École Doctorale
d'Informatique,
Télécommunications
et Électronique de Paris

Thèse

Présentée pour obtenir le grade de docteur
de l'École Nationale Supérieure des Télécommunications

Spécialité : Signal et Image

Charles YAACOUB

Codage conjoint source-canal pour l'optimisation d'un système de codage distribué de sources vidéo transmises sur un lien sans fil

Soutenue le 03 juillet 2009 devant le Jury composé de

M. Claude Berrou	Président
Mme. Christine Guillemot	Rapporteurs
M. Jean-Marc Brossier	
M. Ghaïs El Zein	Examinateur
Mme. Joumana Farah	Directrices de thèse
Mme. Béatrice Pesquet-Popescu	

École Doctorale
d'Informatique,
Télécommunications
et Électronique de Paris

A THESIS

Submitted for the award of the degree of doctor of philosophy
of the National Higher School of Telecommunications

Department of Signal and Image Processing

Charles YAACOUB

Joint source-channel coding for the optimization of the distributed coding of video sources transmitted over wireless channels

July 03, 2009

Examining committee in charge:

Mr. Claude Berrou	Télécom Bretagne, France
Mrs. Christine Guillemot	Institut de Recherche en Informatique et Systèmes Aléatoires, France
Mr. Jean-Marc Brossier	Institut National Polytechnique de Grenoble, France
Mr. Ghaïs El Zein	Institut d'Electronique et de Télécommunications de Rennes, France
Mrs. Joumana Farah	Université Saint-Esprit de Kaslik, Lebanon
Mrs. Béatrice Pesquet-Popescu	Télécom ParisTech, France

Any piece of knowledge I acquire today has a value at this moment exactly proportioned to my skill to deal with it. Tomorrow, when I know more, I recall that piece of knowledge and use it better.

Mark van Doren

To my beloved Petra,

To my Family... with Love

Remerciements

Je voudrais d'abord remercier *Prof. Béatrice Pesquet-Popescu* pour avoir dirigé mes travaux de recherche et pour son hospitalité lors de mes séjours à Télécom ParisTech.

Je suis évidemment très reconnaissant à *Dr. Ing. Joumana Farah*, chef du département de génie informatique et télécommunications à l'Université Saint-Esprit de Kaslik, pour avoir encadré de près et de façon continue mes activités de recherche, pour sa confiance en moi, et pour la gentillesse et la patience qu'elle a manifestées à mon égard durant cette thèse.

Je tiens à exprimer ma profonde gratitude à ces deux personnes, pour leur aide et leur soutien pendant plus de trois ans, et pour leurs efforts à organiser la soutenance de thèse avec un jury d'experts.

Mes remerciements vont également aux rapporteurs, *Mme. Christine Guillemot* et *M. Jean-Marc Brossier* pour avoir accepté d'évaluer et de rapporter mon travail, et pour leur participation au Jury de soutenance. Je remercie de même *M. Claude Berrou* et *M. Ghaïs El Zein* pour m'avoir fait l'honneur de participer au Jury.

Je remercie aussi *M. François Marx* de France Telecom (Orange) pour ses remarques et ses conseils durant ma première année de thèse.

Je tiens très particulièrement à remercier *Dr. Thomas André*, *M. Thomas Maugey*, et *Mme. Teodora Petrisor*, pour leur hospitalité et leur soutien durant mes séjours au sein du groupe multimédia à Télécom ParisTech.

J'exprime aussi mes remerciements à la Faculté des Sciences et de Génie Informatique de l'Université Saint-Esprit de Kaslik, ainsi qu'au département de Traitement des Signaux et des Images à Télécom ParisTech, pour m'avoir offert un bon environnement de travail.

Je remercie aussi mes collègues et amis, *Ing. Roy Abi Zeid Daou*, *Ing. Nancy Rachkidy*, et *Ing. Tilda Akiki* pour leur bonté, leur encouragement et leur foi en moi.

Je voudrais également remercier *Ing. Pascal Damien* pour son soutien moral, mon frère *Ing. Elias Yaacoub* et ma fiancée *Ing. Petra Bilane* pour les discussions techniques et pour leur soutien infini, ainsi que mon ami *Ing. Joseph Rahmé* pour son aide et son hospitalité durant mes séjours en France. Je tiens surtout à remercier ma très chère fiancée *Petra* pour sa patience et sa compréhension, et pour m'avoir soutenu dans les moments les plus difficiles de ma thèse.

Finalement, je voudrais exprimer tout mon amour et ma reconnaissance à mes parents pour m'avoir aidé et encouragé durant mes longues années d'études, et pour m'avoir toujours soutenu moralement et financièrement.

Charles Yaacoub

Contents

Glossary

2D-GA	Two-directional genetic algorithm
ARA	Adaptive rate allocation
ARAQ	Adaptive rate allocation and quantization
ARAQ-D	Adaptive rate allocation and quantization with frame dropping
AVI	Average interpolation
AWGN	Additive white gaussian noise
BER	Bit error rate
BHD	Block histogram difference
bps	Bits per second
BSC	Binary symmetric channel
BVD	Block variance difference
CDF	Cumulative distribution function
CRC	Cyclic redundancy check
CSI	Channel state information
DCT	Discrete cosine transform
DH	Difference of histograms
DSC	Distributed source coding
DVB	Digital video broadcasting
DVC	Distributed video coding
FC	Feedback channel
fps	Frames per second
GA	Genetic algorithm
GOP	Group of pictures
HD	Histogram of difference
HMCI	Hash-based motion compensated interpolation
IDCT	Inverse discrete cosine transform
LSB	Least significant bit

MCI Motion compensated interpolation

MSB Most significant bit

MSE Mean square error

OP Operation

OSI Open systems interconnection

PDF Probability density (distribution) function

PSNR Peak signal to noise ratio

QoS Quality of service

QP Quantization parameter

RCPT Rate-compatible punctured turbo (code)

RD Rate-distortion

RSC Recursive systematic convolutional (code)

SAD Sum of absolute differences

SFT Simple fusion technique

SI Side information

SNR Signal to noise ratio

SSE Sum of squared errors

TRD Traditional

UEP Unequal error protection

UMTS Universal mobile telecommunications system

WSN Wireless sensor network

WVSN Wireless video sensor network

WZ Wyner-Ziv

List of Figures

List of Tables

Résumé

Dans les techniques actuellement utilisées pour la compression vidéo, le codeur exécute un grand nombre d'opérations à cause des étapes d'estimation et de compensation du mouvement dans le codage des trames prédictives. Ceci conduit à un codeur très complexe. Dans un grand nombre d'applications, telles que la diffusion (en anglais "Broadcasting") ou le téléchargement de vidéos sur demande ("streaming video-on-demand"), la complexité du codeur n'est pas de grande importance puisque le processus de codage se fait dans de larges stations de base aux ressources suffisantes (puissance, mémoire,…). Dans ces applications, le décodeur est beaucoup plus simple (petite taille, bon marché), ce qui est un avantage puisque le décodage se fait plusieurs fois (pour différents utilisateurs, dans différents emplacements).

Cependant, d'autres applications exigent un scénario opposé, où un codeur simple est utilisé aux dépens d'un décodeur plus complexe. Les caméras mobiles sans fils transmettant la vidéo à une station de base fixe constituent un exemple typique de telles applications. En vue d'obtenir un codage moins complexe, le codage "Wyner-Ziv" des séquences vidéo a été proposé, dans lequel les trames individuelles sont codées indépendamment (codage intratrame) mais décodées conjointement (décodage intertrame).

Le but de cette thèse est de construire un codec Wyner-Ziv (WZ) qui permette de réaliser un codage conjoint source-canal de sources distribuées vidéo en se basant sur les turbo-codes non binaires. En premier lieu, nous réalisons une étude comparative qui montre les avantages des turbo-codes non-binaires par rapport aux codes binaires, en termes de convergence et de résistance au poinçonnage. Ensuite, nous montrons la supériorité des turbo-codes non binaires dans le domaine de la compression des sources distribuées, en simulant des sources théoriques transmises sur un canal binaire symétrique. Le gain en performances apporté par ces codes est aux dépens d'un décodage plus complexe.

Suite à la compression des sources théoriques, nous implémentons un système de codage distribué vidéo basé sur les turbo-codes quadri-binaires. Nous dérivons ensuite la limite théorique de compression qui peut être atteinte par le système dans le cas du codage source, ainsi que dans le cas du codage conjoint source-canal. Cette limite théorique est ensuite

utilisée dans le cadre d'une technique d'optimisation inter-couches (Cross-Layer) qui vise à éviter l'usage excessif du canal de retour dans un système opérant dans un environnement multi-utilisateurs (ou multi-capteurs). Ainsi, ce système se base sur le niveau de mouvement des séquences vidéo (couche application) et sur l'état du canal de transmission (couche de liaison de données et/ou couche physique) pour déterminer le débit de transmission nécessaire à chaque utilisateur (ou capteur) de façon à améliorer les performances globales du système. En se basant sur les résultats obtenus, nous proposons une technique de codage conjoint qui permet, d'un côté, d'estimer le débit de transmission nécessaire de chaque utilisateur, et d'un autre côté, d'optimiser le choix du paramètre de quantification de chaque trame. Ainsi, nous obtenons un système de codage vidéo distribué avec quantification variable et allocation dynamique du débit. En plus, nous étudions l'influence du codage H.264 des trames clés sur les performances globales du système pour le cas où un codeur H.264 Intra est intégré dans notre codec Wyner-Ziv.

Nous développons ensuite des algorithmes qui permettent de faire varier dynamiquement la taille du GOP (Group of Pictures) dans un système de codage distribué de sources vidéo. Ces algorithmes se basent principalement sur les calculs entropiques pour déterminer le débit de compression nécessaire ainsi que le type de codage (Intra ou Wyner-Ziv) de chaque trame, sans le besoin d'un canal de retour. Ceci rend le codec vidéo plus convenable aux applications de diffusion.

Finalement, vu la grande importance de l'information adjacente dans les systèmes de codage distribué et leur influence sur les performances de ces systèmes, nous proposons une nouvelle technique qui vise à optimiser la génération de ces informations au décodeur en utilisant des algorithmes génétiques.

A- Turbo-codes binaires et non-binaires

Les turbo-codes sont des codes correcteurs d'erreurs constitués par la concaténation (série ou parallèle) de deux codes convolutifs séparés par un entrelaceur. Initialement conçus sous forme binaire (recevant à leur entrée une série de bits), les codes non-binaires (prenant en entrée une série de symboles de M bits/symbole) ont été récemment introduits pour améliorer les performances des codes binaires en termes de convergence et de résistance au poinçonnage. En terme d'implémentation, le passage d'un turbo-code binaire à un turbo-code non-binaire nécessite la modification de l'algorithme de turbo-décodage de sorte à prendre en

considération toute les transitions possibles entre deux états consécutifs dans le treillis du code.

Bien que l'application la plus connue des turbo-codes, depuis leur invention, fût la correction d'erreurs, ils ont été récemment utilisés pour la compression de sources distribuées. L'idée se base sur le théorème de Slepian-Wolf: étant donné deux sources X et Y statistiquement dépendantes, Y étant comprimée séparément à sa limite entropique H(Y), X peut être transmise à un débit très proche de l'entropie conditionnelle H($X|Y$). Ceci aboutit à une structure simplifiée de l'encodeur pour la compression de la source X, à condition que Y soit parfaitement reçue au décodeur et utilisée comme information adjacente pour le décodage de X.

La dépendance statistique entre les sources X et Y peut être modélisée par un canal binaire symétrique (BSC). La compression est réalisée par le poinçonnage des bits de parité à la sortie du turbo-codeur. Les données systématiques ne sont pas transmises, mais plutôt remplacées par l'information adjacente disponible au récepteur.

La Figure R.1 montre le taux d'erreur binaire (BER) obtenu avec des turbo-codes binaires et non-binaires dans le cas d'une transmission à travers un canal à bruit blanc additif gaussien (AWGN). Le BER est présenté en fonction du rapport de l'énergie moyenne d'un bit d'information sur la densité spectrale du bruit E_b/N_0. Pour un rendement de codage de 4/5, nous remarquons un gain de 0.4 dB pour les turbo-codes duo-binaires et de 0.9 dB pour les quadri-binaires, par rapport aux turbo-codes binaires, à un BER de 10^{-5}. Ces gains deviennent respectivement 0.5 dB et 0.9 dB pour un rendement de 2/3.

Dans la Figure R.2, le BER est présenté en fonction de l'entropie conditionnelle H($X|Y$), pour les taux de compression 9:1 et 4:1, dans le cas d'une compression distribuée avec transmission sur un canal idéal. Nous remarquons que les turbo-codes quadri-binaires présentent les meilleures performances par rapport aux turbo-codes binaires et duo-binaires.

Dans le cas d'une application de codage conjoint source canal, les turbo-codes ont pour rôle de comprimer les données et, en même temps, de protéger ces données contre les erreurs de transmission. Dans la suite, nous adopterons les codes quadri-binaires pour la compression distribuées des sources vidéo transmises à travers des canaux bruités.

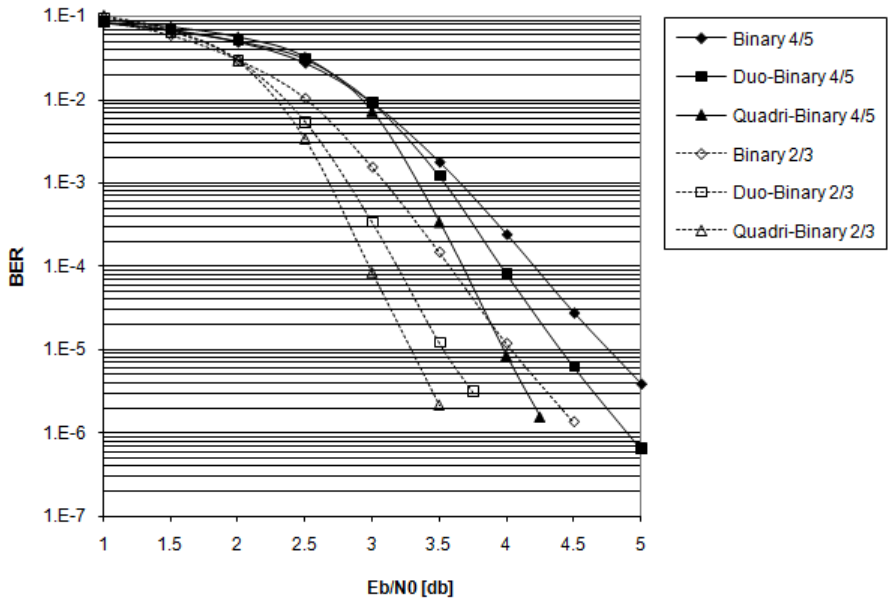


Figure R.1 BER obtenu avec des turbo-codes binaires et non-binaires dans le cas d'un codage correcteur d'erreurs avec des rendements de codage de 2/3 et 4/5, pour un canal AWGN.

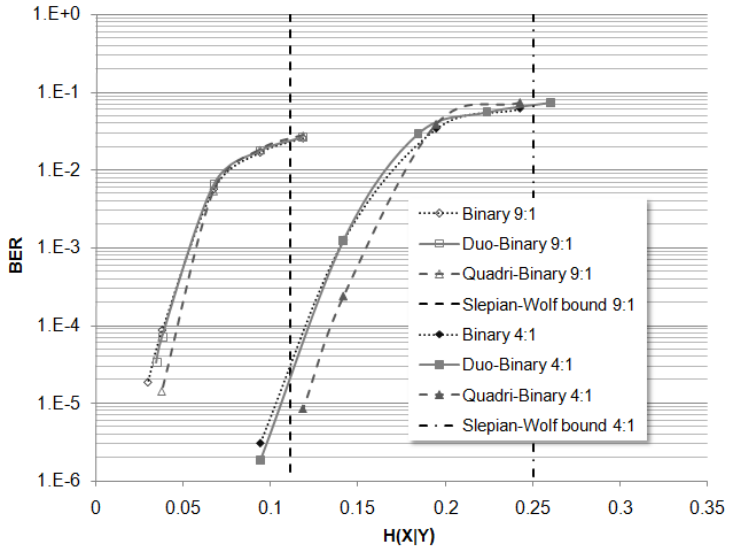


Figure R.2 BER obtenu avec des turbo-codes binaires et non-binaires dans le cas d'une compression distribuée avec des taux de compression 9:1 et 4:1.

B- Codage Wyner-Ziv des séquences vidéo

Le système de codage distribué vidéo (Distributed Video Coding ou DVC) considéré dans cette étude peut être représenté par le schéma-bloc de la Figure R.3. Les trames WZ subissent tout d'abord une quantification scalaire uniforme. Après turbo-codage, les données

systématiques sont rejetées alors que les bits de parité sont stockés dans un tampon. Dans le cas où un canal de retour existe, ces derniers sont partiellement transmis sur demande du décodeur, jusqu'à ce qu'un BER inférieur à 10^{-3} soit atteint. Dans le cas contraire (par exemple, application de diffusion), le système doit être capable d'estimer au préalable le nombre de bits de parité nécessaire pour une qualité de décodage acceptable, sans le besoin d'un canal de retour. Cette quantité peut être estimée à partir des limites théoriques de compression des trames WZ calculées par l'émetteur. Quant aux trames clés, elles sont codées avec un codeur intra-trames traditionnel et sont supposées être décodées sans erreur. Elles sont utilisées pour interpoler l'information adjacente, qui sera utilisée en tant que donnée systématique par le turbo-décodeur, ainsi que dans la reconstruction des trames WZ.

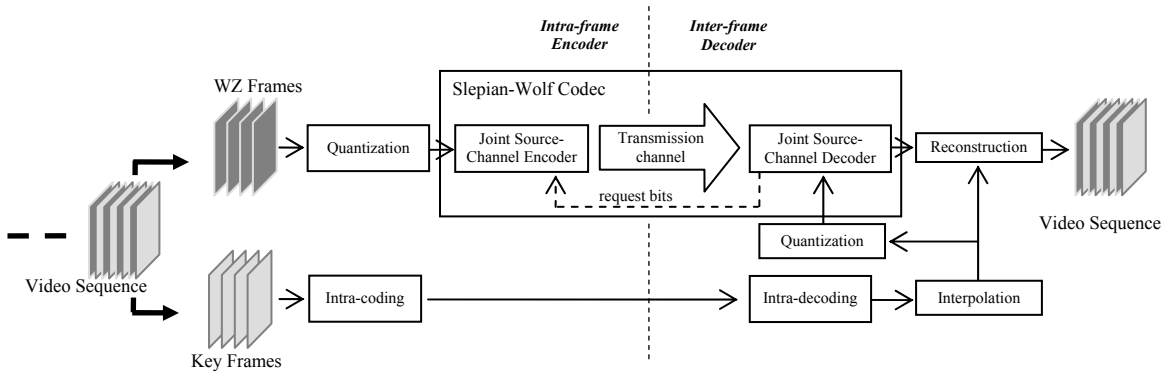


Figure R.3 Schéma-bloc du système de codage conjoint source-canal.

C- Estimation de la limite de compression

La limite théorique de compression est calculée, en l'absence de bruit, par l'entropie conditionnelle H($X|Y$), où X est la source discrète transmise et Y l'information adjacente. X et Y étant des pixels quantifiés à M bits, on peut écrire:

$$H\left(X \mid Y\right) = -\sum_{i=0}^{2^M-1}\sum_{j=0}^{2^M-1} P\left(Y=j\right)\cdot g\left[P\left(X=i \mid Y=j\right)\right], \tag{1}$$

où $g\left(x\right) = x\cdot \log_2\left(x\right)$. Par ailleurs, on peut vérifier que :

$$P\left(X-Y=d\right) = c\frac{\alpha}{2}e^{-\alpha|d_{X-Y}|}, \tag{2}$$

où $d_{X-Y} = 2^{8-M}(X-Y)$, α le paramètre de la distribution Laplacienne, et c est un facteur qui permet de prendre en compte la nature discrète et limitée du signal résiduel entre X et Y. On peut montrer que:

$$H(X|Y) = -\sum_{i=0}^{2^M-1}\sum_{j=0}^{2^M-1} c\frac{\alpha}{2}\frac{\exp\left(-\alpha\left|d_{i-j}\right|\right)}{L_{d_{i-j}}}\log_2\left(2^{M-1}c\frac{\alpha\exp\left(-\alpha\left|d_{i-j}\right|\right)}{L_{d_{i-j}}}\right), \quad (3)$$

où $L_{d_i} = 2^M - |i|$.

Dans le cas d'une transmission sur un canal bruité modélisé par un canal binaire symétrique de probabilité de transition p, la limite de compression devient:

$$H_b(X|Y) = \frac{H(X|Y)}{C(p)}, \quad (4)$$

où $C(p)$ représente la capacité du canal BSC.

Dans la Figure R.4, nous montrons le taux de compression (R: 1dB) obtenu en simulant le système de codage conjoint avec rétroaction, ainsi que la limite théorique inférieure (H: 1dB), pour chaque trame, obtenus avec E_S/N_0 = 1 dB et M = 4. Nous avons considéré 50 trames WZ de la séquence Carphone échantillonnée à 15 trames paires par seconde. Les courbes 'R' et 'H' sont obtenues pour une transmission sans bruit. On remarque, en l'absence de bruit, une différence de performance de la compression réalisée entre 0.06 et 0.14 par rapport aux courbes théoriques. Lorsque E_S/N_0 = 1 dB, la gamme de différence augmente à [0.11 ; 0.18]. Notons que les valeurs importantes des limites de compression pratiques ou théoriques correspondent à de faibles valeurs du paramètre α estimé à M = 4. En fait, une faible valeur de α indique un niveau de mouvement élevé dans la trame considérée, ce qui rend l'information adjacente correspondante moins fiable pour le processus de turbo-décodage. Par suite, une grande quantité d'informations de parité sera requise de la part du codeur.

Des résultats similaires ont été obtenus pour différentes séquences et différentes valeurs de M. Dans tous les cas, nous avons remarqué que le système de compression pratique présente un comportement comparable à celui des résultats théoriques: la limite de compression réalisable pour une trame dépend de son contenu et des conditions de transmission. Par suite, nous pouvons utiliser les résultats analytiques pour prédire le niveau de compression pour

chaque trame et l'utiliser dans un système de diffusion, sans le besoin excessif d'un canal de retour comme dans un système WZ classique.

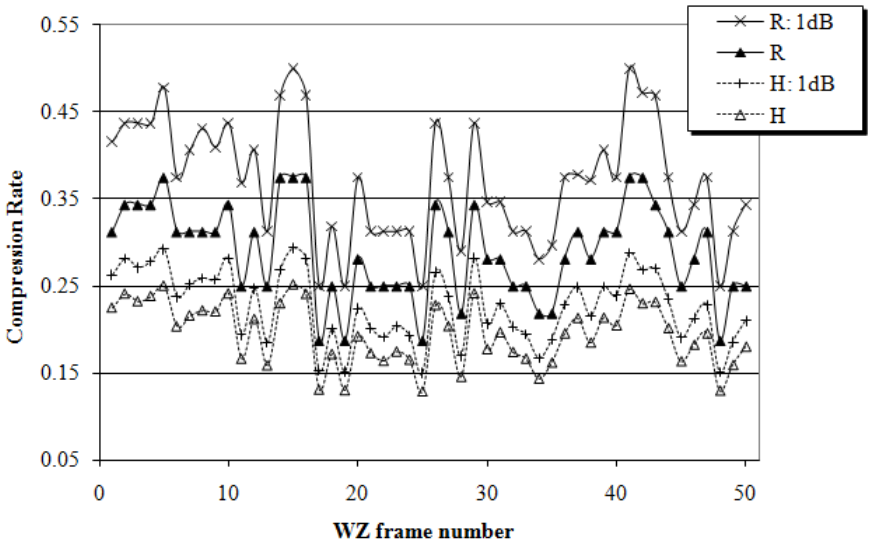


Figure R.4 Limites de compression théoriques et pratiques des trames WZ de la séquence Carphone avec $M = 4$.

