration des bits de parité et même les bits systématiques dans les régions corrompues : ces régions correspondent essentiellement aux endroits mouvement. Pour un $GOP = 4$ et 8, une amélioration plus remarquable est observée vu que l'information latérale est de plus en plus bruitée lorsque le GOP augmente. Une amélioration de 20% du débit WZ est observée pour séquence vidéo `Soccer` avec un $GOP = 8$. Ceci revient à une amélioration des performances de R-D allant jusqu'à 1.8 dB.

9.4 Mécanisme hybride de contrôle du débit pour la minimisation des délais de décodage

Le paradigme de codage vidéo distribué consiste essentiellement à la redistribution de la complexité de calcul de l'encodeur vers le décodeur. Toutefois, en raison des nombreuses demandes de retour

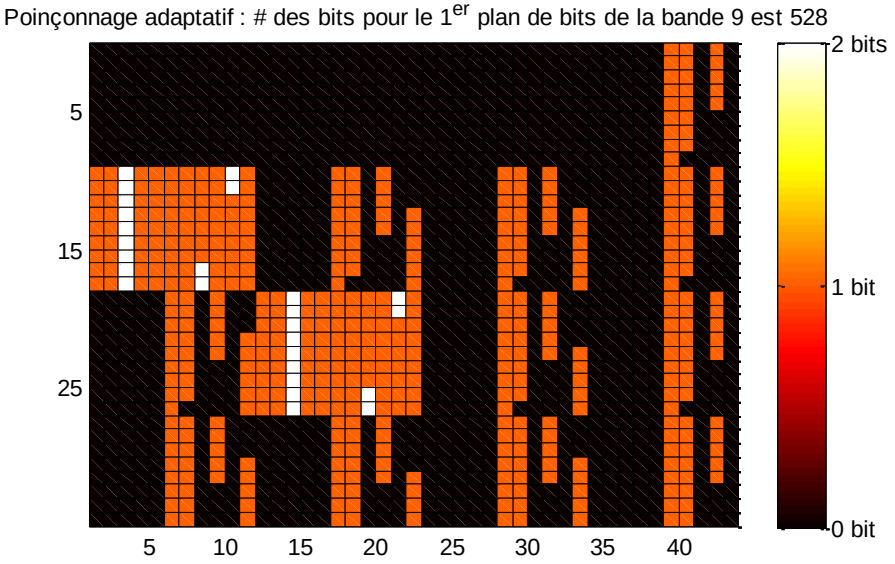

FIGURE 9.10: Distribution des bits de parité et des bits systématiques pour les mécanismes de poinçonnage adaptatif avec la modification nécessaire pour gérer les situations où toutes les régions du groupe IV sont sur la même ligne : premier plan de bits de la bande DCT 9 de la troisième trame WZ de la séquence vidéo `Soccer` et matrice de quantification $Qi = 7$.

et des cycles à répétition de décodage, de longues latences, inacceptables pour une implémentation pratique, sont engendrées au niveau du décodeur. Une approche pour résoudre le problème de latence consiste à estimer un nombre initial de fragments de bits de parité (INC : *Initial number of chunks*). Ces fragments sont ensuite envoyés d'un coup au décodeur, réduisant ainsi le nombre des boucles de décodage. Le principal défi consiste à estimer aussi précisément que possible le nombre final de fragments de bits de parité (FNC : *Final number of chunks*) nécessaire pour le processus de décodage turbo, sans sous-estimation ni surestimation. On propose dans cette section deux méthodes d'estimation du FNC basées sur la corrélation temporelle entre les trames Wyner-Ziv (WZ) successives ainsi que sur la corrélation entre les différents plans de bits.

9.4.1 Contrôle du débit dans les architectures DVC

Avant de proposer les nouvelles techniques d'estimation du FNC on présente les 3 principales techniques de contrôle du débit se trouvant dans la littérature. Ces techniques serviront à évaluer les performances des estimateurs proposés.

Contrôle du débit par le décodeur via le canal de retour

Dans les premières architectures DVC [22, 25], l'encodeur envoie graduellement les fragments de bits de parité à la demande du décodeur par le biais d'un canal de retour. Cette technique est connue sous l'appelation de "contrôle du débit par le décodeur" (DRC : *Decoder Rate Control*). Le canal de retour reliant le décodeur à l'encodeur permet ainsi d'ajuster le débit en fonction de la qualité de l'infor-

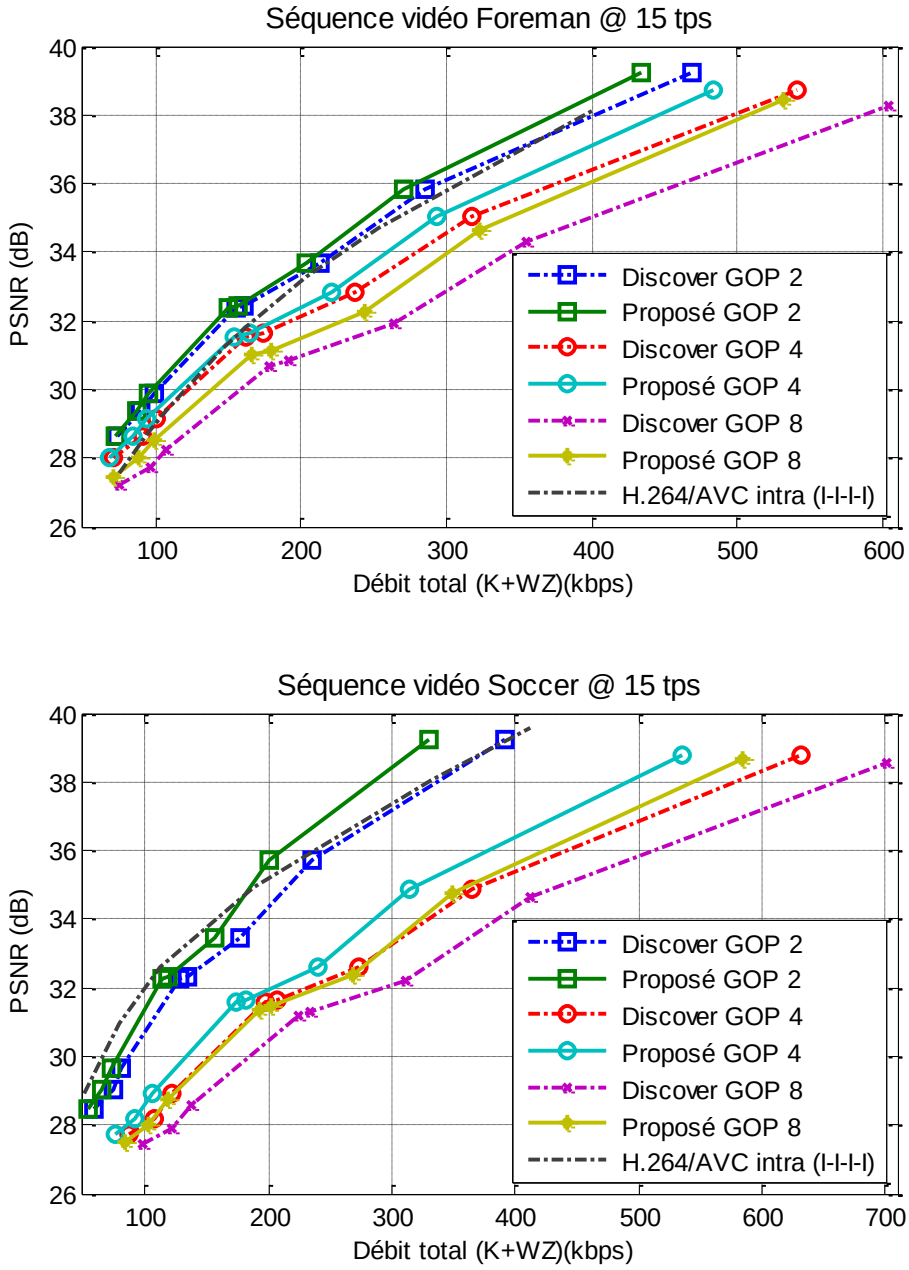

FIGURE 9.11: Comparaison des performances de débit-distorsion entre l'architecture proposée avec poinçonnage adaptatif et l'architecture Discover utilisant le poinçonnage périodique.

mation latérale. Cette méthode permet d'assurer l'envoi du débit minimal requis pour la convergence du décodeur et d'améliorer significativement des performances R-D. Cependant la complexité de décodage inhérente est inacceptable pour la plupart des applications pratiques. Comme cette technique assure l'envoi du nombre minimal requis de fragments de bits, FNC^{DRC}, l'objectif des techniques de contrôle de débit proposées est d'estimer ce nombre et d'envoyer les fragments de bits d'un coup. Le nombre estimé est envoyé initialement au décodeur (INC : *Initial Number of chunks*). Si l'INC dépasse le FNC^{DRC}, il s'agit de surestimation entrainant une augmentation du débit. Si l'INC est inférieur au FNC^{DRC}, il s'agit d'une sous-estimation entrainant des demandes de retour pour des fragments de bits supplémentaire ce qui résulte en des délais de décodage.

Contrôle du débit par l'encodeur

La suppression du canal de retour [84, 85] est une solution qui a été adoptée permettant de réduire drastiquement les délais de décodage mais ce, au prix d'une potentielle détérioration des performances. D'un côté, si le nombre de bits estimé à l'encodeur dépasse le minimum requis, le débit augmente. D'un autre côté, si le nombre de bits est sous-estimé, le décodeur ne parviendra pas à converger ce qui conduit à des artéfacts visuels sévères dans les trames reconstruites.

Contrôle hybride du débit par collaboration entre l'encodeur et le décodeur

Entre ces deux techniques de contrôle de débit, une technique hybride (compromis) peut être adoptée lorsque l'encodeur et le décodeur coopèrent pour estimer le taux minimal en utilisant le canal de retour. Un nombre initial de bits de parité R_{min} est envoyé au tout début. Si le décodeur turbo ne parvient pas à converger, un message est envoyé via le canal de retour, demandant à l'encodeur d'envoyer un nombre supplémentaire de bits de parité. Ce cycle est réitéré jusqu'à ce que le décodeur turbo converge. Des délais de décodage importants sont engendrés en cas de sous-estimation du débit minimal R_{min} et une augmentation du débit est engendrée en cas de surestimation. On détaille ci-dessous deux techniques hybrides de contrôle du débit se trouvant dans la littérature

1. **Contrôle hybride du débit basé sur le théorème de Slepian-Wolf** La technique hybride dans [65] vise à évaluer le nombre minimal des bits de parité R_{min} pour chaque plan de bits de chaque bande DCT. Le décodeur estime le paramètre du modèle Laplacien de corrélation de bruit α qu'il renvoie à l'encodeur. Connaissant les données originales et le paramètre α, l'encodeur estime dans un premier temps la probabilité de croisement p_{co} et ensuite le taux minimum de R_{min} selon le théorème de Slepian-Wolf. Cette technique est adoptée par le codec Discover et elle a été détaillée à la section 7.6.1.
2. **Contrôle hybride du débit avec faible complexité** : La technique précédente engendre une certaine complexité supplémentaire à l'encodeur lors de l'estimation du débit minimal. Dans [86], une technique hybride à faible complexité est proposée. Pour chaque plan de bits j, de la bande i, le nombre initial de fragments de bits de parité (INC) est estimé en utilisant le nombre final de fragments de bits de parité (FNC) envoyé pour les mêmes plans de bits dans les 3 trames

WZ précédentes :

$$\begin{aligned} INC\,(i,j) = \ & \text{floor}\,[(1-k)\times \\ & \text{median}\,(FNC_{-1}\,(i,j)\,,\,FNC_{-2}\,(i,j)\,,\,FNC_{-3}\,(i,j))] \end{aligned} \tag{9.5}$$

où k est un facteur d'échelle tel que $k = 0.1$ pour les cinq premières bandes DCT ($i = 1,\ldots,5$) et $k = 0.05$ pour les bandes restantes. Le terme $(1-k)$ empêche la saturation. L'opérateur "*f*loor" désigne la partie entière et l'opérateur "*m*edian" désigne la valeur médiane.

9.4.2 Algorithme basé sur la corrélation temporelle pour l'estimation du nombre initial de fragments (INC) (TC : *Temporal correlation*)

Cet algorithme exploite la stationnarité temporelle du FNC (*Final number of chunks*) pour effectuer une estimation en deux étapes du débit minimal. Plus précisément, considérons l'estimation de l'INC (*Initial number of chunks*) pour le sixième plan de bits de la première bande DCT de la trame WZ numéro $t = 36$ comme le montre la figure 9.12. Cette figure montre l'évolution temporelle du FNC^{DRC} (*Final number of chunks*) calculé par la technique de contrôle du débit au décodeur. La première étape consiste au calcul de l'INC en utilisant les trois valeurs précédentes FNC :

$$\begin{aligned} INC_{t=36}^{(\text{étape 1})}\,(\ \text{band} = 1,\ \text{bp} = 6) = & \ a \times FNC_{35}\,(1,6) \\ & + a^2 \times FNC_{34}\,(1,6) + a^3 \times FNC_{33}\,(1,6) \end{aligned} \tag{9.6}$$

Ici a est un facteur d'échelle tel que $a + a^2 + a^3 = 1 \Rightarrow a = 0.54$ s'il y a sous-estimation à la trame précédente ($t = 35$) et tel que $a + a^2 + a^3 = 0.8 \Rightarrow a = 0.47$ si il y a surestimation pour éviter la saturation. Dans le cas de surestimation, le décodeur turbo converge dès sa première exécution en utilisant le nombre initial de blocs de bits de parité sans avoir à envoyer une requête de retour pour des bits supplémentaires. Ici, deux cas possibles sont à mentionner. Le premier cas est effectivement une surestimation qui se produit lorsque le nombre initial de blocs (INC) estimé dépasse le nombre nécessaire pour la convergence du décodeur. Ce nombre étant celui retrouvé avec le mécanisme de contrôle par le décodeur (FNC^{DRC} : voir la section 9.4.1). Ainsi on a :

$$INC_{35}\,(1,6) > FNC_{35}^{DRC}(1,6) \tag{9.7}$$

Le second cas se produit lorsque l'estimateur parvient à déterminer exactement le nombre nécessaire pour la convergence du décodeur :

$$INC_{35}\,(1,6) = FNC_{35}^{DRC}(1,6) \tag{9.8}$$

Il n'est pas possible de faire la distinction entre ces deux situations car, dans les deux cas, le décodeur turbo converge à la première exécution. Dans la suite de ce chapitre, on considère ces deux cas comme étant une surestimation.

L'estimation à la première étape calcule une moyenne pondérée entre les valeurs FNCs précédentes. Cependant, lorsqu'il y a un pic ou un creux, cette estimation n'est pas assez proche de la valeur réelle.

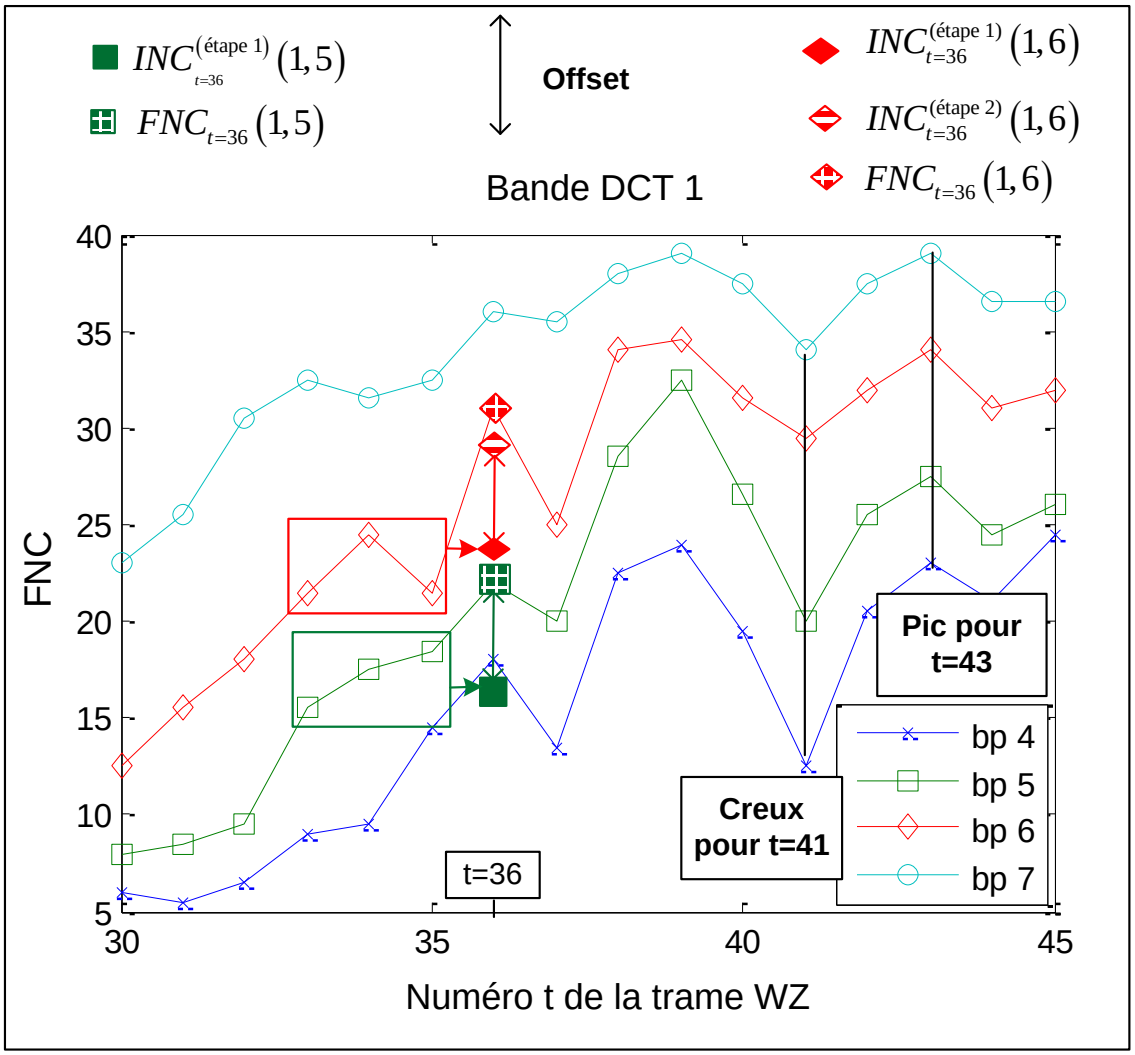

FIGURE 9.12: *Variation temporelle du FNC avec des pics et des creux observés pour les differents plans de bits. Le décalage (offset) entre la première estimation pour le plan de bit $bp = 5$ et le FNC peut être utilisé pour ajuster l'estimation pour le plan de bit $bp = 6$.*