D- Applications à la suppression du canal de retour

Dans cette étude, nous présentons deux applications différentes se basant sur la suppression du canal de retour. La première application consiste à utiliser la limite théorique de compression estimée précédemment de sorte à permettre une allocation dynamique de débit dans un système multi-utilisateurs, tout en prenant en considération la contrainte de débit global du système, le canal de transmission entre la station de base et chacun des utilisateurs, ainsi que le niveau de mouvement dans chacune des scènes vidéo à transmettre. La deuxième consiste à faire varier dynamiquement la taille du GOP en se basant sur notre technique d'estimation de débit, sans le besoin d'un canal de retour comme dans les algorithmes déjà existants dans la littérature.

D.1. Allocation dynamique de débit

Considérons un réseau de N utilisateurs transmettant des flux vidéo à une station de base fixe. Soit R le débit total permis (en bps), $\rho_{f,n}$ le taux de compression de la trame f à l'utilisateur n, $M_{f,n}$ le paramètre de quantification de cette trame et $A_{f,n} = M_{f,n}\rho_{f,n}$ son débit exprimé en bits par pixel (bpp). Dans notre système, $0 \le \rho_{f,n} \le 1/2$ et $M_{f,n} \in \{1,2,4\}$. Le canal de transmission entre chaque utilisateur et la station de base est modélisé par un BSC avec une

probabilité de transition *p*. Dans ce cas, la limite de compression pour la trame *f* à l'utilisateur *n* est exprimée par :

$$H_{f,n} = -\frac{1}{C(p)} \sum_{i=0}^{\left(2^{M_{f,n}}-1\right)} \sum_{j=0}^{\left(2^{M_{f,n}}-1\right)} c\frac{\alpha}{2}\frac{\exp\left(-\alpha\left|d_{i-j}\right|\right)}{L_{d_{i-j}}} \log_2\left(2^{M_{f,n}-1} c\frac{\alpha\exp\left(-\alpha\left|d_{i-j}\right|\right)}{L_{d_{i-j}}}\right). \quad (5)$$

Au lieu de répartir le débit global de façon égale entre les différents utilisateurs, nous proposons une répartition proportionnelle à la limite entropique de l'équation (5), où la trame *f* à l'utilisateur *n* aura le débit :

$$R_{f,n} = \frac{H_{f,n}}{\sum_{n=1}^{N} H_{f,n}} R\,. \quad (6)$$

Par ailleurs, puisque le débit $R_{f,n}$ peut être obtenu avec différentes combinaisons des paramètres $M_{f,n}$ et $\rho_{f,n}$, nous cherchons à optimiser le choix des couples ($M_{f,n}$, $\rho_{f,n}$) afin d'améliorer les performances du système.

Après une analyse approfondie des performances du système observées pour différentes valeurs de *M*, nous avons remarqué que dans la plupart des cas, choisir la plus petite valeur possible de *M* aboutit au meilleur résultat. En effet, en réduisant le nombre de niveaux de quantification, le système est capable de transmettre plus de bits de parité pour mieux protéger le flux de données transmis. Toutefois, dans certains cas, le débit affecté à un certain utilisateur est suffisant pour permettre la protection contre les erreurs de transmissions quand une plus grande valeur de *M* est choisie, et de meilleures performances sont obtenues dans ce cas. En fait, nous avons remarqué que le comportement du système est directement lié au rapport :

$$C_{f,n}(M_{f,n}) = \frac{A_{f,n}}{H_{f,n}(M_{f,n})} \quad (7)$$

Par suite, nous commençons par déterminer de façon expérimentale les seuils T_1, T_2 et T_4 qui indiquent la valeur moyenne du rapport défini en (7) garantissant un décodage correct de la trame WZ transmise, pour $M_{f,n}$ = 1, 2 et 4 respectivement.

Par ailleurs, il faut noter que dans le cas d'une quantification dynamique, notre système ne connait pas le paramètre de quantification M à l'avance. Pour cette raison, nous commençons par estimer (6) en supposant $M = 8$ (puisque nos séquences vidéo sont initialement codées à 8 bpp). Une fois les $R_{f,n}$ (et par conséquent $A_{f,n}$) obtenus, le système détermine la plus grande valeur de $M_{f,n}$ pour laquelle $C_{f,n}(M_{f,n}) > T_{M_{f,n}}$. Si aucune des valeurs possibles de $M_{f,n}$ ne vérifie cette condition, alors $M_{f,n} = 0$ et la trame WZ ne sera pas transmise. Dans ce cas, elle sera remplacée par l'information adjacente correspondante au décodeur. Finalement, après avoir déterminé $M_{f,n}$, le taux de compression sera déterminé comme $\rho_{f,n} = A_{f,n} / M_{f,n}$ (pour $M_{f,n} \neq 0$).

Notons que la description précédente ne prend pas en considération l'effet du codage des trames clés. Pour cette raison, nous avons considéré le cas d'un codage H.264-intra des trames clés et étudié différentes configurations des débits et des niveaux de quantification des trames clés et WZ. Ainsi, nous avons trouvé que le paramètre de quantification QP (en codage H264-Intra) des trames clés ne doit pas dépasser 30 pour une compression efficace, d'un côté, et ne doit pas être inférieur à 20 pour éviter les grandes fluctuations de PSNR entre les trames clés et les trames WZ, de l'autre. Dans la suite, nous fixons QP à 30. La valeur de QP détermine le débit R_S de codage source des trames clés. Pour protéger ces trames contre les erreurs, le flux H.264 est turbo-codé avec un rendement de codage r déterminé à partir de tables de performances pré-établies. Ces tables, disponibles à l'encodeur, permettent de déterminer les valeurs de r nécessaires, suivant l'état du canal de transmission, pour réaliser un turbo-décodage quasi-correct des trames clés. Le débit total des trames clés sera donc :

$$R_{Key} = \sum_{i=1}^{N} R_{S,i} / r_i \,. \tag{8}$$

Dans ce cas, le débit restant pour les trames WZ sera : $R_{WZ} = R - R_{key}$. Ainsi, R_{WZ} pourra être utilisé dans l'algorithme précédent, au lieu de R, pour l'allocation dynamique de débit avec quantification adaptative (ARAQ-D).

Dans nos simulations, nous considérons N=3 (les séquences Carphone, Foreman, et Mother-Daughter). La Figure R.5 montre les courbes de débit-distorsion obtenues avec un système traditionnel (TRD) où le débit des trames WZ est réparti de façon égale entre les utilisateurs et où M est fixe, ainsi que celles obtenues avec l'algorithme ARAQ-D proposé. Nous

pouvons vérifier qu'avec TRD, lorsqu'un débit peut être atteint avec différentes valeurs de *M*, les meilleures performances sont obtenues pour la valeur de *M* la plus petite. Il est clairement remarquable que l'algorithme ARAQ-D permet non seulement d'optimiser le choix de *M* pour de tels cas, mais aussi d'améliorer les performances globales par rapport au système TRD.

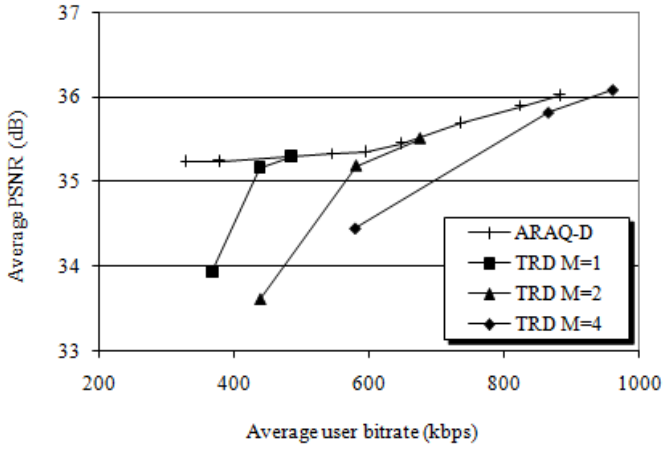


Figure R.5 Courbes de débit-distorsion obtenues avec un système traditionnel (TRD) et avec l'algorithme proposé (ARAQ-D).

D.2. Contrôle automatique de la taille du GOP

Le décodage correct des trames WZ à des débits de transmission abordables nécessite une SI de qualité élevée. Cette qualité se dégrade rapidement lorsque les trames clés sont éloignées, c.à.d. lorsque la taille du GOP augmente. D'un autre côté, le fait de diminuer le nombre des trames clés permet de réduire le débit moyen de transmission d'un GOP, d'où la nécessité d'un compromis.

Lorsque le mouvement est lent dans une séquence vidéo, la corrélation entre les trames successives augmente. Le fait de diminuer le nombre de trames clés peut permettre de mieux exploiter cette propriété, puisque dans ces régions le codage WZ aboutit à de meilleures performances en termes de débit-distorsion. Dans les zones de fort mouvement, la taille du GOP sera réduite à 1, puisque le codage Intra devient plus performant que le WZ.

Soit S_{max} la taille maximale permise d'un GOP. Pour une certaine taille S de GOP, on note par F_0 la trame clé, F_1, F_2, … , F_{S-1} les trames WZ, et F_S la trame clé du GOP suivant. On note respectivement par R_i et $PSNR_i$ le débit moyen et le PSNR de la $i^{ème}$ trame du GOP. L'algorithme de décision de la taille d'un GOP opère de la façon suivante :

Initialisation : $S = 1$.

Tant que $S \leq S_{max}$*:*

Si S==1, aller à l'étape 5, autrement:

Etape 1**:** *Interpolation des trames* F_0 *et* F_S

Puisque l'étape d'estimation de mouvement n'est pas permise au codeur à cause de sa complexité, une interpolation moyenne (AVI) est utilisée. La trame interpolée sert comme estimation grossière de la SI qui sera générée au récepteur (par compensation de mouvement), pour le décodage de la trame WZ $F_{\lfloor S/2 \rfloor}$, localisée à mi-distance entre F_0 et F_S.

Etape 2**:** *Estimation du débit nécessaire* $R_{\lfloor S/2 \rfloor}$

En se basant sur la trame WZ $F_{\lfloor S/2 \rfloor}$ et sa SI estimée, le codeur détermine sa limite de compression inférieure d'après la méthode de calcul entropique introduite ultérieurement, qui permet de prendre en compte le niveau de mouvement dans la trame. Le taux de compression de la trame WZ est ensuite obtenu en multipliant la limite inférieure par une constante T_M dont la valeur dépend du nombre M de bits de quantification par pixel. Ce taux permet de déterminer $R_{\lfloor S/2 \rfloor}$.

Etape 3**:** *Calcul de* $PSNR_{\lfloor S/2 \rfloor}$

En utilisant la trame WZ $F_{\lfloor S/2 \rfloor}$ et sa SI estimée, le codeur peut déterminer une estimation $F'_{\lfloor S/2 \rfloor}$ de la trame qui sera obtenue au récepteur après turbo-décodage, en quantifiant tout d'abord la trame WZ et en réalisant sa reconstruction en se basant sur la SI estimée. Le PSNR est alors calculé entre $F_{\lfloor S/2 \rfloor}$ et $F'_{\lfloor S/2 \rfloor}$.

Etape 4**:** *Répétition des étapes 1 à 3 jusqu'à ce que les estimations de débit et de PSNR soient obtenues pour toutes les trames du GOP.*

La Figure R.6 montre un exemple de toutes les interpolations nécessaires pour une taille de GOP $S = 4$. A chaque estimation, si le processus d'interpolation fait intervenir une trame clé à l'instant k (Ex : trames F_0 et F_4 de l'interpolation 1 dans la Figure R.6), la vraie trame F_k (ou sa version décodée H.264) est utilisée puisqu'elle sera disponible au décodeur. Par contre, s'il fait intervenir une trame WZ précédemment décodée (Ex : trame F_2 pour les interpolations 2 et 3 dans la Figure R.6), l'estimation F'_k de cette dernière (obtenue par l'étape 3 de l'itération

précédente de l'algorithme) est utilisée au lieu de F_k , car F'_k est une estimation plus fidèle de la trame qui sera disponible au décodeur (F_k n'étant pas connue de ce dernier).

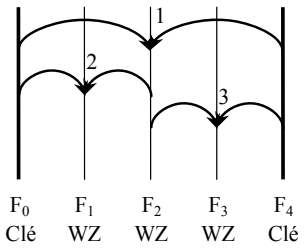


Figure R.6 Etapes d'interpolation pour une taille de GOP $S = 4$.

Etape 5**:** *Estimation du débit et du PSNR moyens obtenus avec un GOP de taille S, respectivement par:*

$$R_{av}^{S} = \frac{1}{S}\sum_{j=0}^{S-1} R_j \text{ et } PSNR_{av}^{S} = \frac{1}{S}\sum_{j=0}^{S-1} PSNR_j .$$

Etape 6**:** *Calcul du rapport* $\lambda_{av}^{S} = PSNR_{av}^{S} / R_{av}^{S}$.

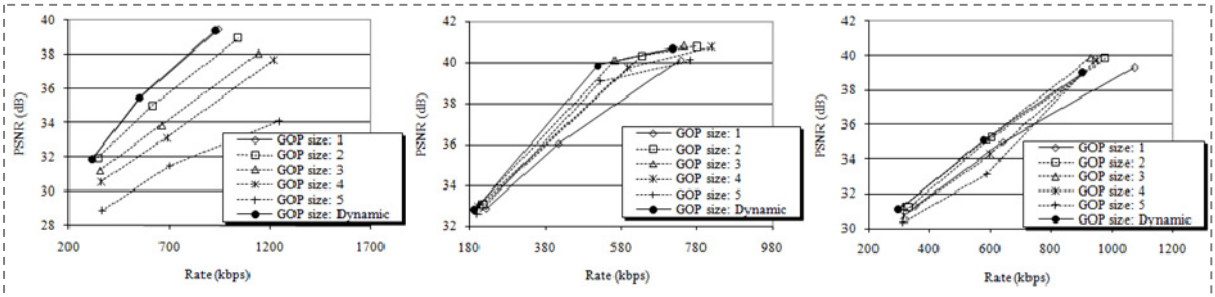
Etape 7**:** *Incrémentation de la taille du GOP:* $S = S + 1$.

Les meilleures performances en terme de débit-distorsion sont obtenues en maximisant le PSNR par unité de débit. Par conséquent, le système décide de la taille optimale L du GOP en prenant: $L = \arg\max_{k=1,2,\ldots,S_{\max}} \left(\lambda_{av}^{k}\right)$.

Ainsi, si L=1, une seule trame codée en Intra est transmise. Sinon, une trame clé (Intra) est transmise, suivie de L-1 trames WZ. Cette procédure est répétée à chaque début de GOP.

Dans nos simulations, nous considérons les séquences Foreman, Grandmother et Salesman. La Figure R.7 représente les résultats en termes des courbes de débit-distorsion obtenues en moyennant le débit et le PSNR des trames clés et WZ. Nous avons choisi S_{max} = 5 pour l'algorithme de variation dynamique du GOP (appelé VDGOP). Nous remarquons que, pour la séquence Foreman, lorsque la taille S du GOP est fixe, les performances diminuent lorsque S augmente. Les meilleures performances moyennes sont obtenues lorsque toutes les trames sont codées Intra. Ceci est dû au niveau de mouvement important dans cette scène, qui conduit à une SI de mauvaise qualité au décodeur lorsque les trames clés sont éloignées les unes des autres. Cependant, on remarque que le VDGOP, qui présente des performances

similaires au cas du codage Intra à bas débit, permet un gain de 35 kbps pour un PSNR de 39.5 dB. Les séquences Grandmother et Salesman, caractérisées par un niveau de mouvement plus faible que Foreman, présentent un comportement différent. Lorsque la taille du GOP est fixe, les meilleures performances sont obtenues pour S=3, en particulier à haut débit. Quant au VDGOP, il est plus performant que tous les autres schémas de codage.



(a) Foreman (b) Grandmother (c) Salesman

Figure R.7 Courbes de débit-distorsion pour différentes séquences, obtenues avec une taille de GOP fixe ou variable.

Toutefois, une perte de 0.4 dB peut être observée avec la séquence Salesman à 900 kbps en utilisant VDGOP, par rapport au GOP fixe de taille 3. Ceci est dû principalement à une inadéquation entre la SI disponible du côté du codeur (estimée par interpolation moyenne), et celle disponible au niveau du décodeur (obtenue par compensation de mouvement). En fait, de tels cas se produisent rarement.

Il est important de noter qu'en pratique, la taille optimale du GOP ne peut pas être connue à l'avance, sans effectuer le codage et le décodage de la séquence pour différentes tailles de GOP. Par contre, avec notre algorithme VDGOP, le système est capable de déterminer la taille optimale du GOP et, en même temps, d'améliorer les performances de débit-distorsion, puisque la taille du GOP est variée dynamiquement le long de la séquence, en prenant en compte le niveau de mouvement dans chaque scène.

Finalement, nous avons observé les pourcentages des tailles de GOP optimales (L) obtenues par le VDGOP, pour chaque séquence et pour les différentes valeurs de M (Table R.1). Pour la séquence Foreman, plus de 85 % des GOP ont une taille 1, ce qui indique que le système sélectionne le codage Intra H.264 la plupart du temps. Plus de variations de L sont observées avec Grandmother et Salesman. Pour la séquence Grandmother avec M = 1, aucun GOP n'a pour taille 1, alors que L = 5 pour 40 % des GOPs. De même avec Salesman, seulement 3.7% des GOPs ont pour taille 1, alors que L = 5 pour 56%.

Gop size:			1	2	3	4	5
Foreman	M=1	%	100	0	0	0	0
	M=2		91.1	6.7	2.2	0	0
	M=4		87.4	10.3	2.3	0	0
Grandmother	M=1		0	4	20	36	40
	M=2		0	4.2	16.7	37.5	41.6
	M=4		23.1	33.3	20.5	2.6	20.5
Salesman	M=1		3.7	22.2	18.5	0	55.6
	M=2		18.2	12.1	39.4	0	30.3
	M=4		22.2	7.4	3.7	0	66.7

Table R.1 Pourcentage des tailles de GOP pour les différentes séquences.

E- Fusion de l'information adjacente par les algorithmes génétiques

La technique utilisée pour la génération de la SI influe énormément sur la qualité de décodage des trames WZ, ainsi que sur leur débit de transmission nécessaire. Les méthodes d'interpolation utilisées dans notre étude, jusqu'à maintenant, sont l'interpolation moyenne et l'interpolation par compensation du mouvement. Plusieurs autres techniques ont été proposées dans la littérature pour améliorer la qualité de la SI générée au décodeur. Toutes ces techniques présentent des avantages et des inconvénients suivant le type et la vitesse de mouvement dans la vidéo codée. Par suite, nous nous proposons d'introduire une technique basée sur les algorithmes génétiques (GA) permettant le fusionnement de plusieurs SI générées par les techniques d'interpolation précédemment développées. Le but du GA est de déterminer la meilleure candidate SI dans différentes régions de la trame vidéo. A notre connaissance, une telle stratégie n'a jamais été utilisée en DVC.

Le GA est un outil d'optimisation basé sur la théorie de l'évolution en génétique. Son application nécessite tout d'abord une représentation génétique de l'espace des solutions, ainsi qu'une fonction d'évaluation de chaque solution possible. Un chromosome (solution candidate) sera défini par un ensemble de pixels d'une trame, arrangés sous la forme d'un bloc. L'ensemble des chromosomes dans l'espace des solutions sera appelé "population". A l'origine, pour un certain bloc d'une trame WZ, chacun des blocs à la même position dans les trames SI, générées par les différentes méthodes d'interpolation, constitue une solution possible. La similarité entre un chromosome et le bloc WZ correspondant est déterminé par la fonction d'évaluation. Puisqu'en DVC seules les trames clés sont disponibles au décodeur, il est indispensable de transmettre certaines informations sur les trames WZ de façon à permettre l'évaluation d'une certaine solution. Etant donné que la complexité de codage est

une contrainte incontournable en DVC, l'extraction de ces informations supplémentaires ne doit pas constituer une charge significative pour le codeur, de même que le débit nécessaire pour leur transmission. Pour cette raison, nous avons opté pour l'utilisation d'un "hash-DVC". Dans ce dernier, un sous-ensemble des coefficients DCT (Discrete Cosine Transform) de chaque trame WZ est transmis sous la forme d'un mot d'empreinte (en anglais "hash word") au décodeur. Dans notre système, nous nous proposons d'utiliser ces empreintes pour le calcul de la fonction d'évaluation. Ainsi, par rapport au hash-DVC, aucune charge calculatoire n'est rajoutée par notre système au codeur. Par suite, nous définissons la fonction d'évaluation d'un chromosome par :

$$F = \frac{K_1}{K_2 + K_3 D}, \tag{9}$$

où D est l'erreur quadratique moyenne (EQM) entre l'empreinte reçue pour le bloc WZ et l'empreinte générée dans le décodeur pour le chromosome en cours d'évaluation, et K_1, K_2 et K_3 sont des paramètres réglables. Pour chaque bloc d'une trame WZ, l'algorithme GA proposé opère comme suit:

1ère étape: Initialisation

La population initiale est créée en considérant, tout d'abord, les trames SI générées par la méthode d'interpolation moyenne (AVI), par l'interpolation par compensation de mouvement à vecteurs symétriques (MCI) et par l'interpolation par compensation de mouvement basée sur les empreintes (HMCI). En HMCI, l'estimation de mouvement se fait par minimisation de l'EQM entre l'empreinte reçue et celle du bloc SI candidat. A ces trois candidats, on ajoute les deux trames clés utilisées dans les processus d'interpolation, d'où un total de cinq candidats initiaux qui seront dupliqués de façon proportionnelle à leur niveau d'adaptation (estimé par la fonction d'évaluation), jusqu'à ce que la population atteigne une taille désirée S_p.

2ème étape: Croisement

Les chromosomes de la population courante sont aléatoirement entrelacés et arrangés en paires. Une paire est combinée par échange de *gènes* (pixels), avec une probabilité P_c, pour produire deux chromosomes « enfants » comme illustré sur la figure 1. Autrement dit, les parents demeurent inchangés avec la probabilité 1-P_c. Quant à la position du croisement, elle est choisie de façon aléatoire.

Le croisement peut être vertical, horizontal, ou bien les deux à la fois. Dans ce dernier cas, on parle d'un GA bidirectionnel, ou 2D-GA. D'autres directions (diagonales par exemple) peuvent être utilisées pour le croisement, mais dans notre étude, nous considérons seulement deux directions pour la simplicité.

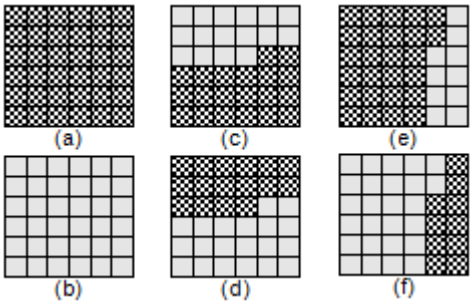


Figure R.8 Exemple de croisement de deux parents (a) et (b) pour produire deux enfants (c) et (d) par croisement vertical, ou (e) et (f) par croisement horizontal.

3ème étape: Mutation

Elle consiste à inverser un bit d'un gène d'un chromosome choisi au hasard. La mutation a lieu avec une faible probabilité P_m et permet d'étendre l'espace des solutions, de façon à diminuer le risque de converger vers un optimum local.

4ème étape: Evaluation et sélection

La fonction d'évaluation (fitness) est tout d'abord estimée pour chaque chromosome de la population courante. Ensuite, parmi les S_p chromosomes de la population, seuls les S_f chromosomes les plus « adaptés » (c.à.d. ayant la plus grande évaluation) sont sélectionnés, tandis que les S_p- S_f chromosomes restants sont abandonnés pour permettre l'introduction de nouveaux chromosomes. A noter que S_f ne doit pas être très proche de S_p, afin de ne pas permettre aux chromosomes les moins adaptés de se reproduire, et ne doit pas être non plus trop faible afin de permettre à une grande variété de chromosomes d'échanger leurs gènes et de se reproduire.

5ème étape: Duplication

Chacun des chromosomes restants est dupliqué un nombre de fois proportionnel à son évaluation, de façon à atteindre la taille de population S_p.

Les étapes 2 à 5 sont répétées jusqu'à un certain nombre maximal (I_{max}) d'itérations. Finalement, le chromosome le plus adapté est retenu pour être utilisé comme SI dans le décodage du bloc correspondant de la trame WZ.