Ceci (un pic ou un creux) peut être détecté en observant le plan de bits précédent. Par exemple, on remarque à partir du plan de bits $pb = 5$, qu'il y a un pic. Un décalage (*offset*) peut être calculé entre la valeur estimée à la première étape et la valeur réelle. Ce décalage devrait être retrouvé pour le plan de bits suivant $bp = 6$. Ainsi la première estimation peut être ajustée comme suit :

$$INC_{t=36}^{(\text{étape } 2)}(1,6) = INC_{t=36}^{(\text{étape } 1)}(1,6) + \text{offset}(bp = 5) \tag{9.9}$$

9.4.3 Algorithme d'estimation basé sur la corrélation entre les plans de bits (BP : *Bit Plane*)

On remarque à partir de la figure 9.13 que le décalage entre les FNCs de deux plans de bits successifs est presque le même aux instants t et $t+1$:

$$\begin{aligned} FNC_t(i,j) - FNC_t(i,j-1) \approx \\ FNC_{t+1}(i,j) - FNC_{t+1}(i,j-1) \end{aligned} \tag{9.10}$$

Cette observation est utilisée pour calculer l'INC pour un plan de bits pb basé sur le FNC du plan de bits précédent $pb-1$. Par exemple, le calcul de l'INC pour la deuxième bande DCT du sixième plan de bits pour la trame WZ $t = 53$, est la suivante :

$$\begin{aligned} INC_{53}(2,6) = FNC_{53}(2,5) + \\ [FNC_{52}(2,6) - FNC_{52}(2,5)] \end{aligned} \tag{9.11}$$

L'équation (9.11) est appliquée lorsqu'il n'y a pas de surestimation ni en ce qui concerne le $FNC_{52}(2,6)$ ni le $FNC_{53}(2,5)$. La surestimation de ces deux plans de bits peut être détectée au cours de leur décodage respectif. En effet, si $FNC_{52}(2,6) > INC_{52}(2,6)$, alors il n'y a pas de surestimation et, dans ce cas, $FNC_{52}(2,6) = FNC_{52}^{DRC}(2,6)$. Rappelons que $FNC_{52}^{DRC}(2,6)$ est le nombre de fragments de bit de parité cible déterminé avec la technique de contrôle du débit par le décodeur (DRC) via le canal de retour. S'il y a une surestimation dans l'un de ces deux plans de bits, alors l'expression (9.11) ne peut plus être appliquée, sinon il y aura un risque d'accumulation de surestimations menant à la *saturation*. Dans ce cas, (i.e. surestimation au niveau du sixième plan de bits de la trame WZ numéro $t = 52$ ou surestimation au niveau du cinquième plan de bits de la trame numéro $t = 53$, exclusivement. Il s'agit d'un ou exclusif "*xor*"), l'equation (9.11) devient :

$$\begin{aligned} INC_{53}(2,6) = FNC_{53}(2,5) + \\ a \times [FNC_{52}(2,6) - FNC_{52}(2,5)] \end{aligned} \tag{9.12}$$

où a est un facteur, fixé empiriquement à la valeur 0.8. Quand il y a une surestimation dans ces deux plans de bits à la fois (i.e. surestimation au niveau du sixième plan de bits de la trame WZ numéro $t = 52$ et surestimation au niveau du cinquième plan de bits de la trame numéro $t = 53$), l'éventualité de surestimation est plus probable dans le plan de bits courant. Ainsi il faut faire preuve de plus de précaution lors de l'estimation, en modifiant le facteur d'échelle a par a^2 et l'équation (9.12) devient :

$$\begin{aligned} INC_{53}(2,6) = FNC_{53}(2,5) + \\ a^2 \times [FNC_{52}(2,6) - FNC_{52}(2,5)] \end{aligned} \tag{9.13}$$

Ainsi, l'algorithme d'estimation du nombre initial de fragments de bits de parité (INC) basé sur la corrélation entre les plans de bits se résume comme suit :

$$
\begin{aligned}
&\textbf{Cas 1 :}\ FNC_{52}(2,6) > INC_{52}(2,6) \\
&\qquad \text{et } FNC_{53}(2,5) > INC_{53}(2,5) \\
&\text{alors} \\
&INC_{53}(2,6) = FNC_{53}(2,5) + \\
&\qquad\qquad [FNC_{52}(2,6) - FNC_{52}(2,5)]
\end{aligned}
$$

$$
\begin{aligned}
&\textbf{Cas 2 :} FNC_{52}(2,6) = INC_{52}(2,6) \\
&\qquad \text{xor } FNC_{53}(2,5) = INC_{53}(2,5) \\
&\text{alors} \\
&INC_{53}(2,6) = FNC_{53}(2,5) + \\
&\qquad\qquad a \times [FNC_{52}(2,6) - FNC_{52}(2,5)]
\end{aligned}
\tag{9.14}
$$

$$
\begin{aligned}
&\textbf{Cas 3 :} FNC_{52}(2,6) = INC_{52}(2,6) \\
&\qquad \text{et } FNC_{53}(2,5) = INC_{53}(2,5) \\
&\text{alors} \\
&INC_{53}(2,6) = FNC_{53}(2,5) + \\
&\qquad\qquad a^2 \times [FNC_{52}(2,6) - FNC_{52}(2,5)]
\end{aligned}
$$

On remarque que la surestimation est signalée lorsque le turbo-décodeur converge directement après avoir reçu le nombre initial de fragments (INC) sans aucune demande de bits supplémentaires via le canal de retour. Ainsi, le nombre final et le nombre initial de fragments sont égaux :

$$FNC_t(\mathrm{b},\mathrm{bp}) = INC_t(\mathrm{b},\mathrm{bp})$$

.

9.4.4 Implémentation des techniques proposées pour l'estimation du débit minimal

Dans cette section, les estimateurs proposés (*EST* : *TC* et *BP*) sont comparés avec l'estimateur de Kubasov *et al.* [65] ainsi qu'avec l'estimateur de Areia *et al.* [86]. Trois séquences vidéo QCIF à 15 images par seconde sont considérées pour les tests de simulation : `Foreman`, `Soccer` et `Coastguard`. Ces séquences sont téléchargées depuis le site de Discover [5]. Toutes les 149 trames des séquences sont considérées, correspondant à 74 trames WZ. La taille de trame est de 144×176 = 25344 pixels, menant à des plans de bits de longueur de 25344/16 = 1584 bits. La longueur de période de poinçonnage est de 48 qui se traduit par une taille de fragment de $(1584/48) \times 2 = 33 \times 2 = 66$ bits de parité envoyés à chaque requête de retour. Ceci correspond à 33 bits de parité pour chacun des deux encodeurs RSC. Après estimation du nombre initial de fragments (INC), l'encodeur envoie INC $\times 66$ bits de parité à la fois. Des retransmissions de fragments supplémentaires s'en suivront en cas de besoin.

Critères de comparaison entre les estimateurs

Pour comparer entre les performances des estimateurs, deux points sont à considérer :

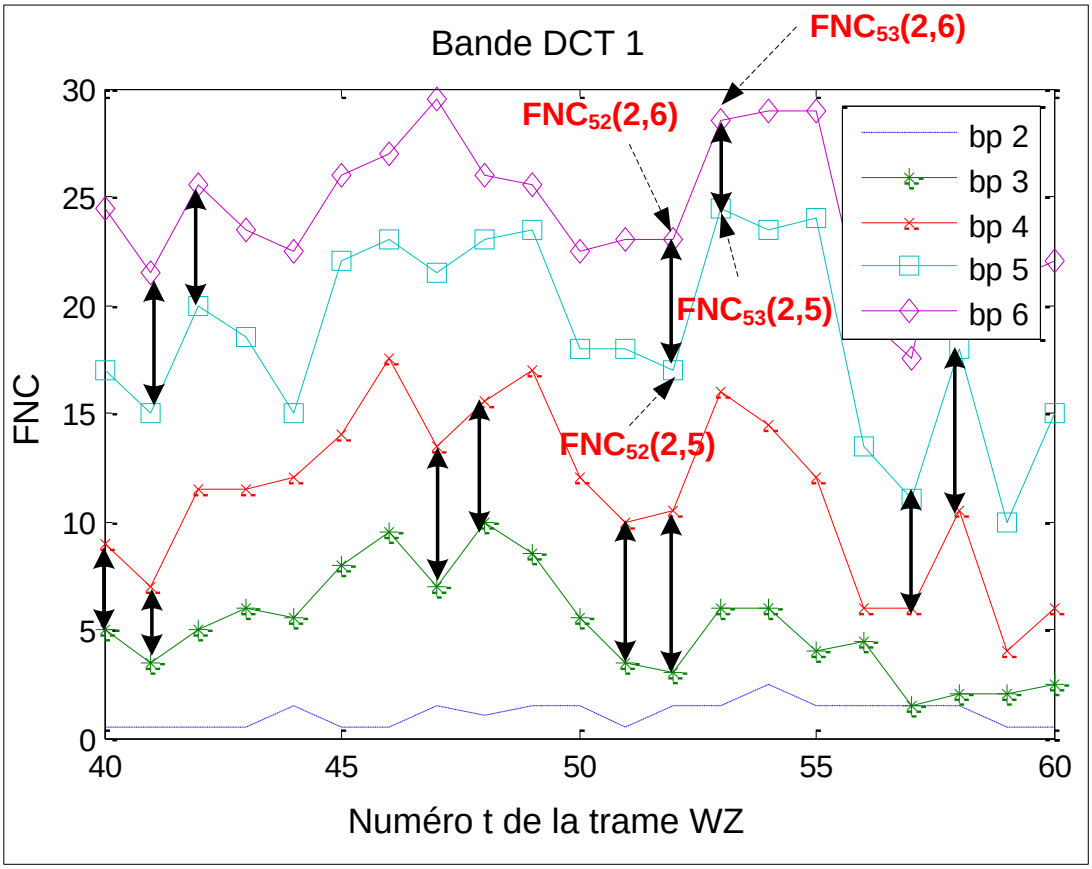

FIGURE 9.13: *Décalage du FNC entre deux plans de bits (bp) successifs .*

1. **Surestimation** : elle engendre une augmentation du débit. La moyenne, sur les N trames des WZ, du nombre de fragments envoyés en excès est donnée par :

$$\text{Excess} = \frac{\sum_{n=1}^{N} \max\left(\left[INC^{EST}(n) - FNC^{DRC}(n)\right], 0\right)}{N} \tag{9.15}$$

2. **Sous-estimation** : lorsque l'INC est en dessous de la FNC^{DRC}, le décodeur demande progressivement des fragments supplémentaires de bits de parité. Pour chaque demande, le décodage turbo est lancé à nouveau, provoquant ainsi des délais au décodeur. Le nombre moyen de demandes, sur les N WZ trames, est donné par :

$$\text{Request} = \frac{\sum_{n=1}^{N} \max\left(\left[FNC^{DRC}(n) - INC^{EST}(n)\right], 0\right)}{N} \tag{9.16}$$

Pour évaluer la précision de l'estimateur dans son ensemble, en tenant compte de la surestimation et de la sous-estimation, on définit un autre critère qui indique l'erreur d'estimation. Il s'agit de la moyenne, sur N trames WZ, de la différence absolue entre l'INC et le FNC^{DRC} obtenu avec contrôle

du débit au décodeur :

$$\text{Difference} = \frac{\sum_{n=1}^{N} \left| FNC^{DRC}(n) - INC^{EST}(n) \right|}{N} \tag{9.17}$$

Dans les simulations effectuées, la longueur des séquences vidéo de test est de 149 trames. Pour un $GOP = 2$, le nombre des trames WZ est alors $N = 74$.

9.4.5 Comparaison entre les algorithmes d'estimation du débit minimal

Avant de comparer les performances des estimateurs dans son ensemble, on montre à la figure 9.14 l'évolution temporelle du nombre de fragments INC estimé avec l'algorithme TC ainsi que le nombre final de fragments déterminé par le contrôle du débit au décodeur FNC^{DRC} pour certains plans de bits des 3 séquences vidéo. FNC^{DRC} indique le nombre cible de fragments qui doit être estimé.

À partir de la figure 9.14, on constate les points suivants :

- **Estimateur d'Areia *et al.* [86]** : Cet estimateur est basé sur le simple calcul de la médiane pondérée du nombre de fragments final (FNC) des 3 trames précédentes. Comme on peut l'observer pour la séquence `Coastguard`, cet estimateur n'est pas en mesure de suivre les variations brusques du nombre de fragments requis. Ce phénomène rappelle le *débordement de pente* lors de la modulation delta avec un pas de quantification faible.
- **Les estimateurs proposés TC et BP :** ces deux estimateurs peuvent fournir des estimations plus précises que l'algorithme de Areia *et. al* [86] et ils sont en mesure de suivre avec plus de précision les variations rapides du nombre de fragments cible (FNC^{DRC}). En fait ces techniques permettent de régler le processus d'estimation en prenant en considération les plans de bits précédemment décodés (de la même trame WZ ou des trames WZ précédentes).
- **Estimateur basé sur la correlation temporelle (TC)** : cet estimateur bénéficie de la structure temporelle du nombre de fragments final (FNC) (voir l'équation (9.6)) ainsi que de la quasi-stationnarité des décalages (*offsets*) entre deux plans de bits successifs bp et $bp+1$ (voir l'équation (9.9)).
- **Estimateur basé sur la corrélation entre les plans de bits (BP) :** cet estimateur exploite la quasi-stationnarité temporelle (c'est-à-dire entre la trame t et la trame $t+1$) de l'écart entre 2 plans de bits successifs.
- Ces deux estimateurs basés sur des observations empiriques permettent d'avoir une estimation précise vu qu'ils s'ajustent au fur et à mesure en utilisant le paramètre d'oubli a (voir les équations (9.6) et (9.14)). En fait, d'une part, lorsque ce paramètre décroît, les occurrences de surestimation diminuent résultant en des meilleures performances de débit-distorsion. D'un autre part, lorsque ce paramètre augmente, les sous-estimations se produisent moins fréquemment résultant en un décodage rapide.

À la figure 9.15, une comparaison des différents estimateurs est présentée pour les critères suivant : nombre de fragments demandés en excès, nombre de retour par trame, différence absolue moyenne et les performances R-D pour les séquences vidéo `Foreman`, `Soccer` et `Coastguard`. Huit matrices de quantification (Q_i) sont considérées. Le critère de la différence absolue stipule que les solutions proposées sont plus précises et donne des estimations plus proche au nombre de fragments nécessaires pour la convergence de décodage. Les solutions d'estimation proposées sont plus précises et les courbes débit-distorsion affichent une légère hausse du débit causée par la surestimation.

Les performances des estimateurs sont ensuite récapitulées dans les tableaux 9.2 et 9.3. Le premier tableau donne le pourcentage de la réduction de la complexité du décodeur par rapport à la solution DRC. Le deuxième tableau présente le pourcentage de l'augmentation du débit par rapport à la solution DRC. Les estimateurs proposés peuvent réduire significativement les latences du décodeur (une réduction moyenne de 87,5% pour la solution TC et de 88.29% pour la solution BP) sans aucun impact sévère sur les performances de débit-distorsion (seulement une augmentation du débit de 8,93 % et 9.37%, respectivement).

Tableau 9.2: *Pourcentage de réduction de la complexité par rapport à la technique de contrôle du débit au décodeur (DRC).*

	Kubasov	Areia	BP	TC
`Foreman`	53.22%	84.59%	88.02%	87.07%
`Soccer`	55.68%	86.74%	90.21%	89.00%
`Coastguard`	64.28%	84.75%	86.65%	86.43%
Moyenne	57.72%	85.36%	88.29%	87.50%

Tableau 9.3: *Pourcentage de l'augmentation du débit causée par la surestimation par rapport à la technique de contrôle du débit au décodeur (DRC).*

	Kubasov	Areia	BP	TC
`Foreman`	0.65%	17.09%	10.88%	10.46%
`Soccer`	0.53%	8.51%	7.92%	7.31%
`Coastguard`	1.04%	15.24%	9.33%	9.03%
Moyenne	0.74%	13.61%	9.37%	8.93%

9.5 Conclusion

Dans ce chapitre de nouvelles techniques touchant à l'aspect performance débit-distorsion et à l'aspect pratique de l'architecture DVC sont présentées. Le mécanisme de poinçonnage adaptatif permet une utilisation efficace des bits générés par l'encodeur turbo. Les bits de parité sont concentrés dans les endroits du treillis de décodage susceptibles d'être bruités. Si ces endroits s'avèrent encore plus bruités et que l'information latérale ne parvient pas à fournir des rapports de vraisemblance canal fiables, l'utilisation des bits systématiques se veut pertinente. La technique de poinçonnage présentée résulte en une économie de 20% du débit WZ.

Pour ce qui a trait à l'aspect pratique, rappelons que l'objectif du paradigme DVC est de transférer la complexité de l'encodeur vers le décodeur. Nonobstant, pour une implémentation réelle, il faut viser une complexité raisonnable du côté du décodeur. L'estimation du débit nécessaire au décodeur permet d'éviter plusieurs lancements du long processus de décodage turbo avec ses itérations inhérentes. Cette estimation doit être précise pour assurer la rapidité du décodage avec un moindre impact sur les performances R-D.