Les performances du GA sont largement affectées par ses différents paramètres. En passant d'une itération à l'autre, les solutions possibles les plus adaptées ont plus de chance de survivre, tandis que les candidats les moins adaptés sont éliminés. Par conséquent, la fonction d'évaluation est un facteur important qui influe sur la qualité finale de la SI. Le nombre I_{max} d'itérations maximal nécessaire pour la convergence du GA dépend de la taille S_p de la population et du nombre S_f de candidats survivants dans chaque itération.

Notons, que le GA peut opérer au niveau des trames au lieu des blocs. Dans ce cas, une trame entière constituera un chromosome. Puisque les opérations de croisement et de mutation affecteront seulement un petit nombre de blocs dans les trames "parents", seuls les blocs où une opération génétique a eu lieu nécessiteront une mise à jour de leurs coefficients DCT. Ceci aboutit à un GA plus simple.

Lorsque le niveau de mouvement est très important dans certaines parties de la vidéo, les techniques d'interpolation utilisées pour initialiser le GA échouent dans ces parties, ce qui se répercute sur les performances du GA. Dans ces conditions, il peut s'avérer plus avantageux d'adopter tout simplement la DCT inverse (IDCT) des mots d'empreinte reçus. Cependant, le bloc IDCT ne peut être introduit comme candidat dans le GA puisqu'avec la fonction d'évaluation adoptée, le GA convergerait constamment vers ce bloc. Pour cette raison, nous avons choisi d'incorporer la IDCT de la façon suivante dans la technique de fusion: la fonction d'évaluation F_{GA} du meilleur chromosome retenu comme sortie du GA est comparée à un seuil F_T: si $F_{GA} > F_T$, le système adopte la sortie du GA comme SI, sinon, il adopte le bloc correspondant de la trame IDCT. Dans la suite, l'application de cette méthode sera indiquée par un signe « + » (par exemple: GA+).

Finalement, nous avons également comparé les performances du GA avec un algorithme de fusion simple (SFT) qui, pour chaque bloc de la trame courante, choisit comme SI le meilleur bloc parmi les cinq candidats initiaux, sans l'exécution du GA.

Quatre séquences vidéo ont été testées dans nos simulations: Carphone (300 trames), Foreman (300 trames), News (300 trames), et Trevor (150 trames). Elles sont échantillonnées au débit de 30 trames/s et codées avec une taille de GOP 2. Les mots d'empreinte transmis

par le codeur constituent 1/8 des coefficients DCT de chaque trame WZ. La taille de bloc considérée est de 16x16 pixels. Quant aux paramètres du GA, les suivants ont d'abord été choisis après un certain nombre de tests: $S_p = 60$, $S_f = 40$, $I_{max} = 10$, $P_c = 0.8$, $P_m = 0.01$, $K_1 = 1$, $K_2 = 0$, et $K_3 = 1$.

Nous estimons tout d'abord les performances du système en fonction du PSNR entre la trame SI générée par le GA et la trame WZ correspondante. Considérons pour commencer le GA opérant au niveau des trames. Une analyse statistique des résultats nous a permis de constater un gain en PSNR pour la plupart des trames. Par exemple, pour la séquence Trevor, nous avons remarqué un gain en PSNR pour plus de 85% des trames. Un gain qui dépasse 1 dB est observé pour environ 70% des trames par rapport à AVI, 15% par rapport à MCI, et 5% par rapport à HMCI. Le pourcentage des trames avec un gain supérieur à 1 dB est de 50%, 25% et 12%, dans la séquence Carphone, 78%, 40% et 2% dans la séquence Foreman, et 75%, 75% et 40% dans la séquence News, par rapport à AVI, MCI et HMCI, respectivement.

Dans le cas du GA+ avec $F_T = 0.03$, le nombre de trames ayant un gain supérieur à 1 dB augmente de 15% à 40% pour Carphone et de 8% à 20% pour Foreman. L'amélioration a été négligeable avec les deux autres séquences, puisqu'elles sont plutôt caractérisées par un faible mouvement par rapport à Carphone et Foreman. D'autre part, de faibles pertes en performances sont observées pour toutes les séquences. En fait, la perte ne dépasse pas 1 dB, et affecte moins de 5% de l'ensemble des trames.

De meilleures performances sont obtenues avec un GA opérant au niveau des blocs au lieu des trames, puisque le GA en bloc s'adapte mieux aux variations locales. Pour $F_T = 0.02$, la proportion des trames ayant un gain supérieur à 1 dB devient 85% par rapport à AVI, 62% par rapport à MCI, et 14% par rapport à HMCI avec la séquence Trevor. Ces pourcentages deviennent 86%, 78% et 62%, avec Carphone, 77%, 42% et 10% avec Foreman, et 80%, 80% et 55% avec News, par rapport à AVI, MCI et HMCI, respectivement. Des performances légèrement meilleures ont été observées avec 2D-GA et 2D-GA+.

Afin d'évaluer les performances de notre algorithme génétique 2D-GA+ en termes de débit-distorsion, nous avons codé les différentes séquences avec notre codec WZ. Nous comparons nos résultats avec ceux obtenus avec la méthode SFT+ (SFT combinée à la IDCT), d'un côté, et la méthode d'interpolation HMCI, de l'autre, puisqu'elle présente les meilleures performances par rapports à AVI et MCI. D'après la Figure R.9, nous observons, dans le cas de SFT+, un gain moyen de 3 dB avec Carphone, 1 dB avec Foreman et Trevor, et 6 dB avec

News, par rapport à la HMCI. Un gain supplémentaire de 0.3 dB est observé pour Carphone et Foreman, 0.6 dB pour Trevor, et 1 dB pour News, en utilisant 2D-GA+.

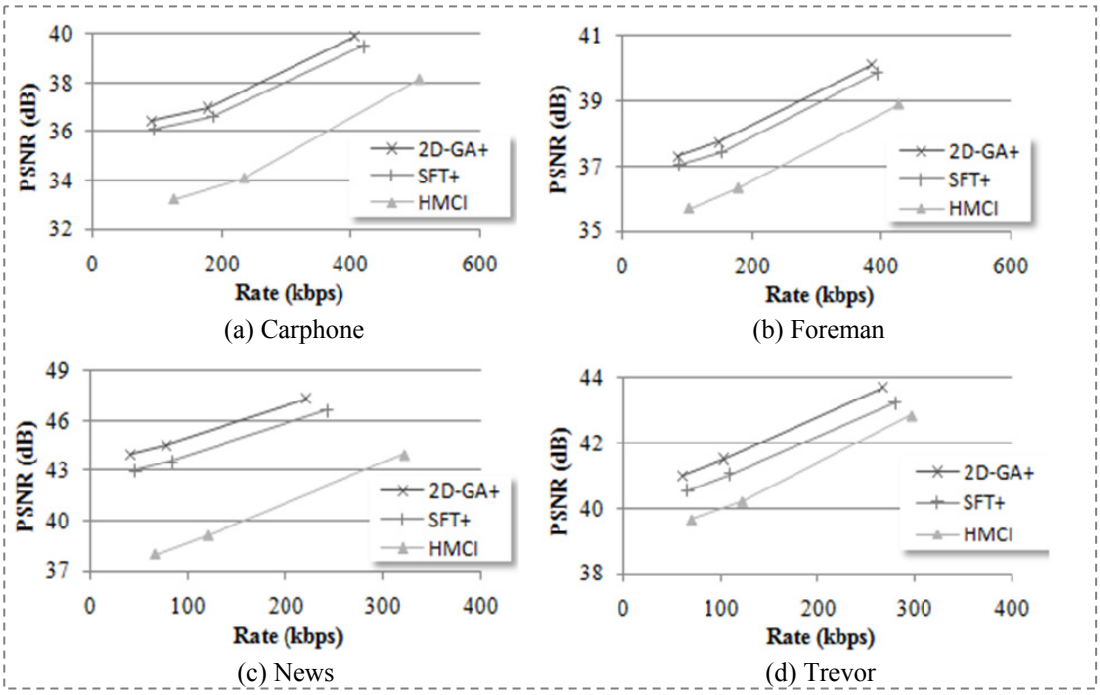


Figure R.9 Courbes de débit-distorsion pour les différentes séquences, avec la SI obtenue par HMCI, SFT+ et 2D-GA+.

Dans toutes nos simulations, nous remarquons des performances exceptionnelles du GA avec la séquence News. En fait, ceci est dû à la nature très particulière du mouvement dans cette séquence. La scène présente deux personnages presque statiques (sans mouvement) à l'avant plan, et un couple de danseurs à l'arrière plan alternant des mouvements lents et rapides. La Figure R.10 montre une suite de trames WZ consécutives de la séquence News. La première ligne montre les trames originales, la deuxième montre les trames SI obtenues par le GA, la troisième montre l'erreur quadratique moyenne entre les deux précédentes, quantifiée à 256 niveaux de gris (8 bits), où une intensité plus claire indique une plus grande erreur, et la dernière ligne montre le PSNR des trames SI obtenues par HMCI et GA. On peut remarquer que l'erreur dominante (régions claires) est surtout concentrée à l'arrière scène (danse du couple). L'erreur augmente chaque deux trames en raison de l'alternance de la nature du mouvement (lent / rapide) qui peut être observée aussi dans les fluctuations du PSNR et du gain par rapport à HMCI. Ce qui montre que le GA est particulièrement avantageux dans le cas de scènes complexes contenant des parties à niveaux de mouvement différents.

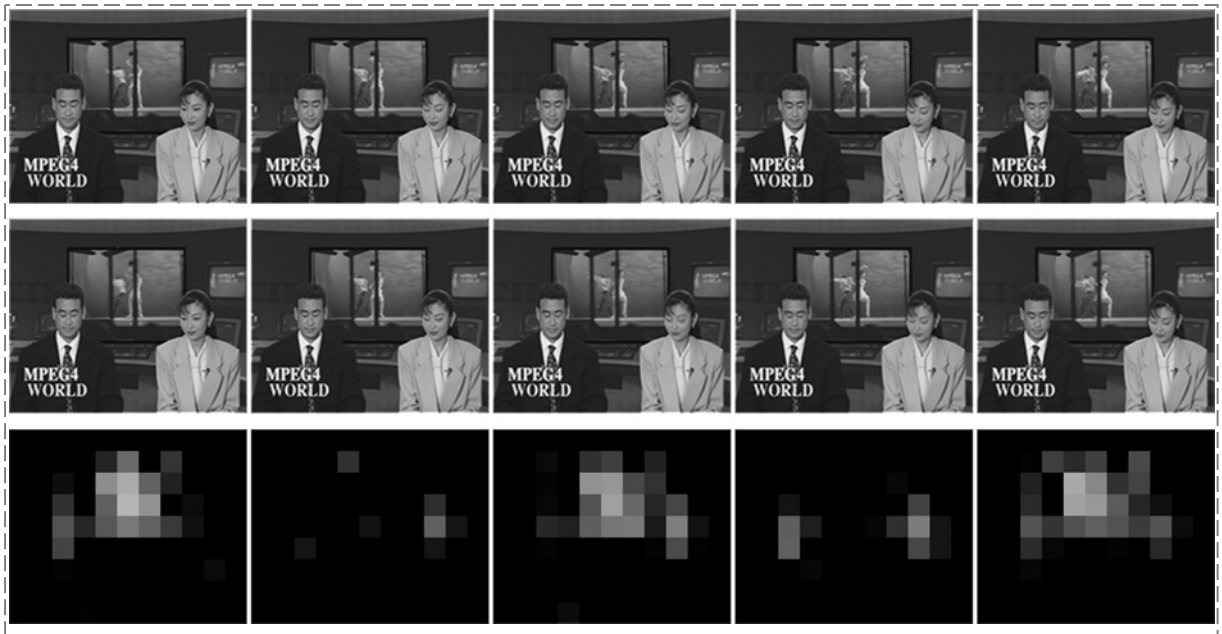


Figure R.10 Série de trames WZ consécutives (trame 111 à 115) de la séquence News. Première rangée: trames originales. Deuxième rangée: SI obtenue par le GA. Troisième rangée: erreur quadratique moyenne. Dernière rangée: PSNR des trames SI.

Notons, pour finir, qu'un gain en performances a pu être réalisé en faisant varier les paramètres K_1, K_2 et K_3. En fait, lorsque $K_1 = 1$, $K_2 = 0$, et $K_3 = 1$ ($F = 1 / D$), une très faible valeur de D, pour un certain chromosome, résultera en une très grande valeur de F. Dans ce cas, le chromosome dominera la population dans l'itération suivante (après les étapes de sélection et de duplication), et le GA convergera rapidement. Ceci n'est pas souhaitable dans la mesure où la métrique que nous utilisons ne constitue pas un indicateur exact de similarité. Bien que ce phénomène ne soit pas fréquent, nous avons remarqué, après un certain nombre de tests, qu'un gain assez remarquable est obtenu pour les valeurs $K_1 = 10$, $K_2 = 100$, et $K_3 = 1$. Par exemple, pour la séquence News, un gain supérieur à 0.5 dB est observé pour plus de 10% des trames.

F- Conclusions

Dans cette étude, nous avons commencé par étudier les performances des turbo-codes binaires et non-binaires dans les applications de codage canal, de compression de sources distribuées, et en codage conjoint source-canal. Après une étude comparative, nous avons choisi d'adopter les turbo-codes quadri-binaires dans le codec Wyner-Ziv des séquences vidéo. Nous avons ensuite déterminé la limite théorique de compression réalisable par ce codec, et que nous avons utilisée pour l'estimation du débit de transmission nécessaire au bon

décodage des trames WZ sans le besoin d'un canal de retour. La suppression du canal de retour a par ailleurs permis de développer un système WZ pour une application multi-utilisateurs qui effectue une allocation dynamique de débit avec une quantification adaptative, en prenant en considérations le niveau du mouvement dans les séquences vidéo, ainsi que le niveau du bruit dans le canal de transmission. Un algorithme qui permet de faire varier dynamiquement la taille du GOP a ensuite été élaboré. Finalement, dans le but d'améliorer la qualité de l'information adjacente au décodeur, une nouvelle méthode de fusion de plusieurs trames SI a été présentée, basée sur les algorithmes génétiques.

Chapter 1

Introduction

Traditional video coding techniques suffer from increased encoding complexity, mainly due to motion estimation performed at the encoder side. On the other hand, only motion compensation is required at the decoder, which results in simple decoders. This configuration (i.e. complex encoders and simple decoders) is suitable for applications where video encoding is performed once in a base station with sufficient resources, and decoded several times in multiple receivers. Low-complexity decoding is desired in this case in order to allow for the design and implementation of low-cost receivers for the end users.

During the last decade, several new applications gave rise to an increasing demand for simple video encoders. Consider for example video-enabled mobile phones: market requirements impose the design of small-sized devices characterized by limited memory resources and low power consumption. Wireless video sensor networks constitute another example, where the resources at the different sensors are scarce. In such applications, decoders are located in base stations with sufficient resources. Therefore, increased decoding complexity can be tolerated in this case.

With the aim of reducing the video encoding complexity, the research community introduced distributed video coding (DVC) in 2002. Based on information-theoretic studies that date back to the 1970's, DVC moves most of the computation burden from the encoder to the decoder side. However, to our knowledge, no practical codec has been put into market up to this date, since DVC still suffers from several problems.

In our study, we target two major problems in DVC: the high dependency on feedback information that is transmitted from the decoder back to the encoder, and the quality of the side information that greatly affects the system performance. Additionally, while most studies consider DVC as a tool only for video compression and ignore the effect of noisy transmissions, we extend the scope of DVC to the case of joint source-channel coding, where channel impairments are also taken into account.

We begin, in Chapter 2, by comparing the performance of binary and non-binary turbo-codes. Mainly designed for channel coding applications, turbo-codes have proven efficient performance when used as a tool for distributed source coding. Therefore, we study the performance of non binary turbo-codes when used for error protection and for distributed source compression, independently. The study presented in this chapter allows for the selection of a suitable code that will be used later in a distributed video coding system.

In Chapter 3, we consider the use of turbo-codes for jointly compressing a message signal and protecting the compressed stream from transmission errors. Therefore, we extend the distributed source coding system studied in the previous chapter for the case where the output data is transmitted over a noisy channel. Additionally, we consider joint source-channel coding from a networking point of view, where it can be seen as a cross-layer optimization approach.

After studying turbo-codes and their applications in joint source-channel coding, we present, in Chapter 4, the functional blocks of a pixel-domain distributed video codec, and then analytically estimate the system's lower compression bound for the case of error-free transmission, as well as for the case of a transmission over error-prone channels. We show how theoretical limits can be used for estimating the amount of data to be transmitted for successfully decoding the received information, without the excessive need for a feedback channel.

Chapter 5 presents two different applications of feedback channel suppression in DVC. Our study presented in the previous chapters is first applied in the context of a multiuser system, where several users transmit different video streams to a base station. A dynamic rate allocation algorithm is developed in order to manage channel resources and adapt to the random variations of the transmission channels. The algorithm is then enhanced using an adaptive quantization technique and a frame dropping mechanism. Up to this point, in our study, a Group of Pictures (GOP) consists of two frames only. Based on our rate estimation technique with feedback channel suppression, we propose novel algorithms for dynamically varying the GOP size in DVC.

In Chapter 6, we seek to improve the side information generated at the decoder. Since different side information can be obtained at the decoder using several interpolation techniques, the decoder can take advantage of this diversity by selecting the best among the different versions of the side information, for every region within a frame. Therefore, a

simple frame fusion approach that combines several interpolated frames is first presented, and a novel technique based on genetic algorithms is then developed.

Finally, conclusions are drawn in Chapter 7, with a discussion of our future visions for the continuity of this study.

Chapter 2

Binary and Non-Binary Turbo-Codes: Decoding Algorithms and Performance Evaluation

2.1 Introduction

In this chapter, we present turbo-codes and their applications in channel and source coding. We start first by briefly introducing source and channel coding. Convolutional codes are presented next, since they constitute a basic element in turbo-codes. Binary and non-binary turbo-codes are then presented, along with their decoding algorithms, in the context of an Additive White Gaussian Noise (AWGN) channel. Mainly designed for channel coding, turbo-codes can be used as an efficient source coding tool. Therefore, we detail afterwards the use of turbo-codes in the context of a source coding application. Finally, simulation results are discussed and a performance comparison between binary and non-binary turbo-codes is presented. This study allows for the selection of a suitable code that will be later used in a distributed video coding system.

2.2 Principles of Source and Channel Coding

Coding of data is essential in any communication system. Whether the data is to be stored on a storage media or transmitted over a channel, reducing the amount of data results in reduced storage and/or transmission costs. Source coding consists of removing redundant information from a given message in order to reduce the amount of significant data to be treated (stored or transmitted), therefore resulting in a compressed signal. However, a compressed signal is more sensitive to transmission channel impairments and to disk read/write operations than a raw signal. The need for error correcting codes thus arises. Channel coding consists of

concatenating some redundant bits to a given signal in order to permit error-correction whenever an error occurs [SAL06, GAL68].

Source coding can be customized for different types of data to yield more efficient compression. For example, in an audio signal, temporal redundancy between different samples is considered, whereas in a video signal, considering both spatial and temporal redundancies by the source coding algorithm enables a better compression. Data compression algorithms fall in two main categories: lossless and lossy. The former is more suitable for applications where the decoded data needs to be an exact match of the original source, such as in file transfer applications or medical file storage, whereas the latter is more suitable for applications where some loss of fidelity is acceptable, such as in speech, image, and video compression. Therefore, in a lossy compression algorithm, there is a tradeoff between the degree of compression and the amount of induced distortion. For this reason, the performance of a lossy compression algorithm is generally evaluated by its rate-distortion (RD) performance [HAY01-1], which shows the amount of distortion obtained for different levels of compression.

Channel codes are error correction codes, mainly characterized by the additional number of bits at the encoder's output given an input signal, and the number of errors they are able to correct for a given number of additional bits. Among the best performing codes these days are the turbo-codes [BER93, HAG96, BER03]. Mainly, a turbo-coder consists of a (parallel or serial) concatenation of two (or more) convolutional encoders, which will be presented in the next section.

2.3 Convolutional Codes

Convolutional codes [BAU07] introduce redundant information to the initial message signal through the use of linear shift registers and modulo-2 adders, as shown in Figure 2.1. The code rate R_c is defined as the ratio of the number k of input bits over the number n of output bits: $R_c = k / n$, and the code memory m is defined as the number of flip-flops used during the encoding process. The constraint length of a code is defined as $m+1$.

There are several types of convolutional encoders. They can be recursive or non-recursive, systematic or non-systematic, binary or non-binary. When the input to an encoder, at a given time instant, consists of a single bit, the code is said to be binary, whereas a non-binary code

accepts *M*-bit symbols at a time. In recursive codes, a feedback link exists between the output and the input of the shift-register (Figure 2.1). In systematic codes, *k* bits in the output *n*-bit word are an exact replica of the *k* input bits. In other words, the output of a systematic convolutional encoder consists of *n*-*k* redundant bits, known as parity bits, appended to the *k* input bits. Recursive systematic convolutional (RSC) encoders are known to offer a better performance compared to non-recursive and/or non-systematic codes. For this reason, we only consider the RSC family of codes in our study.

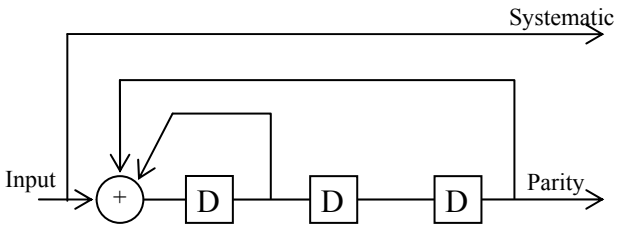


Figure 2.1 Recursive systematic convolutional encoder of rate ½.

In addition to their high performance in terms of error correction capability, convolutional encoders offer the capability of varying the code rate without the need for a different encoder/decoder implementation. This can be performed by puncturing the code, which consists of eliminating a subset of the parity bits (and possibly systematic bits), thus reducing the amount of redundant information, at the output of the encoder [BER03].

2.4 Turbo-Codes

2.4.1 Introduction

The first turbo-code was introduced by Berrou et Al. [BER93] in 1993. It consists of a parallel concatenation of two convolutional encoders separated by an interleaver, as shown in Figure 2.2. The role of the interleaver is to scramble the input sequence so that the output of the second constituent encoder becomes different from the output of the first encoder, thus reducing the correlation between the two encoders' outputs. This allows for a better decoding performance when information is exchanged between the two corresponding decoders.

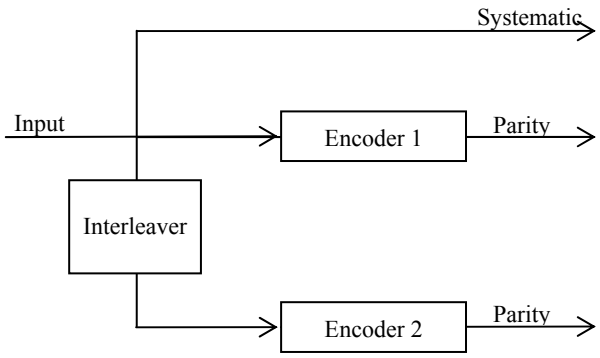


Figure 2.2 Generalized parallel turbo-encoder structure.

2.4.2 Practical examples of turbo-encoders

A turbo-code can be binary or non-binary, depending on the nature of its constituent encoders. In our study, we consider binary, duo-binary, and quadri-binary turbo-codes. As practical cases for encoding structures for each encoder type, we consider the binary code defined in the Universal Mobile Telecommunications System (UMTS) standard [3GP00], the duo-binary code (based on a rate 1/2 convolutional encoder) defined in Digital Video Broadcasting (DVB) [DOU00], and the most performing quadri-binary code (based on a rate 4/5 constituent encoder) determined in [DIV95]. The constituent encoder structure for each of these codes is shown in Figure 2.3.

As for the interleaving, in case of binary codes it is performed by a pseudo-random interchange of bit positions within the input sequence [SKL97]. For non-binary codes, interleaving is realized in two steps [DOU00]: first, *M*-bit symbols pseudo-randomly exchange positions (inter-symbol interleaving). Then, the bits within the same symbol are read in a scrambled order (intra-symbol interleaving).

2.4.3 Turbo-decoding algorithms

Turbo-decoding is based on the following criterion [BER03]: "**When several probabilistic machines work together on the estimation of a common set of symbols, all the machines have to give the same decision, with the same probability, about each symbol, as a single (global) decoder would.**" Therefore, a turbo-decoder consists of two Soft-Input Soft-Output (SISO) decoders, separated by an interleaver identical to the one at the encoder's side, and exchanging probabilistic messages, as shown in Figure 2.4.