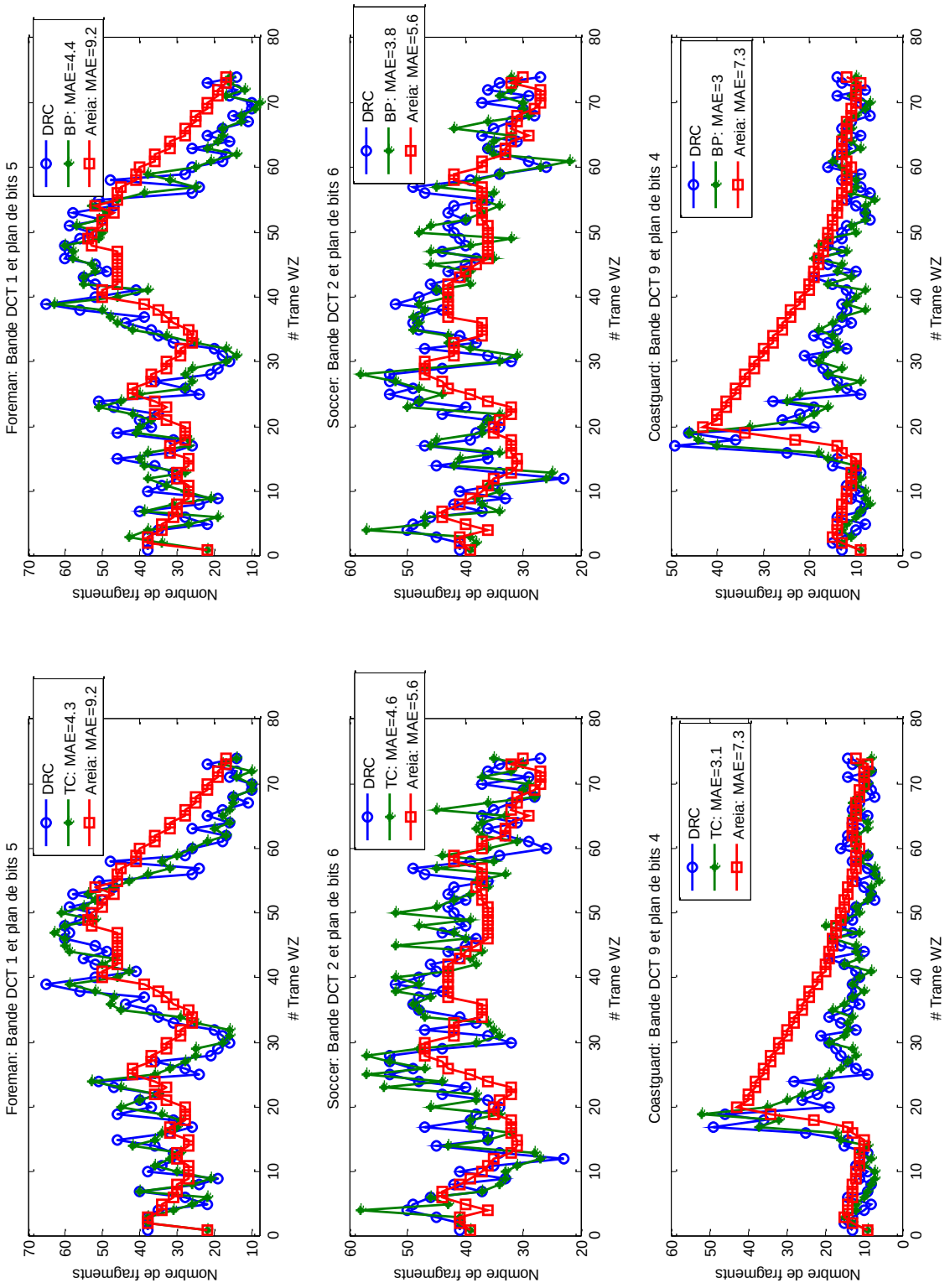

FIGURE 9.14: *Évolution temporelle de l'estimation du nombre de fragments INC en comparaison avec le nombre de fragments cible DRC pour certains plans de bits pour un indice de quantification $Qi = 8$. L'évaluation des différents estimateurs se fait à partir de la moyenne des erreur absolues (MAE).*

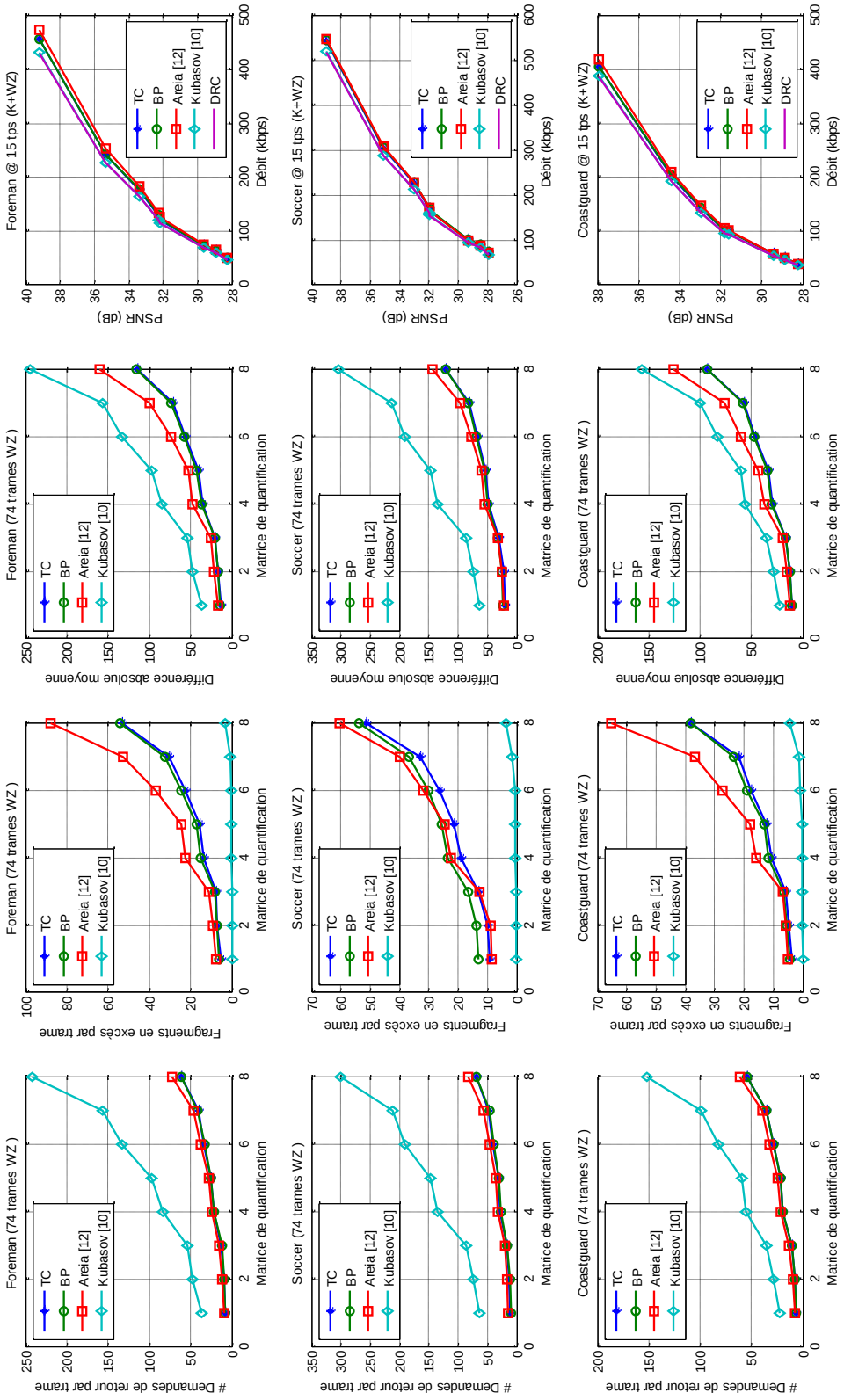

FIGURE 9.15: *Comparaison entre les différentes techniques d'estimation de R_{min}.*

Chapitre 10

Conclusion générale

10.1 Résumé de la thèse

Dans cette thèse de doctorat, on a présenté un rappel des principaux résultats de la théorie d'information qui ont donné lieu à la théorie du codage source distribué. Avec le développement de la nouvelle génération de la téléphonie mobile et des caméras vidéo sans fil, cette propriété du codage distribué retrouve un terrain d'application vaste. En effet, la corrélation présente entre les différentes trames envoyées successivement au récepteur, permet de visualiser le système de communication comme un codeur de Wyner-Ziv dans la mesure où les trames reçues sont une version bruitée, disponible au récepteur, des trames avoisinantes. Elles peuvent être alors traitées pour générer une information latérale au décodeur. L'exploitation de la corrélation temporelle se fait à travers la génération de l'information latérale au niveau du décodeur et acquitte l'encodeur de toute charge d'estimation de mouvement. Le modèle de Wyner-Ziv s'applique alors bien à la compression vidéo lorsque l'on souhaite déplacer la complexité de traitement de l'encodeur vers le décodeur. L'appellation "distribué" provient du fait que l'information latérale n'est disponible qu'au décodeur.

Les meilleures techniques de codage vidéo distribué se basent sur les codes LDPC et les codes turbo. Ces derniers constituent à la base des codes correcteurs d'erreurs dont les performances tendent vers les limites théoriques. Le contexte d'utilisation de ces codes dans le codage vidéo distribué constitue certaines particularités par rapport au contexte habituel. Les trames clés (trames intra) dont dispose le décodeur, présentent une forte similitude avec les trames Wyner-Ziv (trames inter) à transmettre, de telle sorte qu'on peut considérer qu'elles ont été envoyées dans un canal qui rajoute un certain bruit déformant la trame originale. La différence entre les trames consécutives d'une séquence vidéo caractérise les statistiques du canal virtuel. Cette différence tend à suivre une distribution Laplacienne. Ainsi, on peut réduire le débit alloué à la transmission des bits systématiques : la détermination de la vraisemblance canal se basera sur les pixels de la trame interpolée ou sur les coefficients de la bande DCT selon qu'on soit dans le domaine pixel ou transformé. Outre le fait qu'il n'y ait pas de transmission effective de la source à travers un canal, il existe une autre particularité dans le système implémenté. Dans les applications habituelles, le décodeur turbo évalue la vraisemblance canal en

se basant sur les bits systématiques bruités transmis. Le canal virtuel considéré ici ne génère pas à sa sortie des bits systématiques bruités mais plutôt, les valeurs des pixels de la trame interpolée ou les coefficients de la bande DCT. Même si le domaine de codage vidéo distribué constitue un domaine de recherche récent, depuis la dernière décennie les travaux concernant ce sujet se sont multipliés. Les diverses composantes de ce système ont été étudiées intensivement et les performances de l'état de l'art sont de plus en plus avancées. Cependant, l'architecture reste tout de même assez figée autour de celle présentée par les chercheurs de l'Université de Stanford en 2002 [22]. Pour pouvoir présenter des améliorations en codage vidéo distribué, nous avons repris les différents modules de l'architecture DVC dans le but d'implémenter les principaux systèmes DVC de la littérature. On a procédé à l'implémentation du système DVC dans le domaine pixel avec turbo code à 2 RSCs de taux 4/5 et avec turbo code à 2 RSCs de taux 1/2 et extractions des plans de bits. On a aussi mis en place le système DVC dans le domaine transformé avec différents quantificateurs pour les coefficients AC.

Suite à l'implémentation du système DVC dans le domaine pixel avec un turbo code à 2 RSCs 4/5, nous avons constaté que le poinçonnage peut s'effectuer de façon plus efficace en dirigeant les bits de parité vers les zones les plus bruitées de la trame. Cette technique de poinçonnage adaptatif dans le domaine pixel a été présentée en conférence [83] et constitue la base pour d'autres investigations ultérieures pour l'amélioration du codec W-Z dans le domaine des transformées. Des techniques d'interpolation bidirectionnelle sont ensuite étudiées et implémentées pour l'amélioration de l'architecture DVC de base. L'interpolation étant un élément clé dans le système DVC ayant un impact direct sur les performances R-D globales, que ce soit en la réduction du débit qu'en l'augmentation du PSNR. En effet une interpolation plus précise conduit à une information latérale plus corrélée avec l'information originale. Ainsi moins de bits de parité sont requis pour la correction de l'information latérale. Et vu que cette dernière est également exploitée dans le bloc de reconstruction, une meilleure qualité, en termes de PSNR, de la séquence vidéo reconstruite est assurée. Dans le même élan d'amélioration des performances R-D, s'inscrit aussi le passage du domaine des pixels vers le domaine des transformées. L'application de la transformation, en particulier la DCT 4×4, permet une certaine exploitation de la corrélation spatiale au niveau des trames WZ. L'objectif de l'implémentation de ces diverses techniques consiste à reproduire le codec état de l'art de Discover. Cette étape est importante pour bien comprendre, déterminer les points à améliorer et disposer d'un codec de base servant à des fins de comparaison avec les techniques qu'on propose. Étant donné que l'architecture DVC présenté par le projet Discover était considéré par la communauté comme une architecture de l'état de l'art, les diverses techniques que ce projet avait proposé sont à leur tour implémentées. Des performances R-D similaires sont obtenues permettant la validation des travaux de simulation de ré-implémentation.

10.2 Contributions

Dans ce projet de doctorat, ralliant les techniques de compression vidéo avec les techniques de codage canal, on a placé une attention particulière à chacune des composantes de l'architecture globale. Ces efforts ont permis de disposer du codec état de l'art Discover dans l'optique d'incorporer certaines

idées avec la possibilité de mener une étude comparative. On récapitule ci-dessous les 3 grandes directions empruntées à cet effet :

1. **Exploitation de la corrélation spatiale** : L'architecture DVC conventionnelle ne présente pas un mécanisme efficace pour exploiter la corrélation spatiale au niveau de la trame. Ceci explique en partie l'écart de performances entre le standard H264/AVC et le DVC. Pour cette raison, on a proposé une architecture DVC progressive permettant d'exploiter la corrélation intra-trame au niveau du décodeur. Ainsi le paradigme DVC avec un encodeur léger est maintenu. Une trame est découpée en groupes de blocs spatialement corrélés et les groupes sont envoyés et décodés successivement. Chaque groupe de blocs décodé et reconstruit sert à l'amélioration de la qualité de l'information latérale pour le groupe suivant. Cette direction de recherche a conduit à la publication d'un article de journal [87] et d'un article de conférence [88]. Un second article de journal [89] a été soumis dans ce même contexte.
2. **Mécanisme de poinçonnage adaptatif** : Le dispositif de poinçonnage depuis l'apparition du paradigme DVC est conforme au même fonctionnement qui élimine les bits systématiques et répartit uniformément les bits de parité le long du plan de bits. On a proposé en premier lieu un mécanisme permettant une variation adaptative de la matrice de poinçonnage. Selon la progression de l'envoi des bits, la matrice est selectionnée, permettant de mélanger les bits de parité et les bits systématiques. Ces derniers sont parfois d'une grande utilité, lorsque l'information latérale (SI) est fortement bruitée, pour amortir les cas d'expansion. La technique d'utilisation des bits systématique a été publiée dans un article de conférence [78]. Ce mécanisme de poinçonnage a été ensuite amélioré pour assurer une réparation efficace des bits de parité et des bits systématique. Ces bits sont dirigés vers les emplacements où l'information latérale est corrompue, correspondant généralement aux emplacement du mouvement. Une réduction considérable du débit WZ est obtenue car les bits sont envoyés là où il le faut au lieu que d'être disperser à l'aveuglette. Ce travail a été récapitulé dans un article et soumis à une conférence [79].
3. **Réduction des délais de décodage** : Étant donné que l'architecture de Discover est basée sur un canal de retour pour le contrôle du débit, le décodage WZ peut faire face à des délais importants. Ces délais sont dûs aux boucles de transmission de fragments de bits supplémentaires et de décodage turbo. Pour réduire ces délais et la complexité du décodeur, le nombre de fragments de bits de parité est estimé et envoyé d'un coup. Deux techniques pour une estimation précise ont été présentées permettant d'évaluer le nombre de fragments tout en amortissant les situations de surestimation et de sous-estimation. La complexité de décodage a été grandement réduite pour une faible augmentation du débit. Ces techniques ont fait l'objet d'un article de conférence [80] et d'un article de journal [81].

10.3 Suggestion de travaux futurs

En dépit des efforts multiples consentis pour la mise en place de systèmes DVC de plus en plus compétitifs et performants, des étapes importantes restent à franchir avant de voir un dispositif vi-

déo opérationnel se basant sur ce nouveau paradigme. On énumère ici quelques points dont l'étude permettra au DVC d'assurer sa place dans l'industrie vis-à-vis des standards vidéo conventionnels :

1. **Amélioration du concept du codage vidéo distribué progressif** : L'objectif de l'architecture progressive est d'exploiter de la redondance spatiale au niveau du décodeur sans induire une complexité à l'encodage. Cette architecture se base sur la subdivision de la trame en groupes de blocs spatialement corrélés entrainant une longueur moindre du code turbo. Sachant que les performances du code décroît quand la longueur du plan de bits diminue, le concept progressif peut être ajusté afin de garder la même longueur du code. L'idée consiste à encoder plusieurs trames WZ en même temps. Plus spécifiquement considérons le schéma DVC progressif avec 4 groupes de blocs. On peut rassembler 4 trames WZ consécutives et procéder à l'encodage habituel. Quatre nouvelles trames seront générées qui consistent aux 4 premiers groupes des 4 trames WZ consécutifs, aux 4 seconds groupes, aux 4 troisièmes groupes et aux 4 quatrièmes groupes. De cette manière, il est possible de regagner la taille originaire du code avant subdivision de la trame. De plus, les nouvelles trames créées assurent une certaines dispersion des erreurs vu que chaque nouvelle trame est une mosaïque des 4 trames WZ. Cependant, cette idée nécessite plus d'espace de stockage et induit certains délais d'affichage des trames vu que chaque ensemble de 4 trames WZ est décodées en bloc.
2. **Raffinement progressif de l'information latérale** : Dans l'architecture DVC dans le domaine transformé, les bandes DCT sont décodées une par une, plan de bits par plan de bits. Après le décodage de chaque bandes et de chaque plan de bits même, le récepteur acquiert une nouvelle information dont il ne disposait pas lors de la génération de l'information latérale. Cette nouvelle information sur la trame originale peut être exploitée pour améliorer la qualité de l'information latérale pour les bandes DCT restantes et les plans de bits suivants. Après le décodage d'une bande DCT, deux possibilités sont à explorer pour le raffinement des vecteurs de mouvement. L'estimation de mouvement peut, d'un côté, être réalisée entre la bande DCT décodée et la bande DCT des trames voisines. D'un autre côté, l'estimation de mouvement peut se faire après avoir regagné le domaine des pixels par l'application de la DCT inverse sur les bandes DCT déjà décodées.
3. **Raffinement de l'information latérale par décodage des hautes fréquences en premier** : Lors d'un mouvement d'un objet dans une séquence vidéo, ses arêtes sont capables de tracer le déplacement effectué. Par ailleurs, les bandes DCT vers les hautes fréquences définissent mieux les arêtes et les frontières des objets dans une trame que les bandes DCT vers les basses fréquences. Il est ainsi pertinent de décoder et de reconstruire, en premier, les composantes dans les hautes fréquences et effectuer un raffinement des vecteurs de mouvement pour générer une meilleure information latérale pour les basses fréquences. De plus, il est à noter que les coefficients DCT dans les hautes fréquences nécessitent un débit moindre que les coefficients dans les basses fréquences vu que la plage dynamique est plus petite et le nombre de bits de quantification est plus faible.
4. **Élimination du canal de retour** : Dans certains scénarios de transmission vidéo où le canal

de retour est inexistant ou lorsque les délais de décodage ne peuvent être tolérés, le contrôle du débit doit se faire au niveau de l'encodeur. Il revient alors à l'encodeur de déterminer le nombre de bits de parité nécessaire pour le décodage. La question devient encore plus délicate car l'encodeur n'a pas accès à l'information latérale (SI) pour déduire le niveau de corrélation avec l'information originale. L'idée consiste à produire une version grossière de la SI au niveau de l'encodeur sans que cela ne requiert beaucoup de ressources. À partir de cette version, la corrélation et le débit sont estimés. Même si le cas de surestimation se traduit par une augmentation du débit, c'est le cas de sous-estimation qui est le plus critique. En effet, le plan de bits du coefficient DCT ne pourra pas être décodé correctement, entrainant des artéfacts dans certains blocs de pixels relatif à ce coefficient DCT lors de la reconstruction. Une technique de raffinement de la trame reconstituée doit être mise en place dans ce genre de situation.

5. **Parallélisme de la procédure du décodage** : Conjointement avec l'élimination du canal de retour, le parallélisme de la procédure du décodage peut encore réduire les délais et ainsi les latences de tout le système. Ceci permet potentiellement d'assurer de la transmission vidéo à temps réel avec le paradigme DVC.