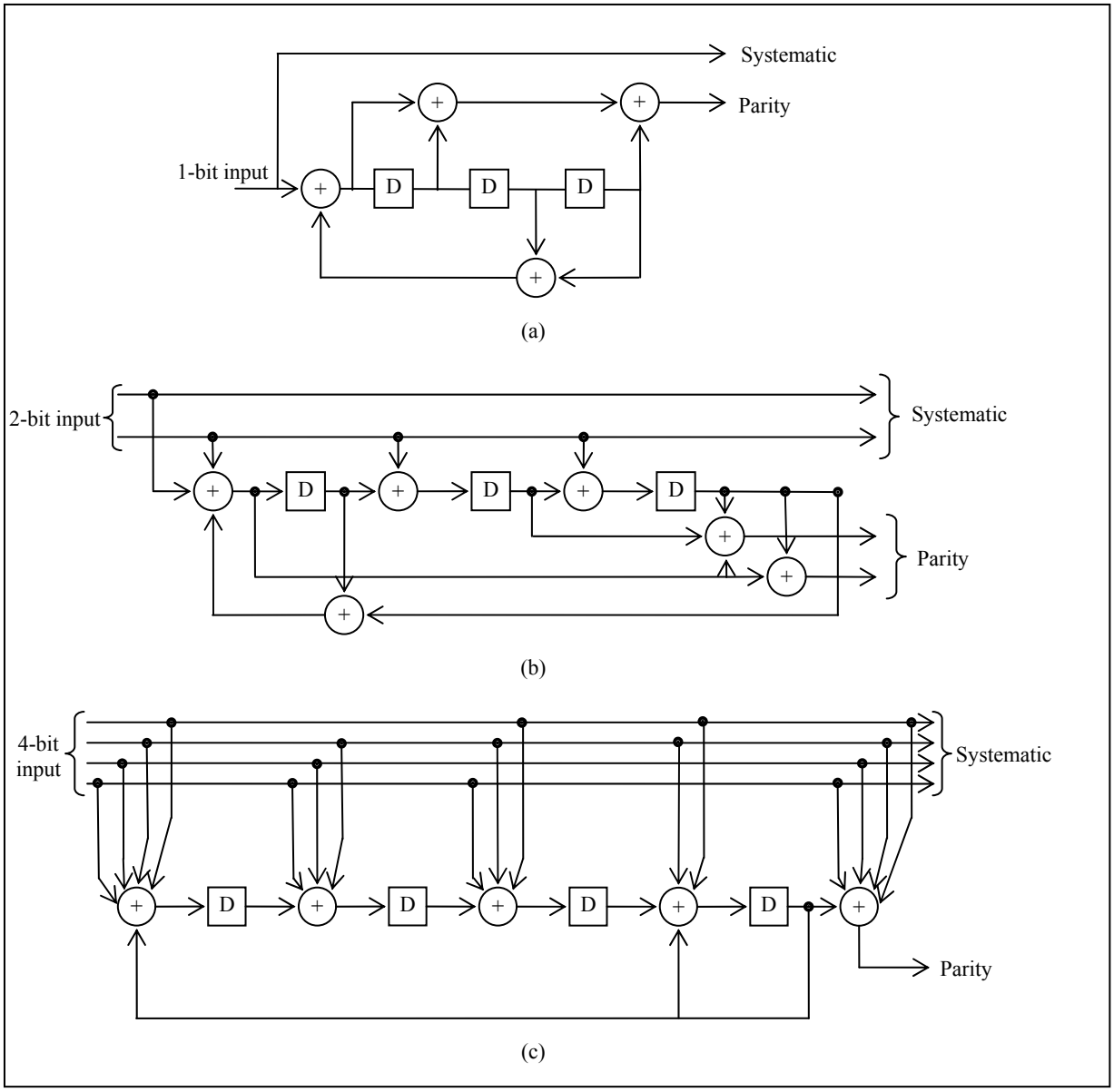


Figure 2.3 Binary (a), duo-binary (b) and quadri-binary (c) convolutional encoders.

L_i is the "a priori information", obtained by de-interleaving the "extrinsic information" L_i^a provided by the preceding decoder.

Optimal decoding can be performed using the maximum a posteriori (MAP) algorithm [BER93]. However, due to its very high complexity, MAP is generally substituted by sub-optimal algorithms such as the Max-Log-MAP or the Soft Output Viterbi algorithms (SOVA) [ROB97]. In this study, we consider the Max-Log-MAP algorithm presented in [GAO02] for a 16-state (code memory m = 4) triple-binary turbo-code, and we generalize it for M-ary codes.

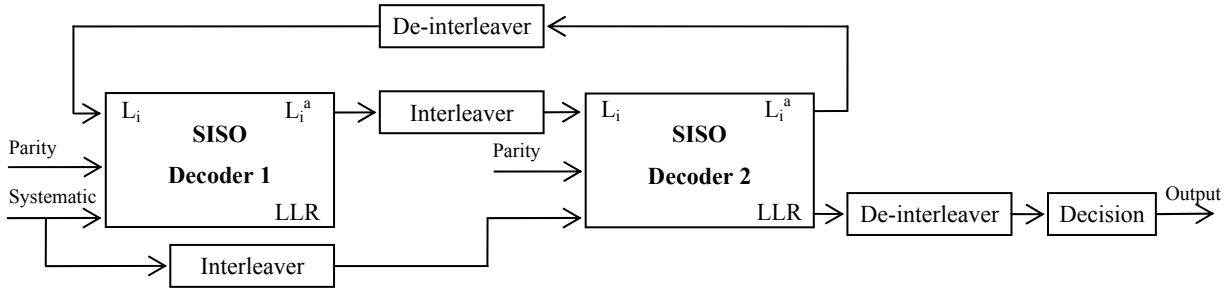


Figure 2.4 General structure of a turbo-decoder.

2.4.4 Generalized Max-Log-MAP algorithm for symbol-based turbo-decoding

Let m represent the code memory of each constituent convolutional encoder. The trellis [HAY01-2] of the convolutional code has therefore $S = 2^m$ states. Assuming the encoder accepts M bits at a time, each node in the trellis is connected to $L = 2^M$ transition branches to L possible states in the subsequent stage, as shown in Figure 2.5.

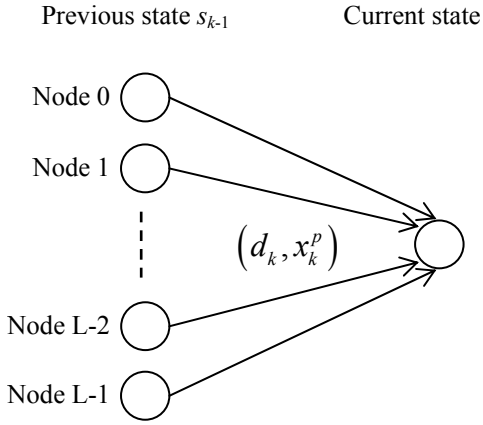


Figure 2.5 Code trellis (d_k: input symbol, x_k^p : parity output).

Let s_k be the encoder state at time instant k in the trellis, and d_k the input symbol associated with the transition from time k-1 to time k, $d_k \in \{0, 1, .., L\text{-}1\}$.

The log likelihood ratio (LLR) of a symbol $d_k = i$ is defined as:

$$LLR_i(d_k) = \ln \frac{P(d_k = i \mid observation)}{P(d_k = 0 \mid observation)} = \ln \frac{\sum_{\substack{(s_{k-1}, s_k) \\ d_k = i}} P(s_{k-1}, s_k, y_k)}{\sum_{\substack{(s_{k-1}, s_k) \\ d_k = 0}} P(s_{k-1}, s_k, y_k)} \tag{2.1}$$

where y_k is the received symbol (observation) at time k, which consists of parity (y_k^p) and systematic (y_k^s) information. The summation in both the numerator and denominator of Eq. (2.1) is performed over all possible transitions from state s_{k-1} to state s_k. By assuming a memoryless channel, the joint probability $P(s_{k-1}, s_k, y_k)$ can be written as the product of three independent probabilities:

$$\begin{aligned} P(s_{k-1}, s_k, y_k) &= P(s_{k-1}, y_{j<k}) \cdot P(s_k, y_k \mid s_{k-1}) \cdot (y_{j>k} \mid s_k) \\ &= \alpha_{k-1}(s_{k-1}) \cdot \gamma_k^i(s_{k-1}, s_k) \cdot \beta_k(s_k). \end{aligned} \tag{2.2}$$

In the above equation, $y_{j<k}$ represents the sequence of the received symbols y_j from the beginning of the trellis till the time instant k-1, and $y_{j>k}$ the sequence received from time k+1 till the end.

In the MAP algorithm, the forward and backward state metrics can be calculated recursively [GAO02] as:

$$\begin{cases} \alpha_k(s_k) = \sum_{s_{k-1}} \sum_{i=0}^{L-1} \gamma_k^i(s_{k-1}, s_k) \alpha_{k-1}(s_{k-1}), \\ \beta_{k-1}(s_{k-1}) = \sum_{s_k} \sum_{i=0}^{L-1} \gamma_k^i(s_{k-1}, s_k) \beta_k(s_k). \end{cases} \tag{2.3}$$

Whenever a transition occurs from state s_{k-1} to state s_k, the branch transition metric is obtained by:

$$\gamma_k^i(s_{k-1}, s_k) = P(y_k \mid d_k) P(d_k). \tag{2.4}$$

On the other hand, given a set x of symbols x_j, $j \in \{1,2, \ldots, N\}$, the logarithm of the sum of all the elements' exponents in the set can be approximated [GAO02] as:

$$\ln \sum_{j=1}^{N} e^{x_j} = \max(x) + \ln \sum_{j=1}^{N} e^{x_j - \max(x)} \approx \max(x). \tag{2.5}$$

The Max-Log-MAP algorithm consists of taking the logarithm of the metrics derived in equations (2.3) and (2.4), then applying the above estimation. Therefore, the branch metric is first expressed as:

$$\overline{\gamma}_k^i(s_{k-1}, s_k) = \ln \gamma_k^i(s_{k-1}, s_k) = \ln P(y_k \mid d_k) + \ln P(d_k). \tag{2.6}$$

The forward and backward metrics are then derived by:

$$\begin{cases} \bar{\alpha}_k(s_k) = \ln \alpha_k(s_k) \approx \max\limits_{s_{k-1}, s_k, i} \left[\bar{\gamma}_k^i(s_{k-1}, s_k) + \bar{\alpha}_{k-1}(s_{k-1}) \right], \\ \bar{\beta}_{k-1}(s_{k-1}) = \ln \beta_{k-1}(s_{k-1}) \approx \max\limits_{s_k, s_{k-1}, i} \left[\bar{\gamma}_k^i(s_{k-1}, s_k) + \bar{\beta}_k(s_k) \right]. \end{cases} \tag{2.7}$$

As a result, the log likelihood ratio in Eq. (2.1) can be estimated as:

$$\begin{aligned} LLR_i(d_k) \approx & \max_{(s_{k-1}, s_k)} \left[\bar{\gamma}_k^i(s_{k-1}, s_k) + \bar{\alpha}_{k-1}(s_{k-1}) + \bar{\beta}_k(s_k) \right] \\ & - \max_{(s_{k-1}, s_k)} \left[\bar{\gamma}_k^0(s_{k-1}, s_k) + \bar{\alpha}_{k-1}(s_{k-1}) + \bar{\beta}_k(s_k) \right]. \end{aligned} \tag{2.8}$$

In each decoding iteration, L-1 LLRs have to be computed for each symbol d_k using Eq. (2.8).

On the other hand, the a priori information [ROB97, GAO02] on the symbol d_k is given by:

$$L_i(d_k) = \ln \frac{P(d_k = i)}{P(d_k = 0)}. \tag{2.9}$$

Therefore:

$$P(d_k = i) = P(d_k = 0) e^{L_i(d_k)}. \tag{2.10}$$

Taking the sum of Eq. (2.10) over all possible values of i except for $i = 0$ yields:

$$\begin{aligned} \sum_{i=1}^{L-1} P(d_k = i) &= \sum_{i=1}^{L-1} P(d_k = 0) e^{L_i(d_k)} \\ &= P(d_k = 0) \sum_{i=1}^{L-1} e^{L_i(d_k)} = 1 - P(d_k = 0). \end{aligned} \tag{2.11}$$

By dividing both sides of the above equation by $P(d_k = 0)$ we obtain:

$$\sum_{i=1}^{L-1} e^{L_i(d_k)} = \frac{1}{P(d_k = 0)} - 1. \tag{2.12}$$

As a result:

$$P(d_k = 0) = \frac{1}{1 + \sum\limits_{i=1}^{L-1} e^{L_i(d_k)}}. \tag{2.13}$$

By substituting $P(d_k = 0)$ in Eq. (2.10) with the expression in Eq. (2.13), we obtain:

$$P\left(d_k = i\right) = \frac{e^{L_i(d_k)}}{1 + \sum_{i=1}^{L-1} e^{L_i(d_k)}}. \tag{2.14}$$

The logarithm of symbol probabilities for the next decoder can thus be estimated using Eq. (2.5) as:

$$\begin{cases} \ln P\left(d_k = 0\right) = -\max\left[0, L_i\left(d_k\right)\right], \\ \ln P\left(d_k = i\right) = L_i\left(d_k\right) - \max\left[0, L_i\left(d_k\right)\right]. \end{cases} \tag{2.15}$$

As for the conditional probability $P(y_k \mid d_k)$, it varies depending on the nature of the transmission channel. Therefore, it will be derived later for every particular application case for turbo-codes.

In [GAO02], the extrinsic information on a symbol d_k to be transmitted to the next decoder was estimated by its LLR. Whether this was done intentionally to simplify the decoder's implementation, or by mistake, we noticed that this estimation results in some performance loss. While this loss is not of great importance in this chapter, since our aim is only a performance comparison between binary and non-binary codes, we noticed that it significantly degrades a system's performance when the estimation is used with video applications, due to the high sensitivity of video streams to coding errors. Therefore, when turbo-codes will be used for video coding (in the next chapters), the estimation used by Gao will be corrected. In this case, the extrinsic information used, after interleaving, as a priori information in the next decoder will be calculated by:

$$L_i^a\left(d_k\right) = LLR_i\left(d_k\right) - L_i\left(d_k\right) - L_i^c\left(d_k\right), \tag{2.16}$$

where $L_i^c\left(d_k\right)$ is a measure of the channel reliability [SKL97] defined as:

$$L_i^c\left(d_k\right) = \ln \frac{P\left(y_k^s \mid d_k = i\right)}{P\left(y_k^s \mid d_k = 0\right)}. \tag{2.17}$$

In fact, as it can be seen from Eq. (2.16), the extrinsic information transferred from one decoder to the next is the LLR from which the channel reliability and the a priori information

have been removed. In other words, only the updated information obtained by the current decoder (using the Max-Log-MAP algorithm) is transferred to the next one.

At the last iteration, d_k is decoded using the L-1 LLRs, estimated for each symbol d_k using Eq. (2.8), by taking:

$$\tilde{d}_k = \begin{cases} \arg\max_i LLR_i(d_k) & \text{if } \max LLR_i(d_k) > 0, \\ 0 & \text{otherwise.} \end{cases} \tag{2.18}$$

2.5 Distributed Source Coding Using Turbo-Codes

Distributed source coding (DSC) consists of coding several, statistically dependent sources, by exploiting the correlation statistics between them [AAR02-1]. It is known from information theory that, given two statistically dependent sources X and Y, each source can be independently compressed at a rate very close to its entropy limit H(X) and H(Y) respectively. However, by exploiting the correlation between the two sources, X and Y can be jointly compressed to the joint entropy H(X,Y). This results in a stronger compression, since: $H(X,Y) \leq H(X) + H(Y)$. The idea behind DSC goes back to the 1970s, when Slepian and Wolf proved in [SLE73] that, if Y is compressed to its entropy limit H(Y), X can be transmitted at a rate very close to the conditional entropy H(X|Y), provided that Y is perfectly recovered at the receiver as side information for decoding X. Since H(X,Y) = H(Y) + H(X|Y), X and Y can be independently encoded and jointly decoded without any loss in the compression efficiency, compared to the case where both sources are jointly encoded and decoded. The application of this concept to lossy source coding is known as the Wyner-Ziv coding [WYN76].

In order for the sources to be jointly encoded and/or decoded, additional intelligence is required for the codec to be able to achieve a compression rate close to the joint entropy, which results in a significant increase in complexity. In applications where the encoder has very limited resources (e.g. power and memory), such as the sensors in a wireless sensor network (WSN), increasing the encoder's complexity is barely possible. Therefore, in such applications, the computation burden should be shifted to the decoder, which is usually located in a base station with sufficient (theoretically infinite) resources.

In the current work, we limit our study to the case where only two correlated sources X and Y exist, for simplicity. We assume that the source Y is encoded and decoded using any conventional codec that achieves a compression rate close to H(Y), and that Y is perfectly recovered at the receiver. Since X and Y are correlated, Y can be considered as a noisy version of X, and a channel code can be used to recover X, given Y. Therefore, by feeding the source X to a turbo-coder, compression can be achieved by eliminating the systematic data at the encoder's output, provided that the number of output parity bits is less or equal to the amount of information bits. The compression rate can then be varied by puncturing the parity bits appropriately. At the decoder side, Y can be used as an initial estimate of the missing systematic data, and the received parity bits can be used to correct the estimation errors.

From a complexity point of view, turbo-encoding is a very simple process which consists of convolution and interleaving. The computational burden mainly resides in the iterative turbo-decoding algorithms. As a result, turbo-codes can be used for the compression of distributed sources, especially in applications where the encoder's complexity is a main concern. In the sequel, we will study the performance of binary and non-binary turbo-codes to determine the code that converges the most toward the Slepian-Wolf bound H(X|Y).

2.6 Simulation Models and Results

In this paragraph, we compare the performances of binary and non-binary turbo-codes in two different contexts: first, in a channel coding application with an AWGN channel, and then in a distributed source coding application, where the correlation channel is modeled by a Binary Symmetric Channel (BSC). The codes used in our simulations are based on the constituent encoders shown in Figure 2.3.

2.6.1 System model for a channel coding application

Consider first the case where turbo-codes are used as channel codes. In our simulation model, a random binary sequence is first generated and fed to a turbo-encoder input. The output of the encoder is modulated using a binary phase shift keying (BPSK) modulator. The modulated output is then transmitted over an AWGN channel. At the receiver side, the demodulator output is fed to the turbo-decoder and the decoded output is then compared to the initial sequence in order to determine the bit-error-rate (BER). The BER is determined for

different code rates and for different values of the information bit energy to noise density (E_b/N_0) ratio.

We assume that the noise is zero-mean with a variance σ^2. Let $N_0/2$ represent the noise power spectral density, T_s the symbol duration, and $B = 1/T_s$ the channel bandwidth. The noise power at the output of the pre-detection filter, in the receiver, is expressed as:

$$\sigma^2 = N_0 B = \frac{N_0}{T_s}. \tag{2.19}$$

For a code of rate R_c, the average symbol energy E_s can be determined as a function of the average information bit energy E_b by:

$$E_s = R_c E_b \tag{2.20}$$

The average symbol power can thus be obtained by:

$$P_s = R_c E_b / T_s \tag{2.21}$$

Therefore, the signal to noise ratio (SNR) can be determined by:

$$SNR = \frac{R_c E_b / T_s}{\sigma^2} = \frac{R_c E_b / T_s}{N_0 / T_s} = R_c \frac{E_b}{N_0} \tag{2.22}$$

On the other hand, for a BPSK modulation, the signal power at the output of the pre-detection filter is $P_s = A^2/2$, where A is the symbol amplitude after baseband BPSK mapping (considering equiprobable levels $\pm A$). If we consider $A = 1$, the SNR is then given by:

$$SNR = \frac{1}{2\sigma^2} \tag{2.23}$$

By combining Eq. (2.22) and Eq. (2.23), we obtain:

$$\sigma^2 = \frac{1}{2R_c \frac{E_b}{N_0}} \tag{2.24}$$

Eq. (2.24) expresses the noise variance as a function of the channel parameter E_b/N_0.

As stated earlier, the conditional probability $P(y_k \mid d_k)$ in the turbo-decoder varies depending on the nature of the channel. Therefore, we need to determine $P(y_k \mid d_k)$ for the AWGN channel. Since the channel is memoryless,

$$P\left(y_k \mid d_k\right) = P\left(y_k^s \mid x_k^s\right) P\left(y_k^p \mid x_k^p\right), \tag{2.25}$$

where y_k^s and y_k^p represent the received sets of systematic and parity bits at time instant k at the input of a SISO decoder, and x_k^s and x_k^p represent the transmitted sets of systematic and parity bits at time instant k, respectively.

Since the sets y_k^s and x_k^s consist of M bits each and since the channel is Gaussian:

$$P\left(y_k^s \mid x_k^s\right) = \prod_{i=1}^{M} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{\left(y_{ik}^s - x_{ik}^s\right)^2}{2\sigma^2}\right). \tag{2.26}$$

Similarly,

$$P\left(y_k^p \mid x_k^p\right) = \prod_{i=1}^{M_p} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{\left(y_{ik}^p - x_{ik}^p\right)^2}{2\sigma^2}\right), \tag{2.27}$$

where M_p represents the number of parity bits at the output of each constituent encoder (M_p *depends on the encoder code rate before puncturing)*.

By substituting equations (2.26) and (2.27) in (2.25) after some manipulations, and knowing that $x_{ik}^s = \pm 1$ and $x_{ik}^p = \pm 1$ (because of BPSK modulation), we obtain:

$$P\left(y_k \mid d_k\right) = B_k \exp\left(\frac{\sum_{i=1}^{M} x_{ik}^s y_{ik}^s + \sum_{i=1}^{M_p} x_{ik}^p y_{ik}^p}{\sigma^2}\right), \tag{2.28}$$

where B_k is a constant term independent of x_k^s and x_k^p:

$$B_k = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{M + M_p + \sum_{i=1}^{M} y_{ik}^s + \sum_{i=1}^{M_p} y_{ik}^p}{2\sigma^2}\right). \tag{2.29}$$

Different puncturing patterns have been used in our simulations in order to vary the code rate R_c within the set of values {1/3, 1/2, 2/3, 3/4, 4/5, 6/7}. We denote $P_c(r)$ the pattern used for a code rate $R_c=r$. Figure 2.6 shows an example of the patterns used with binary and non-binary codes, for a code rate R_c = 4/5. In each puncturing matrix, the first M lines represent the systematic bits, whereas the following M_p lines represent the parity bits. A '1' indicates a transmitted bit whereas a '0' indicates a punctured bit. The puncturing patterns were chosen such that the punctured bits are as equally spaced (in time) as possible.

$$P_c(4/5)_{binary} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

$$P_c(4/5)_{duo-binary} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} \quad P_c(4/5)_{quadri-binary} = \begin{pmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}$$

Figure 2.6 Sample puncturing patterns for a code rate R_c = 4/5.

2.6.2 Simulation results for an AWGN channel

Figure 2.7 and Figure 2.8 show the BER as a function of E_b/N_0, obtained for different code rates with duo-binary and binary codes, respectively. For fair comparison between binary and non-binary codes, we fix the interleaver length to 648 symbols, which corresponds to a binary sequence of length 648xM bits in case of a M-ary code. We notice that for a code rate of 1/3 (without puncturing), both codes have similar performance. When the code rate increases, a better convergence is obtained with duo-binary codes. Duo-binary codes are in fact more resistant to puncturing. Consider, for example, both figures at a BER of 10^{-4}. By moving from a code rate of 1/3 to 2/3, there is a performance loss of 1.1 dB with binary codes, whereas only a loss of 0.8 dB is noticed with duo-binary codes.

In Figure 2.9, we show BER curves for binary, duo-binary and quadri-binary codes for code rates of 2/3 and 4/5. In the case where the turbo-encoders are punctured to achieve a coding rate of 4/5, we can observe a clear gain of 0.4 dB and 0.9 dB for the duo-binary and quadri-

binary encoders, respectively, towards the binary encoder, at a BER around 10^{-5}. These gains become respectively 0.5 dB and 0.9 dB when the coding rate increases to 2/3.