Bibliographie

[1] I. E. G. Richardson, *H.264 and MPEG-4 Video Compression : Video Coding for Next Generation Multimedia*. WileyInterscience, 2003.

[2] J. Slepian et J. Wolf, "Noiseless coding of correlated information sources," *IEEE Trans. Inf. Theory 19 (4)*, pp. 471–480, juillet 1973.

[3] A. D. Wyner et J. Ziv, "The rate-distortion function for source coding with side information at the decoder," *Transactions on Information Theory, vol. IT-22, no. 1*, pp. 1–10, janvier 1976.

[4] L. Bahl, J. Cocke, F. Jelinek et J. Raviv, "Optimal Decoding of Linear Codes for minimizing symbol error rate," *Transactions on Information Theory, vol. 20, no. 2*, pp. 284–287, mai 1974.

[5] Discover, "The DISCOVER codec evaluation." Disponible en ligne : `http://www.discoverdvc.org/` (page consultée le 26 février 2011).

[6] Bjøntegaard, G. , "Calculation of average PSNR differences between RD-curves," *ITU-T Video Coding Experts Group (VCEG)*, Austin, TX, 2001.

[7] C.E. Shannon, "A Mathematical Theory of Communication," *Bell System Technical Journal, 27*, pp. 379–423 et 623–656, juillet et octobre, 1948.

[8] C. Brites, J. Ascenso et F. Pereira, "Improving transform domain Wyner-Ziv video coding," *IEEE ICASSP*, Toulouse, France, mai 2006.

[9] X. Artigas, J. Ascenso, M. Dalai, S. Klomp, D. Kubasov et M. Ouaret, "The DISCOVER codec : Architecture, techniques and evaluation," *in Proc. Picture Coding Symp.*, Lisbon, Portugal, novembre 2007.

[10] Z. Wang, A. C. Bovik, H. R. Sheikh et E. P. Simoncelli, "Image quality assessment : From error visibility to structural similarity," in *IEEE Transactions on Image Processing*, vol. 13, pp. 600–612, avril 2004.

[11] C. Brites, J. Ascenso et F. Pereira, "Improving transform domain Wyner-Ziv video coding," *IEEE ICASSP*, Toulouse, France, mai 2006.

[12] C. Berrou, A. Glavieux et P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding : Turbo-codes. 1," *Technical Program, Conference Record, IEEE International Conference, Vol. 2*, Geneva, 1993.

[13] J. Ascenso, C. Brites et F. Pereira, “Improving Frame Interpolation with Spatial Motion Smoothing for Pixel Domain Distributed Video Coding,” *EURASIP*, République Slovaque, juillet 2005.

[14] J.-Y. Chouinard, *Théorie et pratique des codes correcteurs : Notes de cours*. Québec : Université Laval, 2009.

[15] The Wolfram Functions Site, “Calculating the amount of information in a message.” Disponible en ligne : http ://www.colorado.edu/communication/meta-discourses/Theory/infotheory/sld007.htm (page consultée le 22 avril 2009).

[16] O. Aftab, “Information theory and the digital age,” *The Structure of Engineering Revolutions. Cambridge, Massachusetts Institute of Technology*, 2001.

[17] A. D. Wyner, “On source coding with side information,” *Transactions on Information Theory, vol. 21, no. 3*, pp. 294–300, mai 1975.

[18] Wikipedia, “YUV.” Disponible en ligne : http ://en.wikipedia.org/wiki/YUV (page consultée le 22 avril 2009).

[19] Comment ça marche, “Introduction à la vidéo numérique.” Disponible en ligne : http ://www.commentcamarche.net/contents/video/video.php3 (page consultée le 22 avril 2009).

[20] G. Huchet, “Nouvelles méthodes de codage vidéo distribué,” *Thèse de Doctorat, Université Laval*, 2009.

[21] Wikipedia, “Video compression.” Disponible en ligne : http ://en.wikipedia.org/wiki/Videocompression (page consultée le 22 avril 2009).

[22] A. Aaron, R. Zhang et B. Girod, “Wyner-Ziv coding of motion video,” *Asilomar Conference on Signals, Systems and Computers*, novembre 2002.

[23] R. Puri et K. Ramchandran, “PRISM : A new robust video coding architecture based on distributed compression principles,” *40th Allerton Conf. Communication, Control and Computing*, Allerton, USA, octobre 2002.

[24] R. Puri, A. Majumdar et K. Ramchandran, “PRISM : A Video Coding Paradigm With Motion Estimation at the Decoder,” *IEEE Transactions on Image Processing*, Allerton, USA, octobre 2002.

[25] B. Girod, A. Aaron, S. Rane et D. Rebollo Monedero, “Distributed video coding,” in *Proceedings IEEE, Special Issue on Advances in Video Coding and Delivery*, janvier 2005.

[26] A. Aaron, S. Rane, E. Setton et B. Girod, “Transform-domain Wyner-Ziv codec for video,” *Proc. Visual Communications and Image Processing, VCIP*, San Jose, USA, janvier 2004.

[27] S. Ye, M. Ouaret, F. Dufaux et T. Ebrahimi, “Improved side information generation with iterative decoding and frame interpolation for distributed video coding,” in *Proceedings of IEEE International Conference on Image Processing (ICIP 08)*, États-Unis, octobre 2008.

[28] J.L. Martínez, G. Fernandez-Escribano, H. Kalva et P. Cuenca, "Decoder side motion vector derivation for inter frame video coding," *IET Image Processing*, octobre 2009.

[29] S. Kamp, M. Evertz et M. Wien , "Motion Vector Refinement in a Wyner-Ziv to H.264 Transcoder for Mobile Telephony," *International conference on image processing-ICIP*, 2008.

[30] C. Brites et F. Pereira, "Encoder Rate Control for Transform Domain Wyner-Ziv Video Coding," *IEEE Int. Conf. Image Processing, ICIP-2007*, San Antonio, TX, USA, mai 2007.

[31] Y. Vatis, S. Klomp et J. Ostermann , "Enhanced reconstruction of the quantized transform coefficients for Wyner-Ziv coding," *Proceedings of Int. Conf. on Multimedia and Expo (ICME)*, Beijing, Chine, juillet 2007.

[32] D. Kubasov, J. Nayak et C. Guillemot , "Optimal Reconstruction in Wyner-Ziv Video Coding with Multiple Side Information," *Proceedings of Int. workshop on MMSP*, Grèce, octobre 2007.

[33] W.A.R.J. Weerakkody, J. Nayak et C. Guillemot, "Enhanced reconstruction algorithm for unidirectional distributed video coding," *IET Image Processing*, décembre 2009.

[34] W.A.R.J.Weerakkody, W.A.C. Fernando et A.B.B. Adikari, "Unidirectional Distributed Video Coding for Low Cost Video Encoding," *IEEE Transactions on Consumer Electronics*, vol. 53, mai 2007.

[35] G. Huchet, J.-Y. Chouinard, D. Wang et A. Vincent, "Adaptive source representation for distributed video coding," *IEEE International Conference on Image Processing (ICIP 2009)*, Caire, Égypte, novembre 2009.

[36] S. Sofke, F. Pereira et E. Müller, "Dynamic Quality Control for Transform Domain Wyner-Ziv Video Coding," *EURASIP Journal on Image and Video Processing*, janvier 2009.

[37] A. Aaron, S. Rane, R. Zhang et B. Girod, "Wyner-Ziv coding for video : Applications to compression and error resilience," *in Proc. IEEE Data Compression Conference*, Snowbird, UT, mars 2003.

[38] F. Pereira, L. Torres, C. Guillemot, T. Ebrahimi, R. Leonardi et S. Klomp, "Distributed Video Coding : Selecting the most promising application scenarios," *Signal Processing : Image Communication*, 2008.

[39] F. Pereira, C. Brites ,J. Ascenso et M. Tagliasacchi, "Wyner-Ziv Video Coding : A Review of the Early Architectures and Further Developments," *Proc IEEE International Conf. on Multimedia and Expo - ICME*, Hannover , Allemagne , juin 2008.

[40] I. Bajcsy et P. Mitran, "Coding for the Slepian-Wolf problem using turbo codes," *Globecom, San Antonio*, novembre 2001.

[41] 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.

[42] A. Avudainayagam, J. M. Shea et D. Wu, "A Hyper-Trellis based Turbo Decoder for Wyner-Ziv Video Coding," *Global Telecommunications Conference, 2005. GLOBECOM '05. IEEE*, 2005.

[43] A. Aaron et B. Girod, "Compression with side information using turbo codes," *Proc. Asilomar Conference on Signals and Systems*, 2002.

[44] S. Lin et D.J. Costello, Jr., *Error Control Coding : Fundamentals and Applications*. 2^{me} édition, Englewood Cliffs, NJ, USA : Prentice-Hall, 2004.

[45] W. E. Ryan, *Concatenated Codes and Iterative Decoding*. New York : Wiley and Sons : Encyclopedia of Telecommunications, 2003.

[46] K. Sayood, *Introduction to Data Compression*. 2^{me} édition, Californie, États Unis : Morgan Kaufmann, 2000.

[47] Wikipedia, "Test du χ^2." Disponible en ligne : http ://fr.wikipedia.org/wiki/Test-du-chi-carre (page consultée le 05 mai 2010).

[48] C. Brites, *Advances on distrubuted video coding*. PhD thesis, Universidade Técnica de Lisboa Instituto Superior Técnico, Décembre 2005.

[49] "YUV Video Sequences." Disponible en ligne : http://trace.eas.asu.edu/yuv/index.html (page consultée le 11 mai 2010).

[50] "YUV QCIF reference videos (lossless H.264 encoded)." Disponible en ligne : http://www.tkn.tu-berlin.de/research/evalvid/qcif.html (page consultée le 11 mai 2010).

[51] "CIPR QCIF Sequences." Disponible en ligne : http://www.cipr.rpi.edu/resource/sequences/qcif.html (page consultée le 11 mai 2010).

[52] "QCIF raw video." Disponible en ligne : http://www.madsgroup.org/ (page consultée le 11 mai 2010).

[53] S. C. for Image Systems Engineering, "Test Images and Videos." Disponible en ligne : http://scien.stanford.edu/pages/labsite/scien-test-images-videos.php (page consultée le 11 mai 2010).

[54] L. Alparone, M. Barni, F. Bartolini et V. Cappellini, "Adaptively Weighted Vector-Median Filters for Motion Fields Smoothing," *IEEE ICASSP*, Georgia , États-Unis, mai 1996.

[55] A. Zaccarin, *Notes de cours de traitement des images*. Québec : Université Laval, 2010.

[56] Optical Illusions And Pictures, "Barber Pole illusion." Disponible en ligne : http ://www.123opticalillusions.com/pages/barberpole.php (page consultée le 08 juin 2010).

[57] Wikipedia, "Least squares." Disponible en ligne : http ://en.wikipedia.org/wiki/Leastsquares (page consultée le 08 juin 2010).

[58] X. T. Media, "SD Content and Below." Disponible en ligne : http://media.xiph.org/video/derf/ (page consultée le 10 juin 2010).

[59] C. Brites, J. Ascenso et F. Pereira, “Studying Temporal Correlation Noise Modeling for Pixel Based Wyner-Ziv Video Coding,” *IEEE International Conference on Image Processing*, Atlanta, État-Unis, octobre 2006.

[60] C. Brites et F. Pereira, “ Correlation Noise Modeling for Efficient Pixel and Transform Domain Wyner Ziv Video Coding,” *IEEE transactions on circuits and systems for video technology*, vol. 53, 9 septembre 2008.

[61] J. Ascenso, C. Brites et F. Pereira, “Content adaptive Wyner-Ziv video coding driven by motion activity,” *IEEE International Conference on Image Processing*, Atlanta, États-Unis, octobre 2006.

[62] J. Lee et B. W. Dickinson, “Multiresolution video indexing for subband coded video databases,” *Proc. SPIE, Conference on Storage and Retrieval for Image and Video Databases*, San José, California, États-Unis, février 1994.

[63] A. Aaron, E. Setton et B. Girod, “Towards practical Wyner-Ziv coding of video,” *Proc. IEEE International Conference on Image Processing, ICIP-2003*, septembre 2003.

[64] S. Klomp, Y. Vatis et J. Ostermann, “Side Information Interpolation with Sub-pel Motion Compensation for Wyner-Ziv Decoder,” *International Conference on Signal Processing and Multimedia Applications (SIGMAP)*, 7-10 août 2006.

[65] D. Kubasov, K. Lajnef et C. Guillemot , “A hybrid encoder/decoder rate control for Wyner-Ziv video coding with a feedback channel,” *Proceedings of Int. workshop on Multimedia Signal Processing*, Grèce, octobre 2007.

[66] ISO/IEC 14496-10 :2003, “ Coding of Audio-Visual Objects - Part 10 : Advanced Video Coding, 1ère Édition,” *(also ITU-T : 2003, H.264)*, juillet 2003.

[67] D. Agrafiotis, P. Ferré et D. R. Bull, “Hybrid key/Wyner-Ziv frames with flexible macroblock ordering for improved low delay distributed video coding,” in *Visual Communications and Image Processing*, pp. 3097–3100, CA, USA, 30 janvier 2007.

[68] B. Wu, X. Ji, D. Zhao et W. Gao, “Spatial-Aided Low-Delay Wyner-Ziv Video Coding,” *EURASIP Journal on Image and Video Processing*, pp. 1–11, 2009.

[69] A. Aaron, S. Rane et B. Girod, “Wyner-Ziv video coding with hash-based motion compensation at the receiver,” in *Proceedings of IEEE International Conference on Image Processing (ICIP 04)*, vol. 5, pp. 3097–3100, Singapour, octobre 2004.

[70] M. Tagliasacchi, A. Trapanese, S. Tubaro et J. Ascenso, C. Brites et F. Pereira, “Exploiting spatial redundancy in pixel domain Wyner-Ziv video coding,” in *Proceedings of IEEE International Conference on Image Processing (ICIP 06)*, vol. 5, pp. 253–256, octobre 2006.

[71] M. Guo, Y. Lu, F. Wu, S. Li, W. Gao, “Distributed Video Coding with Spatial Correlation Exploited Only at the Decoder,” in *International Symposium on Circuits and Systems (ISCAS 2007)*, 41-44, pp. 3097–3100, 27-20 Mai 2007.

[72] J. Ascenso, C. Brites et F. Pereira, “Motion compensated refinement for low complexity pixel based distributed video coding,” in *Proceedings of IEEE Conference on Advanced Video and Signal Based Surveillance (AVSS 05)*, pp. 593–598, Italie, Septembre 2005.

[73] X. Artigas et L. Torres, “Iterative Generation of Motion-Compensated Side Information for Distributed Video Coding,” in *Proc. IEEE International Conference on Image Processing (ICIP)*, Italie, septembre 2005.

[74] M. B. Badem, W. A. C. Fernando, J. L. Martinez et P. Cuenca, “An iterative side information refinement technique for transform domain distributed video coding,” in *Proceedings of the 2009 IEEE international conference on Multimedia and Expo*, ICME’09, pp. 177–180, IEEE Press, New York, États-Unis, 2009.

[75] R. Martins, C. Brites, J. Ascenso et F. Pereira, “Refining Side Information for Improved Transform Domain Wyner-Ziv Video Coding,” *IEEE Transactions On Circuits And Systems For Video Technology*, vol. 19, pp. 1327–1341, Septembre 2009.

[76] F. Verbist, N. Deligiannis, S. Satti, A. Munteanua et P. Schelkens, “Iterative Wyner-Ziv decoding and successive side-information refinement in feedback channel-free hash-based distributed video coding,” *Proceedings - SPIE The International Society For Optical Engineering*, 2012.

[77] N. Deligiannis, F. Verbist, J. Slowack, R. Walle, Peter Schelkens et Adrian Munteanu, “Progressively Refined Wyner-Ziv Video Coding for Visual Sensors,” *ACM Transactions on Sensor Networks, from Special Issue on New Advancements in Distributed Smart Camera Networks*, vol. 10, Mai 2014.

[78] M. Haj Taieb, J.-Y. Chouinard et D. Wang, “Adaptive use of systematic bits in distributed video coding with multiple puncturing matrices,” *Visual Communications and Image Processing Conference*, 27-30 novembre , 2012, San Diego, USA.

[79] M. Haj Taieb, J.-Y. Chouinard et D.Wang , “Turbo Code using Adaptive Puncturing for Transform Domain Wyner-Ziv Video Coding,” in *13th Canadian Workshop on Information Theory*, 18-21 juin 2013.

[80] M. Haj Taieb, J.-Y. Chouinard et D. Wang, “Low complexity hybrid rate control schemes for distributed video coding,” *ICSPIE 2012 : International Conference on Signal Processing and Imaging Engineering 2012*, 24-26 octobre, 2012, San Francisco, USA.

[81] M. Haj Taieb, J.-Y. Chouinard et D.Wang, “Low Complexity Rate Estimators for Low Latency Wyner-Ziv Video Decoders,” in *Journal Engineering Letters*, vol. 21, pp. 1–9, Mars 2013.

[82] The DISCOVER codec evaluation. Comparing the RD Performance of LDPC and Turbo Codes, “http://www.img.lx.it.pt/~discover/rd_comparison_qcif_15_gop248_ldpc_tc.html,”

[83] M. Haj Taieb, J.-Y. Chouinard, D.Wang et K. Loukhaoukha, "Turbo code using adaptive puncturing for pixel domain distributed video coding," in *Proceedings of the 4th international conference on Image and signal processing*, ICISP'10, pp. 342–350, Trois-Rivières, Canada. 2010.

[84] Martinez, J.L. Fernandez-Escribano, G. Kalva, H. Weerakkody, W.A.R.J. Fernando, et W.A.C. Garrido, "Feedback free DVC architecture using machine learning," in *ICIP 2008 15ème IEEE International Conference on Image Processing*, pp. 1140 – 1143, Octobre 2008.

[85] 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 (WCN)*, 2008.

[86] J. D. Areia, J. Ascenso, C. Brites et F. Pereira, "Low Complexity Hybrid Rate Control for Lower Complexity Wyner-Ziv Video Decoding," in *16th European Signal Processing Conference (EUSIPCO)*, août 2008.

[87] M. Haj Taieb, J.-Y. Chouinard, D.Wang, K. Loukhaoukha , "Progressive coding and side information updating for distributed video coding," in *Journal of Information Hiding and Multimedia Signal Processing*, vol. 3, Janvier 2012.

[88] M. Haj Taieb, J.-Y. Chouinard, D. Wang, K. Loukhaoukha et G. Huchet, "Progressive Distributed Video Coding with Multiple Passes for Side Information Update," in *SETIT 2012 6th International Conference : Sciences of Electronic, Technologies of Information and Telecommunications*, 23-26 Mars 2012.

[89] M. Haj Taieb, J.-Y. Chouinard, D.Wang , "Spatial correlation based side information refinement for distributed video coding," in *EURASIP Journal on Advances in Signal Processing Special Issue on Advances in Distributed Source Coding*, Soumis en Avril 2013.