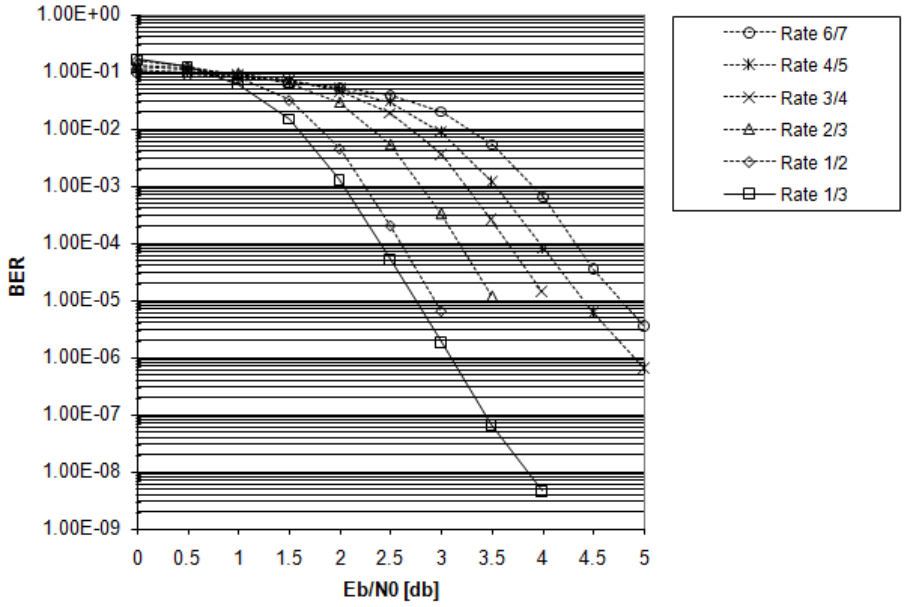


Figure 2.7 BER obtained with duo-binary turbo-codes used for error correction at different coding rates over a Gaussian channel.

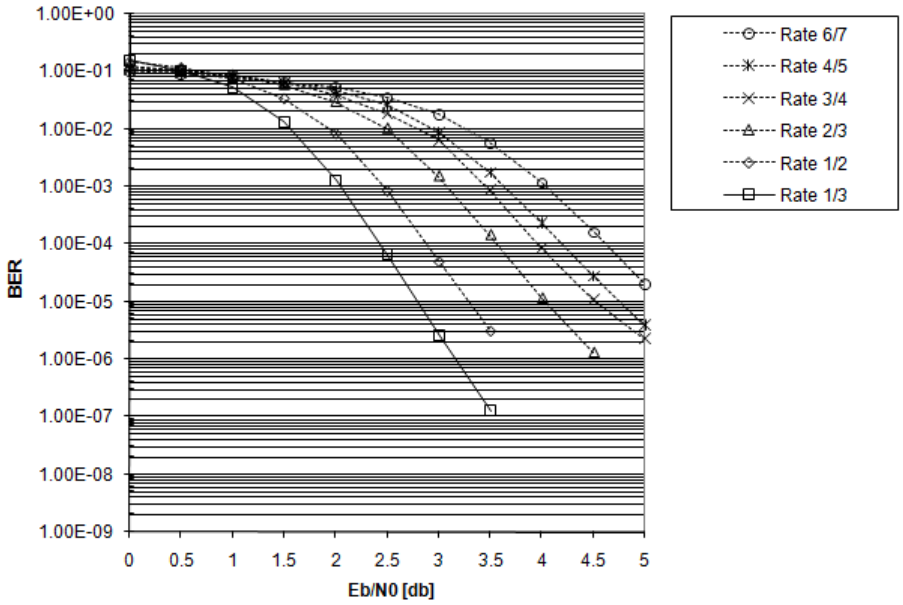


Figure 2.8 BER obtained with binary turbo-codes used for error correction at different coding rates over a Gaussian channel.

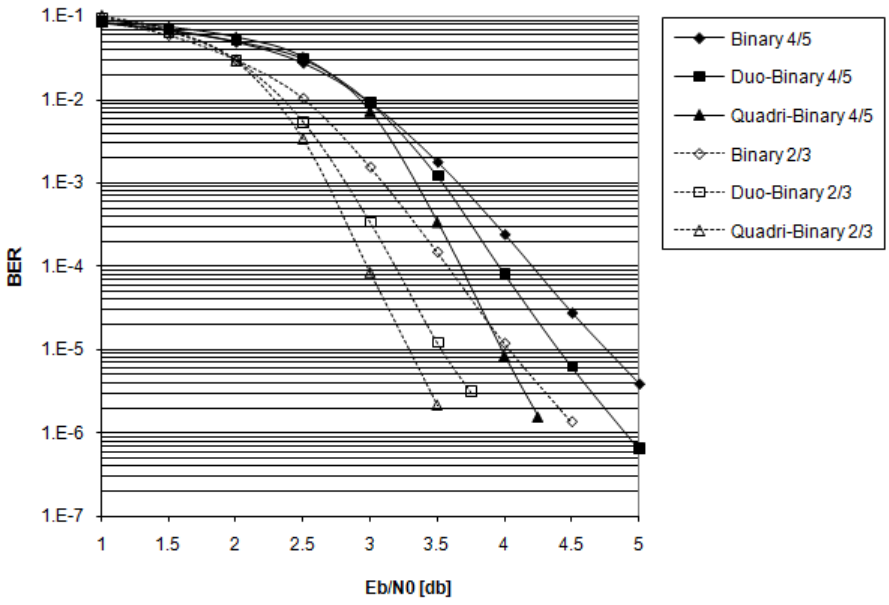


Figure 2.9 BER obtained with binary, duo-binary and quadri-binary codes used for error correction over a Gaussian channel at code rates of 2/3 and 4/5.

Therefore, we conclude from the above analysis that non-binary codes offer a better performance compared to binary codes, when used for channel coding, especially when the code is highly punctured. They approach Shannon's bound (E_b/N_0 = -1.6 dB) faster and they are less sensitive to puncturing.

2.6.3 System model for a distributed source coding application

In our simulated model for distributed compression, a random binary sequence is first generated and turbo-coded. The systematic information is supposed to experience a binary symmetric channel (BSC) with a transition probability p, thus resulting in a statistically dependant second source, considered as side information perfectly recovered at the receiver. As for the parity bits, they are assumed to be received error-free. BPSK is assumed for symbol modulation.

In order to model the correlation between the sources by a BSC with crossover probability p, two different methods are proposed. For an input block of length N, the first method consists of imposing $\lfloor pN \rfloor$ bit inversions at random locations within the block, where $\lfloor x \rfloor$ represents the greatest integer less than or equal to x. This method is summarized as follows:

a. Generate a random variable 'u' uniformly distributed between 0 and 1.

b. Let $i = \lfloor uN \rfloor$.

c. Invert the bit b_i at position i.

d. Repeat the steps (a), (b), and (c), until the number of inverted bits reaches $\lfloor pN \rfloor$.

Even though this method yields $\lfloor pN \rfloor$ bit inversions per block, in realistic situations, the number of inverted bits at the output of a BSC is not necessarily $\lfloor pN \rfloor$ for every block. It is only when the number of blocks is very large (theoretically infinite) that the effective channel transition probability approaches p. Therefore, we have opted for a more realistic modeling of the BSC, where every bit in the block is inverted with a probability p. The algorithm operates as follows: For every position i within the block ($i \in \{0, 1, \ldots, N\text{-}1\}$):

a. Generate a random variable 'u' uniformly distributed between 0 and 1.

b. If $u < p$, invert the bit b_i at position i.

Let $f_U(u)$ represent the probability distribution function of u. Since $f_U(u) = 1$ for $0 \leq u < 1$ and $0 \leq p \leq 1$, it can be verified that with this second method, the bit inversion probability is p:

$$P(b_i \text{ inverted}) = P(u < p) = \int_0^p f_U(u)\,du = p\,. \tag{2.30}$$

The next step consists on deriving a proper formulation for the conditional probability $P(y_k \mid d_k)$ for the case of a BSC channel.

Let us first consider the case of a binary turbo-code having one parity bit at the output of each constituent encoder. Let $L_c = \ln\left(\frac{1-p}{p}\right)$ denote the channel reliability [HAG96]. In the absence of puncturing, and since the parity bits are assumed to be perfectly received, the conditional probability can be written as:

$$P(y_k \mid d_k) = \begin{cases} 0, \text{ if } y_k^p \neq x_k^p \\ \text{otherwise, } \begin{cases} p, & \text{if } y_k^s \neq x_k^s \\ 1-p, & \text{if } y_k^s = x_k^s. \end{cases} \end{cases} \tag{2.31}$$

Eq. (2.31) can be equivalently expressed as:

$$P\left(y_k \mid d_k\right)=\begin{cases}0, \text{ if } y_k^p \neq x_k^p \\ \text{otherwise,} \begin{cases}(1-p)\exp\left(-L_c\right), \text{ if } y_k^s \neq x_k^s \\ p\exp\left(L_c\right), \quad \text{if } y_k^s = x_k^s .\end{cases}\end{cases} \tag{2.32}$$

In order to simplify expressions, especially for the ulterior generalization to the case of punctured M-ary codes with M_p parity bits, let us write $P(y_k \mid d_k)$ in the following form:

$$P\left(y_k \mid d_k\right)=\alpha\left[\beta(1-p)+\gamma p\right]\exp\left[\delta L_c\right]. \tag{2.33}$$

We must determine α, β, γ and δ so that Eq. (2.33) becomes equivalent to Eq. (2.32).

If $y_k^p \neq x_k^p$, we must have $\alpha = 0$ so that $P(y_k \mid d_k) = 0$. If $y_k^p = x_k^p$, let $\alpha = 1$. Therefore, since $x_k^p = \pm 1$ and $y_k^p = \pm 1$, α can be expressed as:

$$\alpha=\frac{1+x_k^p y_k^p}{2}. \tag{2.34}$$

For $\alpha = 1$ ($y_k^p = x_k^p$), we must have:

$$\begin{cases}\beta = 1, \ \gamma = 0 \text{ and } \delta = -1, \text{ if } y_k^s \neq x_k^s \\ \beta = 0, \ \gamma = 1 \text{ and } \delta = +1, \text{ if } y_k^s = x_k^s .\end{cases} \tag{2.35}$$

As a result,

$$\beta=\frac{1-x_k^s y_k^s}{2}, \gamma=\frac{1+x_k^s y_k^s}{2}, \text{and } \delta = x_k^s y_k^s . \tag{2.36}$$

By substituting Eq. (2.34) and Eq. (2.36) in Eq. (2.33), and developing the resulting equation, we obtain:

$$P\left(y_k \mid d_k\right)=\frac{1}{4}\left[1+x_k^p y_k^p\right]\left[1+(2p-1)x_k^s y_k^s\right]\exp\left[L_c x_k^s y_k^s\right] \tag{2.37}$$

Consider now the case where the code is punctured. At the receiver side, y_k^p takes not only the values ±1, but it can also take the zero value (in case it has been punctured). Therefore,

$$P(y_k \mid d_k) = \begin{cases} \frac{1}{4}\left[1 + x_k^p y_k^p\right]\left[1 + (2p-1) x_k^s y_k^s\right] \exp\left[L_c x_k^s y_k^s\right], & \text{if } y_k^p \neq 0 \\ \text{otherwise,} \begin{cases} p, & \text{if } y_k^s \neq x_k^s \\ 1-p, & \text{if } y_k^s = x_k^s. \end{cases} \end{cases} \tag{2.38}$$

To further simplify our metric, Eq. (2.38) can be written as:

$$P(y_k \mid d_k) = A\left\{\frac{1}{4}\left[1 + x_k^p y_k^p\right]\left[1 + (2p-1) x_k^s y_k^s\right] \exp\left[L_c x_k^s y_k^s\right]\right\}, \tag{2.39}$$

where A is to be determined such that Eq. (2.39) becomes identical to Eq. (2.38).

Obviously, if $y_k^p \neq 0$, then A=1. On the other hand, if $y_k^p = 0$, using that $e^{L_c} = \frac{1-p}{p}$, Eq. (2.39) reduces to:

$$\begin{aligned} P(y_k \mid d_k) &= A\left\{\frac{1}{4}\left[1 + (2p-1) x_k^s y_k^s\right] \exp\left[L_c x_k^s y_k^s\right]\right\} \\ &= \begin{cases} \frac{A}{2} p, & \text{if } y_k^s \neq x_k^s \\ \frac{A}{2}(1-p), & \text{if } y_k^s = x_k^s. \end{cases} \end{aligned} \tag{2.40}$$

In this case, we must have A=2 so that Eq. (2.40) verifies:

$$P(y_k \mid d_k) = \begin{cases} p, & \text{if } y_k^s \neq x_k^s \\ 1-p, & \text{if } y_k^s = x_k^s. \end{cases} \tag{2.41}$$

Therefore,

$$A = \begin{cases} 1, & \text{if } y_k^p \neq 0 \\ 2, & \text{if } y_k^p = 0. \end{cases} \tag{2.42}$$

It can be verified that (2.42) can be expressed as:

$$A = 2 - \left|y_k^p\right|. \tag{2.43}$$

By substituting (2.43) into (2.39) we obtain:

$$P(y_k \mid d_k) = \frac{1}{4}\left[2 - \left|y_k^p\right|\right]\left[1 + x_k^p y_k^p\right]\left[1 + (2p-1) x_k^s y_k^s\right] \exp\left[L_c x_k^s y_k^s\right]. \tag{2.44}$$

In the general case of M-ary codes with M_p parity bits at the output of each constituent encoder, the conditional probability can be generalized by the following equation:

$$P(y_k \mid d_k) = \frac{1}{2^{M+M_p}} \cdot \prod_{i=1}^{M_p} \left[2 - \left|y_{ik}^p\right|\right] \left[1 + x_{ik}^p y_{ik}^p\right] \cdot \prod_{j=1}^{M} \left[1 + (2p-1) x_{jk}^s y_{jk}^s\right] \cdot \exp\left[L_c \sum_{n=1}^{M} x_{nk}^s y_{nk}^s\right] \tag{2.45}$$

2.6.4 Simulation results for distributed source coding

Figure 2.10 presents the BER obtained for the compression ratios (4:1) and (9:1), with an interleaver size of 10000 symbols. The results are presented in terms of the conditional entropy H(X|Y), where X and Y represent the two statistically dependent sources. H(X|Y) is related to the channel crossover probability p by:

$$H(X \mid Y) = -p \log_2(p) - (1-p) \log_2(1-p). \tag{2.46}$$

We notice that the quadri-binary codec presents the best performance compared to the binary or duo-binary case. For example, when the compression ratio is 4:1, a BER of 10^{-4} is obtained with quadri-binary codes for an entropy value of 0.15, whereas a BER of approximately 10^{-3} is obtained with binary and duo-binary codes, for the same compression and entropy. In other words, for the above-mentioned values of BER and entropy, the number of errors obtained with binary and duo-binary codes is 10 times the number obtained with quadri-binary codes.

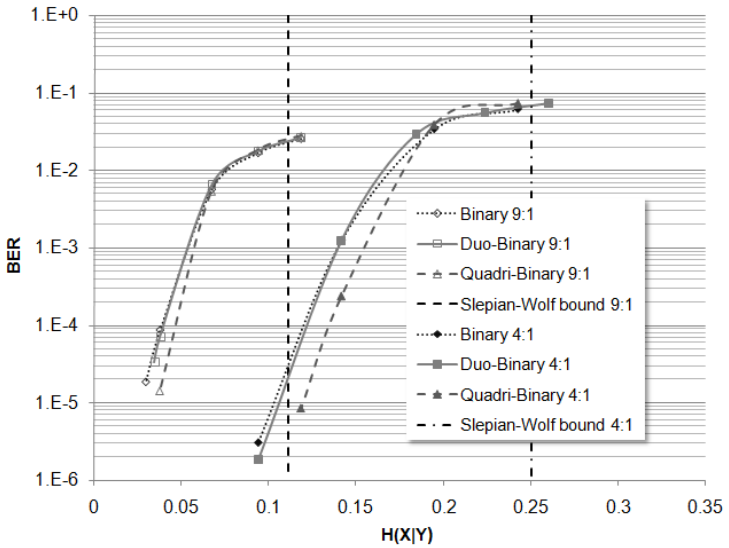


Figure 2.10 BER obtained with binary, duo-binary and quadri-binary codes used for the DSC of two sources with a BSC correlation model and compression ratios 9:1 and 4:1.

The gain obtained with non-binary codes increases (compared to binary codes) when the entropy decreases. Therefore, they offer a better convergence toward the Slepian-Wolf bound [AAR02-1], which can be clearly observed in Figure 2.10.

2.7 Complexity Analysis

In this section, we estimate the necessary number of operations (OPs) to achieve the decoding of one input symbol by the Max-Log-MAP algorithm. This will allow us to determine suitable performance-complexity tradeoffs for applications using turbo-codes.

In the generalized Max-Log-MAP algorithm, we have, for each state s_k, L possible states s_{k-1}. For each state s_k, L floating additions and (L-1) floating comparisons are performed in Eq. (2.7) during the computation of $\bar{\alpha}_k(s_k)$. Therefore, the computation of the forward metrics requires SL floating additions and $S(L\text{-}1)$ floating comparisons. Similarly, the computation of the backward metrics requires SL floating additions and $S(L\text{-}1)$ floating comparisons. As for the calculation of the (L-1) LLRs using Eq.(2.8), each one consists of ($4S$+1) floating additions and 2(S-1) floating comparisons (max operations). Hence, (L-1)($4S$+1) floating additions and 2(L-1)(S-1) floating comparisons are required for the computation of the LLRs.

On the other hand, the complexity of the branch metrics in Eq. (2.6) varies with the nature of the channel. First, the estimation of $\ln P(d_k)$ using Eq. (2.15) requires 1 floating addition and (L-1) floating comparisons. For the case of an AWGN channel, the calculation of $\ln P(y_k \mid d_k)$, using the logarithm of Eq. (2.28) for each state s_k and each possible transition from s_{k-1} to s_k, requires (M_p+M -1) floating additions and (M_p+M) floating multiplications. Therefore, the computation of the branch metrics in Eq. (2.6), for the case of an AWGN channel, requires $SL(M_p+M+1)$ floating additions, $SL(M_p+M)$ floating multiplications, and $SL(L\text{-}1)$ floating comparisons. In the case of a BSC, the calculation of $\ln P(y_k \mid d_k)$, using the logarithm of Eq. (2.45), necessitates ($4M_p$+$3M$-1) additions, (M_p+$3M$) multiplications, ($2M_p$+M) logarithms, and M_p absolute values. Therefore, the branch metrics calculation in BSC, using Eq. (2.6), requires $SL(4M_p+3M+1)$ additions, $SL(M_p+3M)$ multiplications, $SL(2M_p+M)$ logarithms, SLM_p absolute values, and $SL(L\text{-}1)$ comparisons.

Table 2.1 shows practical cases of the binary and non-binary codes parameters. Table 2.2 and Table 2.3 present the estimated total computational load in terms of the number of operations *per bit per half iteration* (one constituent decoder), for AWGN and BSC channels,

respectively. The complexity of the interleaver block is not taken into account. We notice that the complexity of the duo-binary decoder is approximately 1.7 to 1.8 times greater than that of the binary decoder for both AWGN and BSC channels. On the other hand, this ratio is greater than 10 for the case of the quadri-binary decoder. Note that the high complexity in the case of the BSC channel is due to the fact that we also counted the multiplications by 0, -1 and 1, whereas such operations can be avoided in reality.

From the above analysis, we conclude that, in applications where the decoding complexity is a major concern, duo-binary turbo-codes can offer the best tradeoff between decoding complexity and error correction capability. On the other hand, when a very low BER is desired with sufficient resources at the decoder, quadri-binary codes (higher order codes, with $M > 4$, are not used in practice [DIV95]) offer the best performance. Therefore, in the next chapters, duo-binary codes will be used for the compression of theoretical sources with a BSC correlation model, whereas quadri-binary codes will be used for distributed video compression.

	Binary	Duo-binary	Quadri-binary
M	1	2	4
L	2	4	16
M_p	1	2	1
S	8	8	16

Table 2.1 Parameter values for binary and non-binary codes used in this manuscript

	Number of operations	Binary	Duo-binary	Quadri-binary
Additions	$[SL(M_p+M+7)-4S+L-1]/M$	113	161.5	755.75
Multiplications	$[SL(M_p+M)]/M$	32	64	320
Comparisons	$[SL^2+3SL-4S-2L+2]/M$	46	93	1192.5
Total (OPs)	$[SL^2+SL(2M_p+2M+10)-8S-L+1]/M$	191	318.5	2268.25

Table 2.2 Number of operations (per bit per half iteration) for decoding binary and non-binary turbo-codes used for error correction with an AWGN channel.

	Number of operations	Binary	Duo-binary	Quadri-binary
Additions	$[SL(4M_p+3M+7)-4S+L-1]/M$	193	321.5	1459.75
Multiplications	$[SL(M_p+3M)]/M$	64	128	832
Comparisons	$[SL^2+3SL-4S-2L+2]/M$	46	93	1192.5
Logarithms	$[SL(2M_p+M)]/M$	48	96	384
Absolute values	$[SLM_p]/M$	16	32	64
Total (OPs)	$[SL^2+SL(8M_p+7M+10)-8S-L+1]/M$	367	670.5	3932.25

Table 2.3 Number of operations (per bit per half iteration) for decoding binary and non-binary turbo-codes used for DSC of 2 sources with a BSC correlation model.

2.8 Conclusion

In this chapter, we have presented turbo-codes and their applications in source and channel coding. We developed a generalized Max-Log-MAP decoding algorithm and derived decoding metrics for two different application cases: channel coding with a transmission over an AWGN channels, and distributed source coding with a BSC correlation model between sources. Simulation results showed that non-binary codes offer a higher decoding performance and a greater resistance to puncturing, compared to binary codes. Finally, we presented a complexity analysis for the Max-Log-MAP decoding algorithm and determined that duo-binary turbo-codes could be a good choice for performance-complexity tradeoff. This study allowed for the selection of a suitable code that will be used in the next chapters for joint source-channel coding and distributed video compression.

Chapter 3

Applications of Non-binary Turbo-Codes in Joint Source-Channel Coding

3.1 Introduction

In the previous chapter, turbo-codes were introduced as an efficient tool for channel coding and source coding. In the former case (channel coding), they are used to protect the data stream from channel impairments, whereas in the latter case (source coding), their main objective is the reduction of the amount of transmitted data by eliminating redundant information from the input signal. In this chapter, we first show how turbo-codes can be used to jointly compress a message signal and protect the compressed stream from transmission errors. As an application case, we will consider the distributed source coding system studied in the previous chapter. However, the output data will be transmitted over a noisy channel. Finally, we will consider joint source-channel coding from a networking point of view, where it can be seen as a cross-layer optimization approach.

3.2 Distributed Source Compression with Transmission over Error-Prone Channels

3.2.1 Turbo-decoding algorithm

In this section, we present the turbo-decoding procedure in the case where the generated compressed data and the side information are both transmitted in the presence of additive white Gaussian noise (AWGN). As in the previous chapter, the side information is a sequence statistically dependent on the input source and generated by means of a BSC. Figure 3.1 presents the system model considered in this study for the general case of an M-ary turbo-encoder used for data compression.

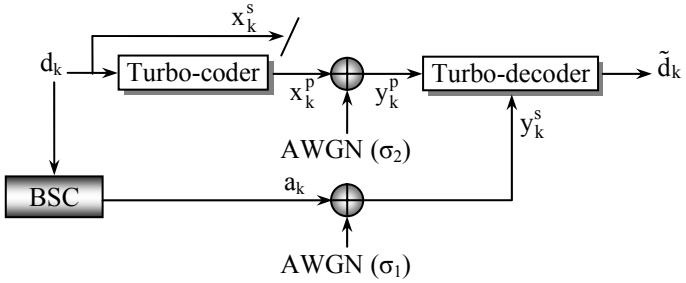


Figure 3.1 Block-diagram of the DSC system model in the presence of noise.

In the presence of noise, the turbo-decoding algorithm remains unchanged, with the exception of the conditional probability $P(y_k \mid d_k)$, which will be derived next.

From Eq. (2.25), we know that $P\left(y_k \mid d_k\right) = P\left(y_k^s \mid x_k^s\right) P\left(y_k^p \mid x_k^p\right)$. The conditional probability for the parity bits can be deduced from Eq. (2.27). Therefore:

$$P\left(y_k^p \mid x_k^p\right) = \prod_{i=1}^{M_p} \frac{1}{\sigma_2 \sqrt{2\pi}} \exp\left(-\frac{\left(y_{ik}^p - x_{ik}^p\right)^2}{2{\sigma_2}^2} \right). \tag{3.1}$$

On the other hand, $P\left(y_k^s \mid x_k^s\right) = \sum_{a_k} P\left(y_k^s, a_k \mid x_k^s\right)$.

Since $P\left(y_k^s, a_k \mid x_k^s\right) = \dfrac{P\left(y_k^s, a_k\right) P\left(x_k^s \mid a_k\right)}{P\left(x_k^s\right)} = \dfrac{P\left(y_k^s, a_k\right) P\left(x_k^s, a_k\right)}{P\left(a_k\right) P\left(x_k^s\right)}$ and $x_k^s = d_k$, we can therefore write:

$$P\left(y_k^s \mid x_k^s\right) = \sum_{a_k} P\left(y_k^s \mid a_k\right) P\left(a_k \mid d_k\right). \tag{3.2}$$

Since a_k goes through an AWGN channel to yield y_k^s, $P\left(y_k^s \mid a_k\right)$ can be expressed as:

$$P\left(y_k^s \mid a_k\right) = \prod_{i=1}^{M} \frac{1}{\sigma_1 \sqrt{2\pi}} \exp\left(-\frac{\left(y_{ik}^s - a_{ik}\right)^2}{2{\sigma_1}^2} \right). \tag{3.3}$$

On the other hand, since a_k and d_k are separated by a BSC,

$$P\left(a_{ik} \mid d_{ik}\right) = \begin{cases} p, & \text{if } a_{ik} \neq d_{ik} \\ 1-p, & \text{if } a_{ik} = d_{ik}. \end{cases} \tag{3.4}$$

Therefore, we can write, in a compact form:

$$P(a_k \mid d_k) = \prod_{i=1}^{M} \left\{ \left(\frac{1 + a_{ik} d_{ik}}{2} \right)(1-p) + \left(\frac{1 - a_{ik} d_{ik}}{2} \right) p \right\} = \prod_{i=1}^{M} \frac{1 - a_{ik} d_{ik} (2p-1)}{2}. \tag{3.5}$$

Note that in case $p \to 0$ and $\sigma_1 = \sigma_2$, the system in Figure 3.1 reduces to the classical problem of transmission over an AWGN. On the other hand, in case $\sigma_1 \to 0$, $\sigma_2 \to 0$ and $p \neq 0$, the system reduces to the classical problem of DSC with a BSC correlation model.

3.2.2 Theoretical compression bound of the joint source-channel coding system

As stated in the previous chapter, based on the Slepian-Wolf theorem [SLE73], the lower bound on the compression system can be theoretically estimated by the conditional entropy H(X|Y) of the compressed source X, given the side information Y, provided that Y is perfectly recoverable at the decoder. Therefore, the ideal compression system, with a transmission through an error-prone channel, can be modeled as in Figure 3.2. The decision device outputs '1' if a non-negative symbol was received at its input. Otherwise, it outputs '0'.

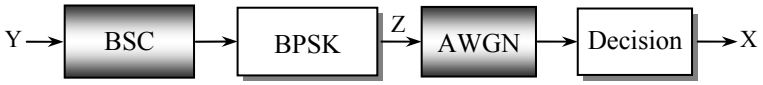


Figure 3.2 Theoretical model of the compression system in the presence of noise.

Let N be a random variable representing the AWGN, and σ^2 the noise variance. We know from equations (2.20) and (2.24) that $\sigma = 1/\sqrt{2E_b / N_0}$, where E_b / N_0 represents the average bit energy per noise density ratio. Note that E_b has been substituted for E_s in the expression of σ since in the theoretical model the coding block is not considered. The probability that a noise sample exceeds 1 can be derived by:

$$P_N = P(N > 1) = \int_1^{\infty} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{n^2}{2\sigma^2} \right) dn = \frac{1}{2} erfc\left(\frac{1}{\sigma\sqrt{2}} \right), \tag{3.6}$$

where *erfc* is the complementary error function. Similary, $P(N < -1) = P_N$.

The conditional entropy of X knowing Y is defined as:

$$H(X \mid Y) = -\sum_{x=0,1} \sum_{y=0,1} P(x,y) \log_2 \left[P(x \mid y) \right] \tag{3.7}$$

To calculate Eq. (3.7), we first need to determine the probabilities $P(x \mid y)$ and $P(x,y)$.

$$\begin{aligned} P(x=0 \mid y=0) &= P(z=-1, n<1 \mid y=0) + P(z=1, n<-1 \mid y=0) \\ &= P(z=-1 \mid y=0) P(n<1) + P(z=1 \mid y=0) P(n<-1) \\ &= (1-p)(1-P_N) + pP_N, \end{aligned} \tag{3.8}$$

where p is the crossover probability of the BSC. On the other hand,

$$\begin{aligned} P(x=1 \mid y=0) &= P(z=-1, n>1 \mid y=0) + P(z=1, n>-1 \mid y=0) \\ &= P(z=-1 \mid y=0) P(n>1) + P(z=1 \mid y=0) P(n>-1) \\ &= (1-p) P_N + p(1-P_N). \end{aligned} \tag{3.9}$$

The symmetry of the Gaussian distribution yields $P(x=1 \mid y=1) = P(x=0 \mid y=0)$ and $P(x=0 \mid y=1) = P(x=1 \mid y=0)$. Additionally, we assume that the sources are uniformly distributed. Therefore, we can write: $P(x,y) = P(x \mid y) P(y) = \frac{1}{2} P(x \mid y)$. By substituting the above equations in Eq. (3.7), we obtain:

$$\begin{aligned} H(X \mid Y) = &-\left[(1-p)(1-P_N) + pP_N \right] \log_2 \left[(1-p)(1-P_N) + pP_N \right] \\ &-\left[(1-p) P_N + p(1-P_N) \right] \log_2 \left[(1-p) P_N + p(1-P_N) \right]. \end{aligned} \tag{3.10}$$

3.2.3 Practical results

We first simulated the system in Figure 3.1 by considering the duo-binary turbo-codec presented in the previous chapter, with the decoding algorithm modified as explained in Section 3.2.1.

Figure 3.3 represents the BER obtained for different values of E_s / N_0 for the case where both the side information and the parity bits are distorted by the same amount of noise ($\sigma_1 = \sigma_2$). We notice a quick deterioration of the system performances when E_s / N_0 is below 5 dB.

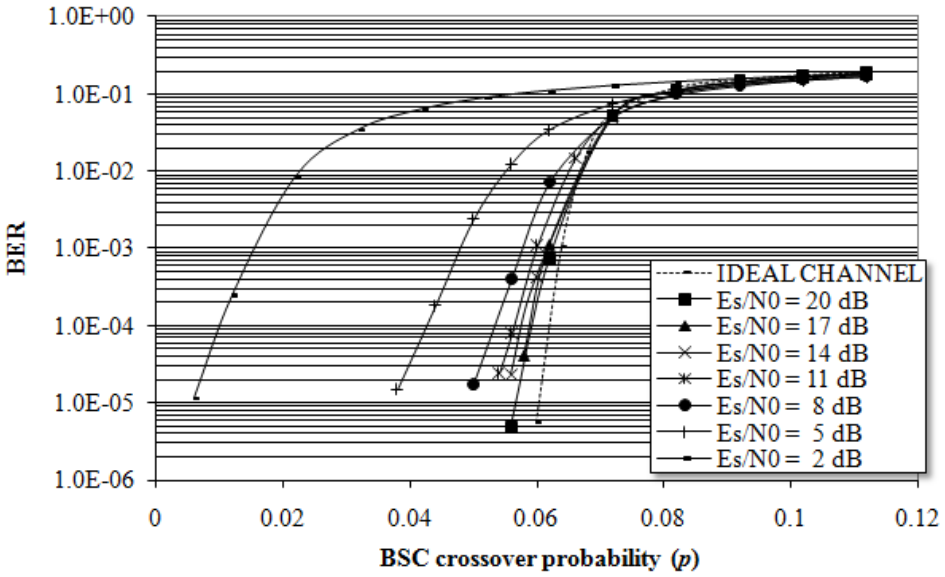


Figure 3.3 Bit Error Rate obtained for a compression ratio 2:1 in the presence of AWGN.

On the other hand, we show in Figure 3.4 the simulation results obtained for the case where the side information and the parity bits experience two transmission channels with different E_s / N_0 ratios. It can be clearly observed that the compression system is much more sensitive to the distortion on the parity information than on the side information, especially for small values of the crossover probability.

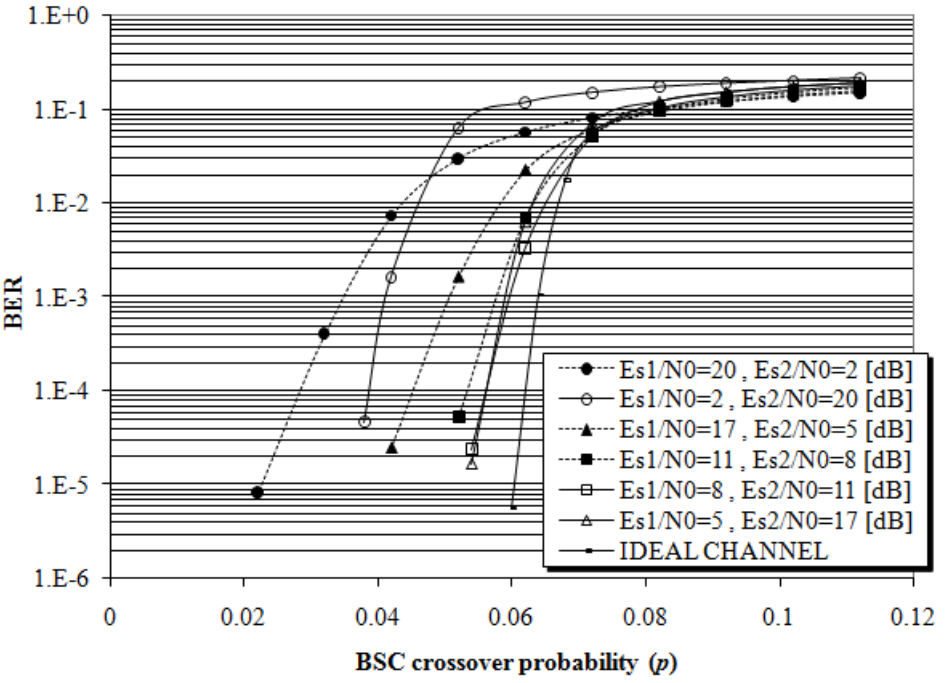


Figure 3.4 Bit Error Rate obtained for a compression ratio 2:1 with different E_s / N_0 ratios.

In Figure 3.5, we represent the compression rate (defined as the ratio of the number of output bits over the number of input bits) required to achieve a BER of 10^{-4} as a function of the crossover probability, for different values of E_s / N_0. In this case, the side information is assumed to be received in the absence of noise. The achievable compression rate can therefore be compared to its theoretical lower bound H (X | Y) calculated using Eq. (3.10) (curves labeled 'TL' in the figure), based on the theoretical model of Figure 3.2. We can observe that the experimental rate increases quickly when p increases, especially when the signal to noise ratio is less than 5 dB. At 5 dB, the additional transmission rate (compared to the theoretical bound) required to achieve an acceptable decoding performance can reach 73% for a crossover probability of 0.05, whereas it is almost 60% in the absence of noise.

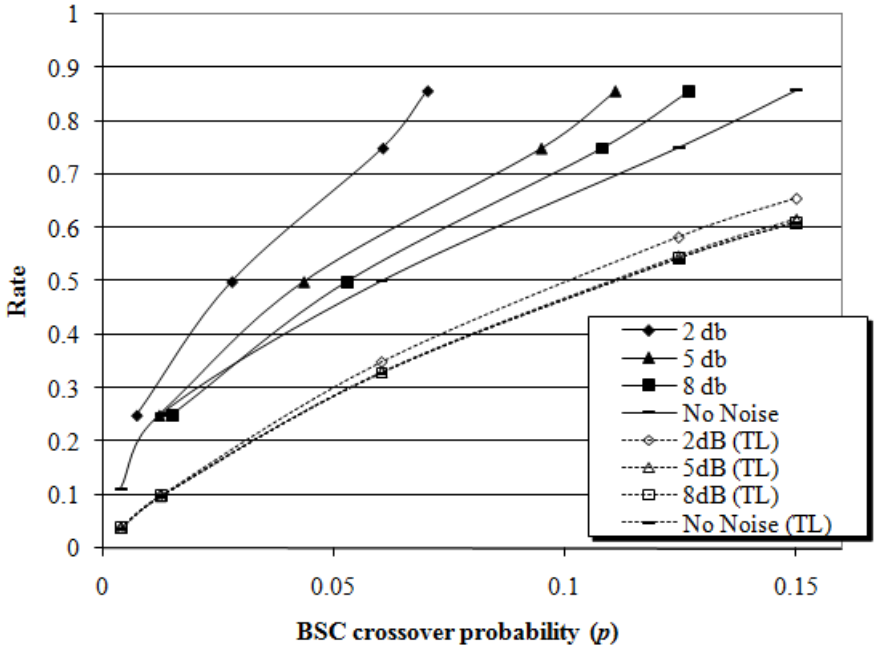


Figure 3.5 Compression rate required to achieve a BER of 10^{-4} as a function of the crossover probability (TL = Theoretical Limit).

3.3 Joint Source-Channel Coding: a Cross-Layer Optimization Approach

In today's networks, layering is used to separate different tasks in the protocol stack where each layer functions independently from the others. This is the traditional approach of the seven-layer Open Systems Interconnections (OSI) model [TAN03]. With the increasing demand on video streaming applications in new generations of wireless and mobile systems [LIU05], existing networks cannot provide end-to-end quality of service (QoS) to a large

number of users using the traditional approach of layer independency, due to user mobility and random channel variations. Optimization techniques have been recently proposed [LIU05, PEN05, BUC04, KHA05, QQU04, WTU04, QQU05] allowing the exchange of information between different layers, thus breaking the traditional rule of layer independency. The so-called "Cross-Layer Optimization" techniques have gained a great deal of attention during the last several years.

On the other hand, turbo-codes introduced in 1993 as near Shannon limit [SHA48] error correcting codes are particularly suitable for video applications, since in such applications a very low bit error rate (BER) is required. Additionally, as stated in the previous chapter, several coding rates could be obtained using a single encoder-decoder system by applying different puncturing schemes [ROW00], therefore allowing flexible rate allocation during data transmission.

Variable channel coding has been used in [QQU04] in a cross-layer approach, where motion information inside each frame is used to assign a different amount of redundancy bits to different parts of the frame, thus achieving unequal error protection (UEP). In [QQU05], the channel code rate is varied according to the source coding rate in order to maintain a constant transmission rate with a minimum distortion. In [KHA05], each user's transmission rate is varied by the modulation type and the retransmission scheme.

In this section, we consider the use of turbo-codes in joint source-channel coding from a networking point of view. Therefore, we develop a simplified cross-layer optimization system model based on rate compatible punctured turbo-codes (RCPT) [ROW00] for rate allocation between several users, where the user with the worst channel is allocated the highest transmission rate. The amount of redundancy bits used to protect each video frame depends on its content, its position within the transmitted Group Of Pictures (GOP) and on the channel conditions. In contrast with previous work, our system relies on a variable channel code not only to achieve UEP, but also to distribute channel resources among several users in a quasi-optimal way in order to improve the overall end-to-end system performance.

3.3.1 System architecture

Consider a set of N users requesting video from a streaming server as shown in Figure 3.6. Since the wireless channel and the network resources are shared, our aim is to optimize the rate allocation between all users, in order to provide improved video quality for users

experiencing bad channel conditions, while maintaining a minimum level of satisfaction for users experiencing good channel conditions.

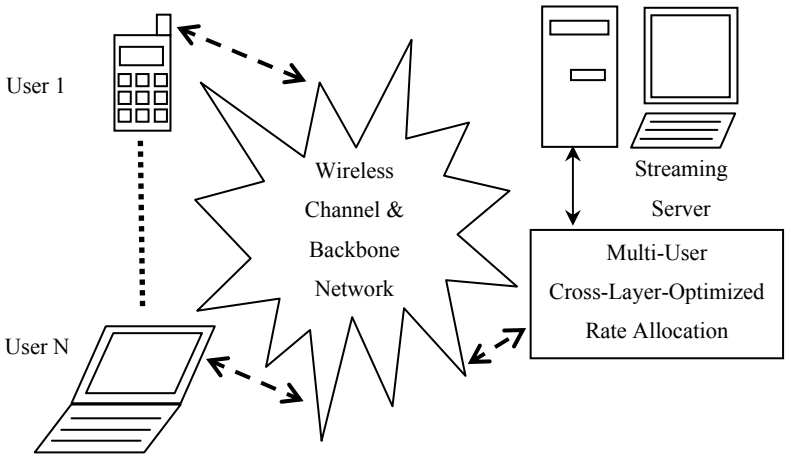


Figure 3.6 Cross-layer optimization system.

Video sequences are encoded and stored on a streaming server. We assume H.264 [ITU03] video coding with an error-concealment strategy [PEN05] described as follows: if an error occurs in a frame, this frame and all the following ones in the same GOP are replaced by the last correctly decoded frame. The distortion matrix [WTU04] for each GOP is calculated as will be described later and transferred to the Rate Control block. A feedback channel between each user and the optimizer is assumed. Therefore, whenever an error occurs at the receiver side, the optimizer recognizes the distortion at the receiver and optimally allocates resources for the next transmission.

From a networking point of view, source coding (video compression) is performed in the application layer, whereas channel coding is performed in the lower layers. Since network communication protocols are beyond our interest in this study, we only consider a simplified two-layer model as shown in Figure 3.7, where the upper layers of the OSI model are referred to as the "Application" layer and the lower layers as the "Link" layer. These layers communicate with the optimizer which selects the best operating mode for each one. As a practical example, channel state information (CSI) could be provided to the optimizer by the Link layer, and the type of data (e.g. video, audio…) by the Application layer; the optimizer then selects optimal parameters and transfers them to the corresponding layers (e.g. source coding rate, channel coding rate, modulation type, etc…) in order to provide improved overall QoS.

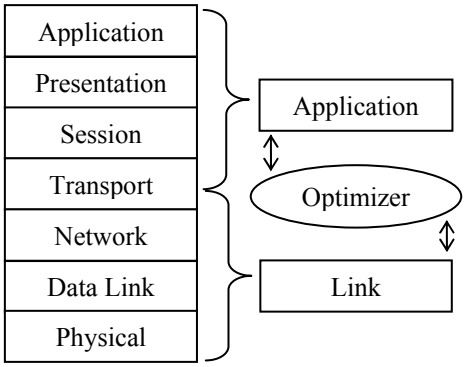


Figure 3.7 Simplified two-layer model of the optimization system.

Figure 3.8 shows a block diagram of the proposed system. Raw video sequences are first source-coded with an H.264 video encoder. The resulting compressed video, along with the corresponding distortion matrix, are then stored on the video streaming server. Upon video streaming request from a mobile user, the H.264 video stream is turbo-coded and stored in a buffer before transmission. The rate-control block decides of the amount of data to be transmitted from the buffer to the mobile device. Based on feedback information from the receiver and the distortion matrix available at the transmitter side, the rate-control block decides either to transmit additional redundancy bits to allow successful decoding of the transmitted block, or to inform the receiver to perform error concealment for the GOP currently being transmitted. This is a cross-layer approach since the transmission rate control which occurs in the Link layer is mainly based on distortion information provided by the Application layer.

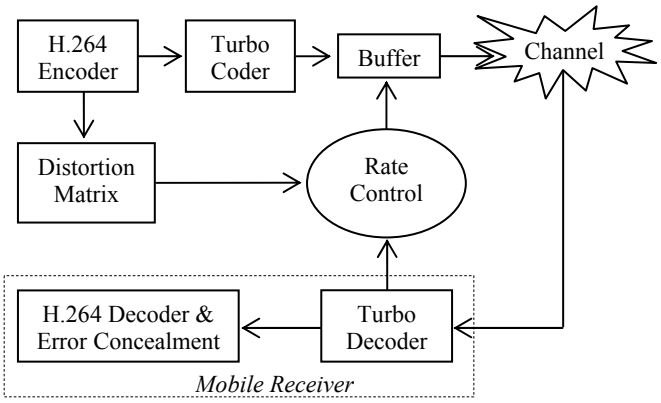


Figure 3.8 Block-diagram of the proposed cross-layer optimization system.

In the previous chapter, we have shown the high gain in performance achieved by using non-binary turbo-codes, especially for highly punctured codes. Therefore, quadri-binary turbo-codes presented earlier are used in our video streaming system.

As we said before, the optimization process in our proposed system uses a matrix containing distortion information calculated at the application layer. The distortion is measured by the mean-square-error (MSE) defined as:

$$D_{F_{or}}^{F_{dec}} = \frac{1}{HW}\sum_{h=1}^{H}\sum_{w=1}^{W}\left(F_{or}(h,w) - F_{dec}(h,w)\right)^2 \tag{3.11}$$

where F_{or} and F_{dec} represent respectively the original HxW frame and the decoded frame after error concealment. In case a frame is correctly decoded and no error concealment is applied, the distortion only represents the loss due to source coding. In this work, we consider an IPPP…P GOP structure where each GOP consists of an intra-coded (I) frame followed by 14 predicted (P) frames. A distortion matrix for a GOP is constructed as shown in Figure 3.9 and is inspired from [WTU04]. However, in our work, the distortion not only includes the influence of the error concealment strategy but also takes into account the residual errors due to lossy H.264 coding. Besides, in our optimization system, it is not necessary to transmit the distortion matrix as side information to the receiver as was done in [WTU04], since it is needed only at the encoder.

$$\begin{bmatrix}
D_{I_{or}}^{R} & D_{P_{1,or}}^{R} & D_{P_{2,or}}^{R} & D_{P_{3,or}}^{R} & \dots & D_{P_{14,or}}^{R} \\
D_{I_{or}}^{I_{dec}} & D_{P_{1,or}}^{I_{dec}} & D_{P_{2,or}}^{I_{dec}} & D_{P_{3,or}}^{I_{dec}} & \dots & D_{P_{14,or}}^{I_{dec}} \\
D_{I_{or}}^{I_{dec}} & D_{P_{1,or}}^{P_{1,dec}} & D_{P_{2,or}}^{P_{1,dec}} & D_{P_{3,or}}^{P_{1,dec}} & \dots & D_{P_{14,or}}^{P_{1,dec}} \\
D_{I_{or}}^{I_{dec}} & D_{P_{1,or}}^{P_{1,dec}} & D_{P_{2,or}}^{P_{2,dec}} & D_{P_{3,or}}^{P_{2,dec}} & \dots & D_{P_{14,or}}^{P_{2,dec}} \\
. & . & . & . & \dots & . \\
. & . & . & . & \dots & . \\
. & . & . & . & \dots & . \\
D_{I_{or}}^{I_{dec}} & D_{P_{1,or}}^{P_{1,dec}} & D_{P_{2,or}}^{P_{2,dec}} & D_{P_{3,or}}^{P_{3,dec}} & \dots & D_{P_{14,or}}^{P_{14,dec}}
\end{bmatrix}$$

Figure 3.9 Distortion matrix stored at the server side.

As stated earlier, when a frame is lost, this frame and all the following ones in the GOP are replaced by the last correctly decoded frame. However, when the (I) frame of the GOP is lost,

all the frames are replaced by (R), where (R) is the (I) frame of the previous GOP. Each matrix row corresponds to a last correctly decodable frame used for error concealment, and each column corresponds to the original frame before coding and transmission. For example, the first row contains the distortions for the case where the (I) frame of the GOP is lost, and (R) is used for error concealment. In the third row, the distortions are computed for the case where the frame P_2 is lost, the preceding frames are correctly received and P_1 is used for error concealment in the following frames. During the optimization process, the average GOP distortion is needed. It is computed by averaging the corresponding row in the matrix.

Figure 3.10 shows the distortion profile of a GOP from three different QCIF video sequences sampled at the rate of 30 frames per second: Foreman, Carphone, and Mother-Daughter. Each point represents the average MSE of the GOP whenever a frame is lost. The index represents the position of the lost frame within the GOP. For example, when the (I) frame of the GOP is lost, the average GOP distortion is shown at index 1. The index 16 represents the case where all the frames of the GOP are correctly received. It can be noticed that the distortion in this case (due only to source coding) is negligible compared to the case where the first frame of the GOP is lost.

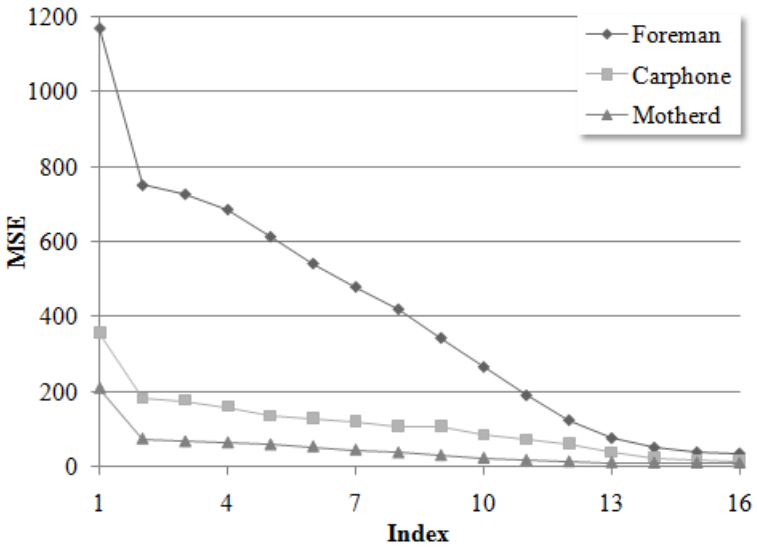


Figure 3.10 Distortion profile of a GOP from 3 different sequences.

3.3.2 Single user communication protocol

Now that we have presented our system architecture, let us describe the communication protocol (Figure 3.11) between a single user and the streaming server. When a QCIF video sequence is requested by a mobile user, it is encoded with an H.264 video coder at a target

bitrate of 100 kbps and then arranged (partitioned) into blocks of 3 kbits. Each block is appended with CRC bits, then turbo-coded at a rate of 2/3 and stored in a buffer. Only

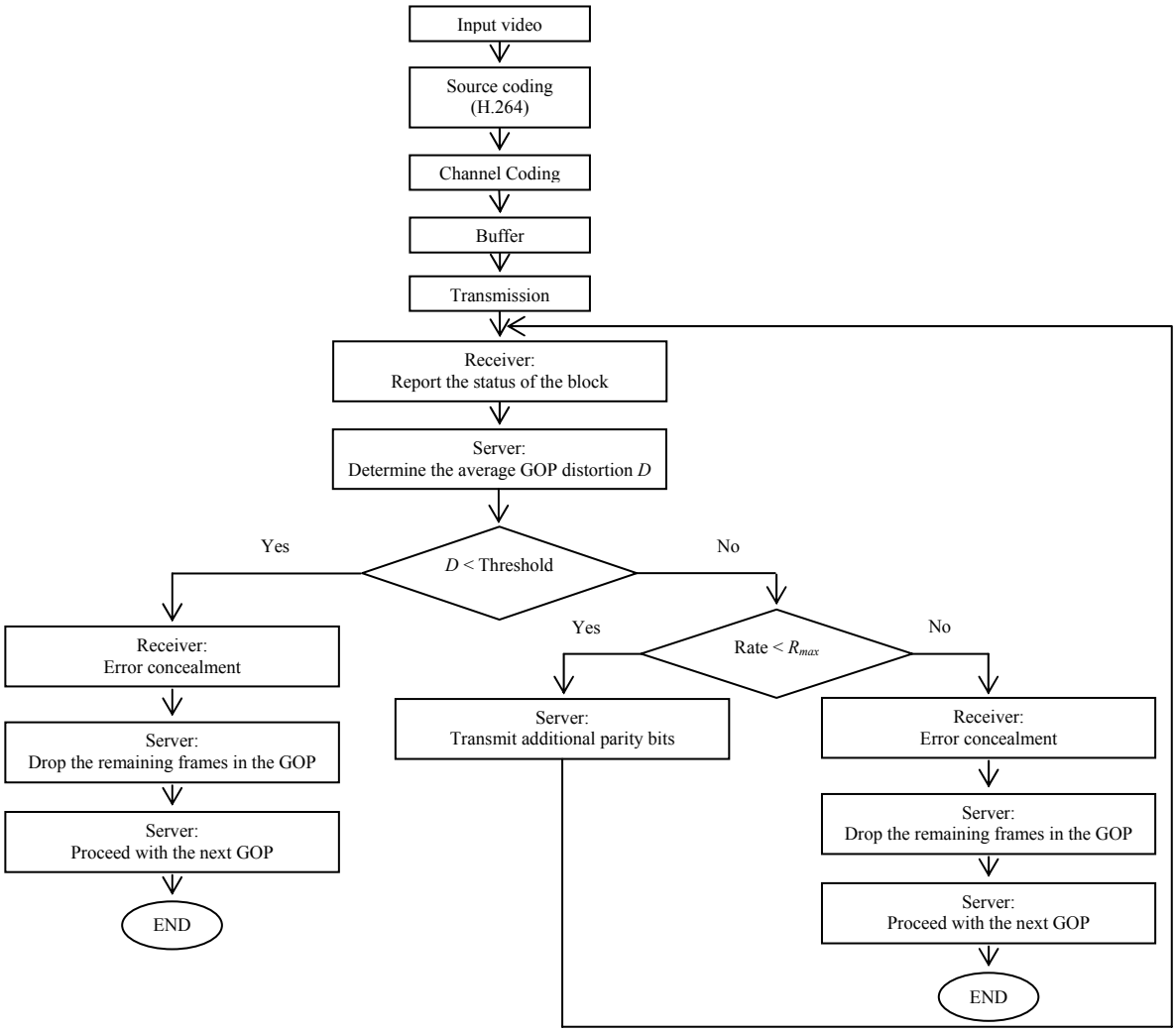


Figure 3.11 Single user communication protocol.

systematic bits are transmitted to the mobile device at the first transmission. The receiver notifies the streaming server whether the received block has been correctly decoded or not. Based on this information, the transmitter determines the average GOP distortion at the receiver side. If the distortion has fallen below a given threshold, which ensures acceptable quality to the mobile user, error concealment is performed. In this case, all remaining transmission blocks of the same GOP are dropped and the system proceeds with the transmission of the next GOP. If the distortion is above the threshold and the maximum allowed transmission rate (R_{max}) has not been reached yet, additional parity bits are transmitted from the buffer to allow correct decoding of the current block, and the same cycle

is repeated again. In this scenario, rate saving is performed on two levels. First, not all of the available capacity is used for the correct decoding of a block, since the amount of transmitted parity bits is the minimum needed to allow successful decoding. Secondly, when the transmitter informs the mobile device to perform error concealment for a given GOP and move to the next one, some blocks are dropped; this leads to additional rate saving and allows transmission at average bit rates that could be even lower than the target source coding rate of 100 kbps. In a multiuser system with rate sharing, the rate saving that occurs with a user experiencing good channel conditions allows the assignment of a greater amount of parity bits to another user experiencing a bad channel, thus improving this user's end-to-end QoS.

3.3.3 Simulation setup and rate allocation strategy for multiuser communication

In our simulations, we consider the case of three mobile users requesting video from the streaming server through an AWGN channel. The users are sharing a data transmission rate of 384 kbps and requesting three different video sequences from the streaming server: Mother-Daughter, Carphone, and Foreman. In a system without optimization, each user will be assigned a constant transmission rate of 128 kbps. As stated earlier, our aim is to unequally allocate the available capacity (384 kbps) between users in order to improve the overall system performance. Upon the transmission of the first GOP, users are allocated 128 kbps each. The system then determines the distortion for each user separately and places the users in a priority queue where the user with the greatest distortion is assigned the highest priority. Upon the next transmission, the user with the worst video quality is allowed up to a maximum of 150 kbps, which corresponds to the case where all the parity bits generated by the turbo-encoder are transmitted (code rate = 2/3). As described earlier, when this user's distortion falls below a given threshold, error concealment is performed and the remaining frames of the same GOP are dropped. At this point, the system determines the effective transmission rate allocated for this user and calculates the remaining rate R' to the two other users. If R' is greater than 250 kbps, the second user in the queue is assigned a maximum of 150 kbps, otherwise it is assigned R'-100 kbps. This aims to guarantee a minimum of 100 kbps for the third (last) user in the queue. This minimum corresponds to its source code rate. The second user is then served as the first one. Finally, the system determines the effective bit rate occupied by the first two users and assigns the remaining bit rate to the last user in the queue. After the transmission of a GOP for every user, the system determines the distortion for each user and updates its priority queue accordingly. The same process is repeated until the video streaming session is terminated.

3.3.4 Simulation results

In order to simulate different transmission conditions for the different users, we assume that E_s / N_0 varies between 0 dB and 2 dB for the first user (Mother-Daughter), between 0 dB and 4 dB for the second user (Carphone) and between 2 dB and 4 dB for the last user (Foreman). We further assume that E_s / N_0 remains constant for the duration of a GOP and that a Peak Signal-to-Noise ratio (PSNR) of 27 dB gives sufficient user satisfaction, which corresponds to an MSE distortion threshold of $D_{threshold} = 130$.

Figure 3.12, Figure 3.13, and Figure 3.14 show the Cumulative Distribution Function (CDF) of the PSNR of the Foreman, Carphone, and Mother-Daughter sequences, respectively, obtained after 100 transmissions through the AWGN channel. In the absence of the optimization technique, the user with the good channel (Foreman) is satisfied almost all the time while the users experiencing average (Carphone) and poor (Mother-Daughter) channel conditions are satisfied for 60% and 20% of the transmission time respectively. Our optimized system improves the performance of the user requesting the Carphone sequence by 20% and the one requesting the Mother-Daughter sequence by 50% when a distortion threshold (T) is not specified, without any significant loss for the user requesting the Foreman sequence. If a distortion threshold is specified, the system tends to approach the target PSNR of 27 dB thus, allowing more satisfaction time for the users having average and poor transmission conditions at the expense of some performance loss for the user having a good channel. For example, without a threshold, the PSNR of the Carphone sequence exceeds 27 dB during 80% of the transmission time, which increases to 90% when the threshold is specified. Similarly with the Mother-Daughter sequence, the PSNR is above 27 dB for 70% and 80% of time, for the cases with and without a threshold, respectively. As for the Foreman sequence, its PSNR never drops below 27 dB in all cases.

Figure 3.15 shows the CDF of the effective transmission rate occupied by all users. In a system without optimization, the capacity of 384 kbps would be constantly fully occupied, while our optimized system requires much less than the available capacity. In fact, the gain in the global transmission rate can reach 256 kbps. It is higher than 96 kbps for 30% of the time, without the threshold (T). In case a distortion threshold (T) is specified, an additional gain of more than 64 kbps during 80% of the transmission time is noticed, compared to the case where no threshold is specified.

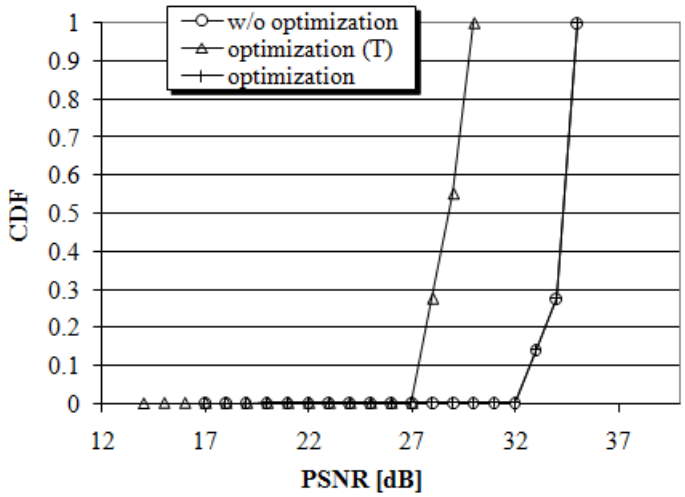


Figure 3.12 CDF of the PSNR of the Foreman sequence.

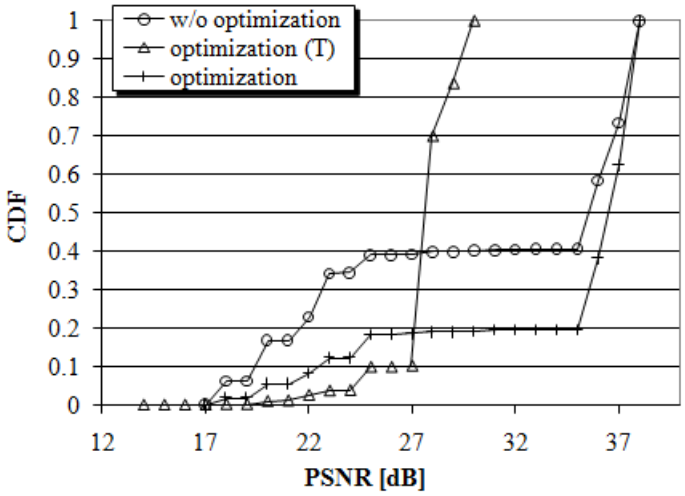


Figure 3.13 CDF of the PSNR of the Carphone sequence.

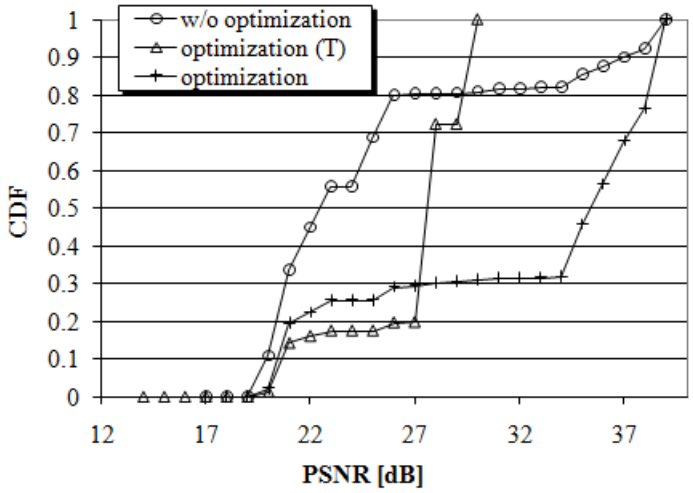


Figure 3.14 CDF of the PSNR of the Mother-Daughter sequence.

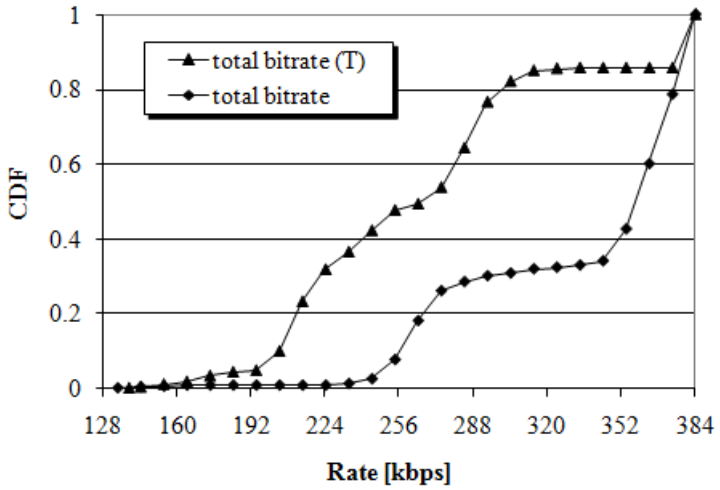


Figure 3.15 CDF of the total transmission rate after optimization.

3.4 Conclusion

In this chapter, we presented two different applications of non-binary turbo-codes in joint source-channel coding. We first studied distributed source compression with transmission over an AWGN channel, where turbo-codes were jointly used for compression and error protection. Then, we considered joint source-channel coding from a networking point of view, and a multiuser video streaming system was developed as a sample application. In this case, turbo-codes were used for channel coding, in a cross-layer optimization approach, taking into account the channel conditions at the link layer and the distortion at the application layer. In the next chapters, turbo-codes will be used as a joint source-channel coding tool for distributed video compression.

Chapter 4

Distributed Video Coding Based on Non-binary Turbo-Codes

4.1 Introduction

In the previous chapters, we presented a detailed study on binary and non-binary turbo-codes and their applications. We also considered distributed source compression for the case of theoretical sources, where the source correlation was modeled as a binary symmetric channel. In this chapter, we study the distributed compression of video sources. After a brief introduction on distributed video coding, we first present a detailed description of the compression system, and derive suitable decoding metrics for the case of noiseless transmission, as well as for the case of error-prone channels. We derive next the theoretical system compression bound, which will be used for the prediction of the necessary compression rate without the need for a feedback channel, therefore allowing the use of distributed video coding in broadcast applications.

4.2 Distributed Video Compression

In traditional video coding techniques such as MPEG or H.26x [ITU03], motion estimation is performed at the encoder side in order to transmit motion information (encoded displacement vectors) to the decoder. At the decoder side, motion compensation is performed based on the received information, in order to reconstitute the video frames. Therefore, this configuration results in very complex inter-frame encoders and simple decoders. It is suitable for applications where a video scene is encoded once in a base station with sufficient resources, and decoded several times in simple receivers. Video broadcasting and video streaming on demand are the most common examples for this configuration: video sequences are compressed and stored on a server, and then streamed to multiple users, upon request for

video-on-demand. A low-complexity decoder is desired in such applications in order to permit low-cost receivers for the end users.

However, in other situations, a simple encoder is desired. Distributed video coding (DVC) was introduced [PUR02, AAR02-2] to permit low-complexity encoding for small power-limited and memory-limited devices, such as camera-equipped mobile phones or wireless video sensors, by moving the computation burden from the encoder side to the decoder. In such scenarios, the decoder is assumed to be located in a base station with sufficient resources.

As in the general case of distributed source coding (section 2.5), DVC mainly relies on Slepian-Wolf [SLE73] and Wyner-Ziv [WYN76] theorems. In practical DVC systems [PUR02, AAR02-2, AAR04-1], a subset of frames, known as key frames, is usually compressed using traditional intra-coding techniques like JPEG or H.264 (intra) [ITU03]. One or more frames following each key frame, known as Wyner-Ziv (WZ) frames, are then compressed by appropriate puncturing of the parity bits at the output of a channel coder. At the receiver side, key frames are used to interpolate the necessary side information for the decoding process.

One of the first practical DVC systems proposed by Puri and Ramchandran [PUR02] used syndrome encoding. In [AAR02-2], Aaron et al. used turbo-codes for the compression of WZ frames. Later on, Ascenso et al. [ASC05-1] proposed a refined motion compensation technique to generate more efficient side information for the system in [AAR02-2]. One of the main drawbacks in all these systems is the use of a feedback channel (FC) [BRI06-1] to allow flexible rate control and to ensure successful decoding of WZ frames. The FC is not suitable for real-time systems due to transmission delay constraints. Additionally, in multiuser applications with rate constraints, the application of WZ coding becomes impractical because of the difficulty of implementing appropriate rate allocation algorithms. Furthermore, since several decoding runs are required to successfully recover a WZ frame, the FC imposes instantaneous decoding in the receiver. For all these reasons, the introduction of new techniques for estimating the necessary bit rate to successfully decode each WZ frame becomes crucial. In fact, the problem of the return channel in DVC has rarely been targeted in the literature. Kubasov et al. proposed in [KUB07-2] an encoder rate control technique that reduces the use of the return channel in a feedback-based DVC codec, by determining the minimal amount of parity bits to be transmitted for the first decoding run in the decoder. A

simple technique that allows the removal of the FC was proposed by Artigas and Torres in [ART06]. The necessary compression rate of a given frame was estimated based on empirical results. This estimation requires building performance tables for the channel code in use, for all possible compression rates and correlation levels between the side information and the WZ frames in a given sequence. Such tables can be built by running offline simulations. However, the influence of the transmission impairments on the decoding performance is not considered. In this case, performance tables should be built not only for all possible correlation levels and compression rates, but also for all possible channel states. This results in a significantly large (theoretically infinite) number of tables that cannot be stored in memory-limited devices. Thus, the proposed technique cannot be used in practical real-time applications with video sequences containing different levels of motion and transmitted over time-varying wireless channels. Morbée et al. [MOR07] proposed another technique for the removal of the feedback channel in DVC. First, the correlation between a WZ frame and the corresponding side information is modeled by a binary symmetric channel (BSC) with a different transition probability for each bitplane. Then, the performances of the channel code in use are estimated offline as a function of the transition probabilities using random binary sequences. The compression rate for a given frame is then determined based on the two previous estimations. This technique presents several disadvantages as well. First, it does not take into consideration the rate constraints in limited bandwidth applications. Besides, when the WZ frame is decoded with a high error rate, the decoded data is discarded in the receiver and replaced by the available side information, yielding wasted channel use. Furthermore, the influence of the channel impairments on the proposed rate allocation technique is not considered.

In this chapter, we target the problem of the return channel in DVC systems and present a novel technique for estimating the necessary compression rate without the need for feedback information from the decoder. The proposed technique relies on information theoretic calculations that take into account the amount of motion in the captured video scene, as well as the transmission channel conditions, in order to predict a suitable compression rate for each frame. Compared to previous studies, our system allows the development of a simple rate allocation algorithm for a multiuser scenario with rate constraints, which will be elaborated in the following chapter.

4.3 System Description

Distributed video coding can be performed in spatial (pixel) domain or in transform domain. We first focus our study on the pixel domain DVC codec shown in Figure 4.1 (transform domain systems will be considered in Chapter 6). As previously mentioned, key frames are compressed using traditional intra-coding techniques. As for the encoding of the WZ frames, the different operational blocks of the system will be explained in the following.

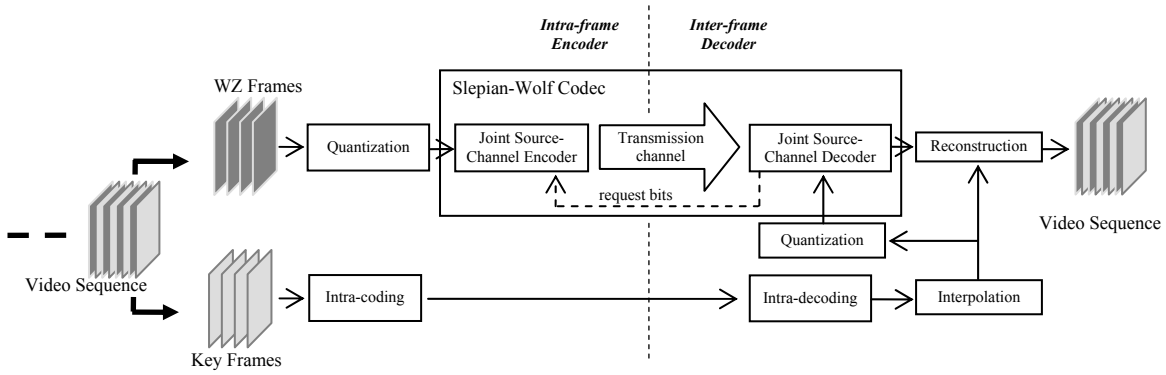


Figure 4.1 Block-diagram of the pixel-domain distributed video codec

4.3.1 Quantization

We consider a uniform linear quantizer that takes at its input 8-bit pixels, and outputs M_q-bit pixels, $M_q \leq 8$. The quantization of WZ frames is performed by taking the M_q most significant bits from each pixel. Quantized pixels are then serially concatenated and fed to the source-channel encoder. In our codec, we allow three possible values for M_q as in [AAR02-2], M_q = 1, 2 or 4.

4.3.2 Source-channel encoder

The joint source-channel encoder (Slepian-Wolf encoder) considered in the system of Figure 4.1 consists of the quadri-binary turbo-encoder presented in the earlier chapters. At the encoder output, systematic information is discarded, while parity information is punctured and transmitted to the decoder. The amount of puncturing is determined by the desired compression rate for each frame. In case of error-free transmission, turbo-coding and puncturing are performed only to achieve source compression. However, for a transmission through a noisy channel, error-protection is also desired, yielding joint source-channel coding.

In the case of a transmission system with a feedback channel, the parity bits at the encoder output are stored in a buffer. Upon request from the decoder, a subset of the parity information is transmitted for the decoding of the current WZ frame. This process is then iterated until a decoding bit error rate of 10^{-3} is reached. Another stopping criterion based on Cyclic Redundancy Check (CRC) can also be used in the receiver, as was done in [KUB07-2]. However, in the case of a broadcasting application, a feedback channel does not exist. Therefore, the system must rely on a fixed amount of parity information, which will be determined later in this chapter.

As stated earlier, quantized pixels of the WZ frames are serially concatenated and fed to the turbo-encoder. In other words, in our system (Figure 4.1), no bitplane extraction [AAR02-2, AAR04-1, ASC05-1, BRI06-1] is performed. Aaron et al. mentioned in [AAR04-1] that both techniques (serial concatenation and bitplane extraction) yield similar results. However, we have noticed that bitplane coding presents a major disadvantage in case the FC is suppressed: If the decoding of a current bitplane involves previously decoded bitplanes as in [ASC05-1, AAR04-1], than the necessary compression rate to successfully decode the current bitplane cannot be determined without a return channel. Moreover, the compression of a certain bitplane cannot begin until the necessary parity bits corresponding to the previous bitplane have all been transmitted. In fact, this procedure guarantees the least compression on the most significant bits (MSB) in the quantized pixels, and the strongest compression on the least significant bits (LSB). However, in the absence of a return channel, a very strong compression on a bitplane can lead to a high decoding bit error rate (BER) and, consequently, a bad reconstructed output. If the compression was slightly stronger on the MSB and softer on the LSB, while keeping the same average compression rate per pixel, the system error correction capability would often yield a better result. This can be performed by directly feeding the quantized pixels to the turbo-encoder instead of extracting bitplanes. Additionally, the use of a quadri-binary turbo codec, by a serial concatenation of the quantized pixels at the encoder input, allows for non-binary turbo-coding, which yields improved decoding performance compared to binary coding, as shown in the previous chapters.

In the case where the quantization parameter M_q is 4, the quadri-binary turbo-encoder input corresponds to one quantized pixel. In case $M_q = 2$ (resp. 1), it corresponds to two (resp. four) serially concatenated pixels.

4.3.3 Interpolation of the side information

In Wyner-Ziv coding of video, key frames are used to interpolate the side information necessary for the turbo-decoding process. Several interpolation techniques for generating accurate side information have been presented in the literature. Aaron et al. first proposed average interpolation and motion-compensated interpolation in [AAR02-2]. In [ASC05-2], Ascenso et al. presented an improved motion-compensated interpolation using spatial motion smoothing. In hash-based DVC [AAR04-2, ASC07], the transmission of hash bits, extracted from the WZ frames, was used to improve the quality of the side information. Edge-based and mesh-based interpolations were proposed in [TOA07] and [KUB06], respectively. Multiple hypothesis techniques were developed in [MIS05] and [KUB07], where two different frames were used as side information for the decoding of a single WZ frame. Each of these methods outperforms some of the others in particular situations (e.g. background motion, moving objects, etc…).

The problem of improving the side information for DVC will be targeted later in Chapter 6, where a novel algorithm will be developed for this purpose. In this chapter, we mainly focus our study on the suppression of the feedback channel. Therefore, we consider only two simple interpolation techniques based on [AAR02-2]: average interpolation (AVI), which will be used later in this chapter, and motion compensated interpolation (MCI) that will be used in the next chapter.

AVI is a very simple interpolation technique since it does not involve motion estimation. The side information Y for a given WZ frame X, located at mid-distance between two key (or previously decoded) frames X_1 and X_2, is obtained by:

$$Y = \frac{X_1 + X_2}{2}. \tag{4.1}$$

More accurate side information can be obtained using MCI [AAR02-2]. For a given block Y_b in the interpolated frame, block-based motion search is performed using X_1 and X_2, assuming symmetric motion vectors between X_1 and Y on one hand, and between Y and X_2 on the other, as shown in Figure 4.2. Block matching is performed using the MSE as a distance measure. The matched blocks X_{1b} and X_{2b} (dotted blocks in the figure) are then used to interpolate Y_b as:

$$Y_b = \frac{X_{1b} + X_{2b}}{2}. \tag{4.2}$$

It is important to note that with MCI, it is necessary to perform motion search in a small search window to avoid false matches, since the matching criterion is the MSE between X_{1b} and X_{2b}.

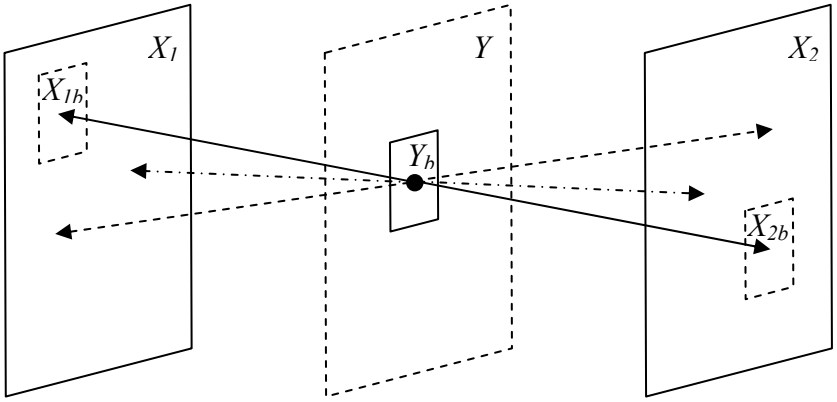


Figure 4.2 Motion compensated interpolation with symmetric motion vectors.

The histogram of the residual error (X-Y) can be estimated by a Laplacian distribution [AAR02-2], which can be expressed as:

$$P(X - Y = d) = \frac{\alpha}{2}\exp(-\alpha|d|) \tag{4.3}$$

The parameter α can be obtained by fitting the Laplacian distribution function to the histogram of the residual X-Y, as shown in Figure 4.3.

Let Z be a random variable following a Laplacian distribution. The expectation of Z is obtained by:

$$\begin{aligned} E[Z] &= \int_{-\infty}^{+\infty} z\frac{\alpha}{2}\exp(-\alpha|z|)dz \\ &= \int_{-\infty}^{0} z\frac{\alpha}{2}\exp(+\alpha z)dz + \int_{0}^{+\infty} z\frac{\alpha}{2}\exp(-\alpha z)dz \\ &= -\frac{1}{2\alpha} + \frac{1}{2\alpha} = 0. \end{aligned} \tag{4.4}$$

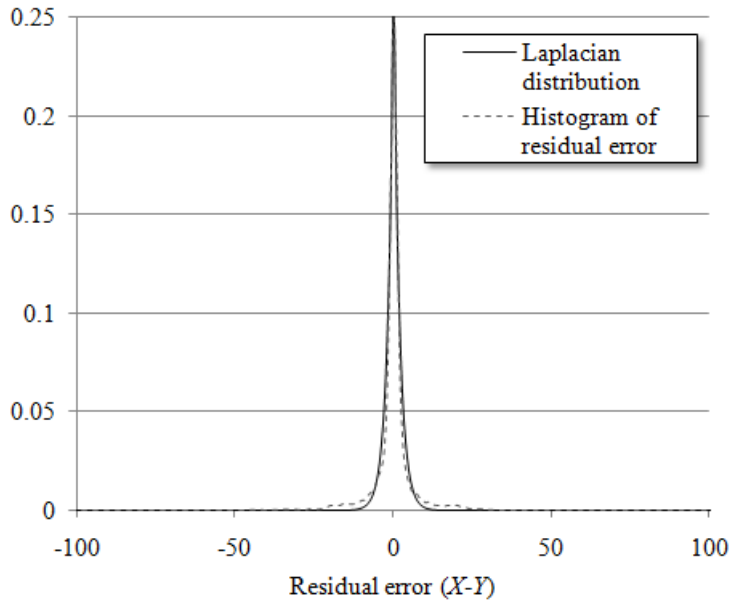


Figure 4.3 Laplacian distribution (α=0.52) and sample histogram of the residual error *X*-*Y* from the Carphone video sequence.

On the other hand, the variance of *Z* is expressed as:

$$\begin{aligned}\sigma_Z^2 &= E\left[Z^2\right]-E^2\left[Z\right] \\ &= \int_{-\infty}^{+\infty} z^2 \frac{\alpha}{2}\exp\left(-\alpha|z|\right)dz \\ &= \int_{-\infty}^{0} z^2 \frac{\alpha}{2}\exp\left(+\alpha z\right)dz + \int_{0}^{+\infty} z^2 \frac{\alpha}{2}\exp\left(-\alpha z\right)dz \\ &= \frac{1}{\alpha^2}+\frac{1}{\alpha^2}=\frac{2}{\alpha^2}.\end{aligned} \tag{4.5}$$

As a result, α can be simply estimated as:

$$\alpha = \sqrt{\frac{2}{\sigma_{X-Y}^2}}, \tag{4.6}$$

where σ_{X-Y}^2 is the variance of the residual *X*-*Y*.

The parameter α is needed for the turbo-decoding process. It can be computed at the encoder and transmitted as additional side information to the decoder, or it can be estimated at the decoder side using the frames X_1 and X_2. In the first case, since motion estimation cannot be performed in the encoder (for complexity reasons), *Y* is always replaced, in equation (4.6), by its average-interpolated estimation. It was observed in [BRI06-2] that the performance loss

due to such imperfect measures of the parameter α is minor. Therefore, in our study, we will assume that α can be perfectly estimated at the decoder.

4.3.4 Source-channel decoder

The joint source-channel decoder (Slepian-Wolf decoder) consists of a quadri-binary turbo-decoder where each constituent decoder is based on the Max-Log-MAP decoding algorithm. The decoding metrics are the same as the ones we detailed in Chapter 2, with the exception of the conditional probability $P(y_k \mid d_k)$ in Eq. (2.6), which needs to be modified for the case of DVC.

First, it should be noted that the Laplacian distribution function in Eq. (4.3) is for the case of continuous random variables. Since we are dealing with discrete values (quantized pixels), we express the distribution of the residual X-Y as:

$$P\left(X-Y=\Delta\right)=c\frac{\alpha}{2}\exp\left(-\alpha\left|d_{\Delta}\right|\right), \tag{4.7}$$

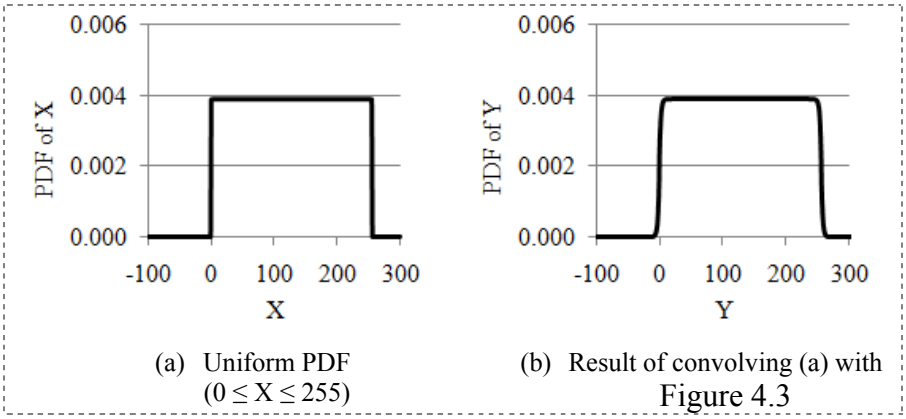
where $d_{\Delta}=2^{8-M_q}\Delta$, and c is a scaling factor calculated such that $\sum_{\Delta}P\left(X-Y=\Delta\right)=1$:

$$c=\frac{2/\alpha}{\sum_{\Delta=-\left(2^{M_q}-1\right)}^{2^{M_q}-1}\exp\left(-\alpha\left|d_{\Delta}\right|\right)}=\frac{2/\alpha}{1+2\exp\left(-\alpha 2^{8-M_q}\right)\dfrac{1-\exp\left(-\alpha 2^{8-M_q}\left[2^{M_q}-1\right]\right)}{1-\exp\left(-\alpha 2^{8-M_q}\right)}}. \tag{4.8}$$

Obviously, $P\left(X-Y\right)=P\left(Y-X\right)$. On the other hand,

$$P\left(X-Y=\Delta\right)=\sum_{b}P\left(X=a=\Delta+b\mid Y=b\right)P\left(Y=b\right). \tag{4.9}$$

Since the Laplacian probability distribution function (PDF) is very sharp at 0, it can be seen as an impulse function (Figure 4.3). On the other hand, the PDF of the pixels values in a given frame X is highly dependent on the content of the captured video. In the general case, we can assume that X follows a uniform distribution, i.e. all pixel intensity values are equally likely to occur. Therefore, the PDF of the side information Y, being the result of the convolution of the PDF of X with the PDF of the residual, can also be approximated by a uniform distribution, as it can be seen in the example of Figure 4.4.



(a) Uniform PDF ($0 \le X \le 255$)

(b) Result of convolving (a) with Figure 4.3

Figure 4.4 Example of convolving a Laplacian PDF with a uniform PDF.

The number of non-zero terms in Eq. (4.9) is equal to the number $L_{d_{a-b}}$ of couples (a,b) that yield the residual difference $\Delta = a - b$. Assuming an equiprobable source, these couples can be considered to be equally likely; for example, considering a 2-bit quantization (M_q=2), the possible values of Δ are 0, ±1, ±2 and ±3 (i.e. d_Δ = 0, ±64, ±128 and ±192), with a decreasing order of probability of occurrence. For a particular value of the difference Δ, couples (a,b) that yield the difference Δ have the same probability to occur. For example, for Δ = 1, the occurrences of (a = 1, b = 0), (a = 2, b = 1) and (a = 3, b = 2) are equally likely. Therefore, we can write:

$$P\left(X = a \mid Y = b\right) = \frac{2^{M_q}}{L_{d_\Delta}} P\left(X - Y = \Delta\right). \tag{4.10}$$

By calculating L_{d_Δ} for different values of Δ (refer to Appendix A for more details), we found that it can be expressed as:

$$L_{d_\Delta} = 2^{M_q} - \left|\Delta\right|. \tag{4.11}$$

Besides, since $P\left(X - Y\right) = P\left(Y - X\right)$, $\left|d_\Delta\right| = \left|2^{8-M_q}\Delta\right| = \left|2^{8-M_q}\left(-\Delta\right)\right| = \left|d_{-\Delta}\right|$, and $L_{d_\Delta} = 2^{M_q} - \left|\Delta\right| = L_{d_{-\Delta}}$, we have:

$$\begin{aligned} P\left(Y = b \mid X = a\right) &= \frac{2^{M_q}}{L_{d_{b-a}}} P\left(Y - X = b - a\right) = \frac{2^{M_q}}{L_{d_{a-b}}} P\left(X - Y = a - b\right) \\ &= P\left(X = a \mid Y = b\right). \end{aligned} \tag{4.12}$$

On the other hand, from Eq. (2.25), we know that $P\left(y_k \mid d_k\right) = P\left(y_k^s \mid x_k^s\right) P\left(y_k^p \mid x_k^p\right)$. Let us first derive $P\left(y_k^s \mid x_k^s\right)$.

Let $x_k^s = \left\{x_{1k}^s, x_{2k}^s, x_{3k}^s, x_{4k}^s\right\}$ be a group of four systematic bits at the input of the turbo-encoder at time instant *k*, and $y_k^s = \left\{y_{1k}^s, y_{2k}^s, y_{3k}^s, y_{4k}^s\right\}$ its corresponding side information. Therefore,

$$P\left(y_k^s \mid x_k^s\right) = P\left(y_{1k}^s, y_{2k}^s, y_{3k}^s, y_{4k}^s \mid x_{1k}^s, x_{2k}^s, x_{3k}^s, x_{4k}^s\right). \tag{4.13}$$

As stated before, in the case where the quantization parameter $M_q = 4$, the four systematic bits at the turbo-decoder input correspond to one quantized pixel from the interpolated frame.

Let $X = 8x_{1k}^s + 4x_{2k}^s + 2x_{3k}^s + x_{4k}^s$ be the bin index of the quantized pixel in the WZ frame, and $Y = 8y_{1k}^s + 4y_{2k}^s + 2y_{3k}^s + y_{4k}^s$ the bin index of the corresponding quantized side information. Using equations (4.10) , (4.12) and (4.13), we can write:

$$P\left(y_k^s \mid x_k^s\right) = P\left(Y \mid X\right) = \frac{2^{M_q}}{L_{d_{Y-X}}} P\left(Y - X\right), \tag{4.14}$$

where $P\left(Y - X\right)$ is defined in Eq (4.7).

In the case where $M_q = 2$, the four systematic bits correspond to two quantized pixels. Let $X_1 = 2x_{1k}^s + x_{2k}^s$ and $X_2 = 2x_{3k}^s + x_{4k}^s$ be the bins of the two WZ quantized pixels with their corresponding side information $Y_1 = 2y_{1k}^s + y_{2k}^s$ and $Y_2 = 2y_{3k}^s + y_{4k}^s$. Therefore, assuming that succeeding quantized pixel values are independent,

$$\begin{aligned} P\left(y_k^s \mid x_k^s\right) &= P\left(Y_1 \mid X_1\right) P\left(Y_2 \mid X_2\right) \\ &= \frac{2^{2M_q}}{L_{d_{Y_1-X_1}} L_{d_{Y_2-X_2}}} P\left(Y_1 - X_1\right) P\left(Y_2 - X_2\right). \end{aligned} \tag{4.15}$$

When $M_q = 1$, each systematic bit corresponds to one pixel. Let $X_i = x_{ik}^s$ be the bin index of the quantized pixel *i* (*i* = 1, …, 4) and $Y_i = y_{ik}^s$ the corresponding side information. In this case,

$$P\left(y_k^s \mid x_k^s\right) = \prod_{i=1}^{4} P\left(Y_i \mid X_i\right) = \prod_{i=1}^{4} \frac{2^{M_q}}{L_{d_{Y_i - X_i}}} P\left(Y_i - X_i\right). \tag{4.16}$$

Now that $P\left(y_k^s \mid x_k^s\right)$ has been determined for all possible values of M_q, we can proceed with the calculation of $P\left(y_k^p \mid x_k^p\right)$. In contrast with our metrics derived in the previous chapters, to further improve our decoding performance, we will take into consideration the puncturing probability. First, let $P_{punct}(\rho)$ the probability of puncturing a parity bit expressed as a function of the compression rate ρ for the current WZ frame. ρ is defined as the ratio of the number N_p of remaining parity bits after puncturing over the total number of systematic bits N_s. Since the number of parity bits at the turbo-encoder output is $N_s/2$ before puncturing, we can write:

$$P_{punct}(\rho) = \frac{\frac{N_s}{2} - N_p}{\frac{N_s}{2}} = 1 - 2\rho. \tag{4.17}$$

In case x_k^p has been punctured at the output of the turbo-encoder, y_k^p has no value ($y_k^p = nv$). Therefore, regardless of the nature of the transmission channel,

$$P\left(y_k^p \mid x_k^p, x_k^p \text{ is punctured}\right) = \begin{cases} 1, \text{ if } y_k^p = nv \\ 0, \text{ otherwise} \end{cases} \tag{4.18}$$

On the other hand, if x_k^p has not been punctured, y_k^p can take two possible values, 0 or 1. In this case, we consider three types of transmission channels: noise free, AWGN, and BSC.

In the case of a noise free channel, no transmission errors occur. Therefore,

$$P\left(y_k^p \mid x_k^p, x_k^p \text{ is not punctured}\right) = \begin{cases} 0 & , \text{ if } y_k^p = nv \\ \begin{cases} 0, \text{ if } y_k^p \neq x_x^p \\ 1, \text{ if } y_k^p = x_x^p \end{cases} & , \text{ if } y_k^p = 0 \text{ or } 1. \end{cases} \tag{4.19}$$

In case of a BSC with a transition probability q,

$$P\left(y_k^p \mid x_k^p, x_k^p \text{ is not punctured}\right) = \begin{cases} 0 & , \text{ if } y_k^p = nv \\ \begin{cases} q, & \text{if } y_k^p \neq x_x^p \\ 1-q, & \text{if } y_k^p = x_x^p \end{cases} & , \text{if } y_k^p = 0 \text{ or } 1 \end{cases}$$
$$= \begin{cases} 0 & , \text{if } y_k^p = nv \\ q\left[1-\delta\left(y_k^p - x_k^p\right)\right] + (1-q)\,\delta\left(y_k^p - x_k^p\right), & \text{if } y_k^p = 0 \text{ or } 1, \end{cases} \tag{4.20}$$

where $\delta(k) = \begin{cases} 1, & \text{if } k = 0 \\ 0, & \text{otherwise.} \end{cases}$

In case of an AWGN channel with a noise variance σ^2,

$$P\left(y_k^p \mid x_k^p, x_k^p \text{ is not punctured}\right) = \begin{cases} 0 & , \text{ if } y_k^p = nv \\ \dfrac{1}{\sigma\sqrt{2\pi}} \exp\left(-\dfrac{\left(\widehat{y_k^p} - \widehat{x_k^p}\right)^2}{2\sigma^2}\right), & \textit{otherwise}, \end{cases} \tag{4.21}$$

where $\widehat{y_k^p}$ and $\widehat{x_k^p}$ represent the BPSK modulated versions of y_k^p and x_k^p, respectively.

Finally, $P\left(y_k^p \mid x_k^p\right)$ is calculated by:

$$\begin{aligned} P\left(y_k^p \mid x_k^p\right) &= P\left(y_k^p \mid x_k^p, x_k^p \text{ is punctured}\right) P\left(x_k^p \text{ is punctured}\right) \\ &\quad + P\left(y_k^p \mid x_k^p, x_k^p \text{ is not punctured}\right) P\left(x_k^p \text{ is not punctured}\right) \\ &= P\left(y_k^p \mid x_k^p, x_k^p \text{ is punctured}\right) P_{punct}(\rho) \\ &\quad + P\left(y_k^p \mid x_k^p, x_k^p \text{ is not punctured}\right)\left[1 - P_{punct}(\rho)\right] \\ &= (1-2\rho) P\left(y_k^p \mid x_k^p, x_k^p \text{ is punctured}\right) \\ &\quad + 2\rho P\left(y_k^p \mid x_k^p, x_k^p \text{ is not punctured}\right). \end{aligned} \tag{4.22}$$

In the case of a BSC, Eq. (4.22) becomes:

$$\begin{aligned} P\left(y_k^p = 0,1 \mid x_k^p\right) &= P\left(y_k^p = 0,1 \mid x_k^p, x_k^p \text{ is not punctured}\right) P\left(x_k^p \text{ is not punctured}\right) \\ &= P\left(y_k^p = 0,1 \mid x_k^p, x_k^p \text{ is not punctured}\right) \cdot \left(1 - P_{punct}(\rho)\right) \\ &= \left(q\left[1-\delta\left(y_k^p - x_k^p\right)\right] + (1-q)\,\delta\left(y_k^p - x_k^p\right)\right) \cdot 2\rho, \end{aligned} \tag{4.23}$$

and

$$\begin{aligned} P\left(y_k^p = nv \mid x_k^p\right) &= P\left(y_k^p = nv \mid x_k^p, x_k^p \text{ is punctured}\right) P\left(x_k^p \text{ is punctured}\right) \\ &= P\left(y_k^p = nv \mid x_k^p, x_k^p \text{ is punctured}\right) \cdot P_{punct}\left(\rho\right) = 1 - 2\rho. \end{aligned} \tag{4.24}$$

Eq. (4.23) and (4.24) will be used in the next chapter, where a BSC will model the transmission channel.

4.3.5 Reconstruction

The reconstruction block plays the role of inverse quantization, in other words recovering 8-bit versions of the quantized pixel values at the output of the source-channel decoder, based on the knowledge of the side information.

Let X and X_q represent an 8-bit pixel and its quantized counterpart, respectively, Y and Y_q their corresponding side information, X'_q the turbo-decoded version of X_q, and X' the final reconstructed output. X'_q is obtained at the output of the Slepian-Wolf decoder. Therefore, we can assume that $X'_q = X_q$, since the BER after Slepian-Wolf decoding is very low. Based on this assumption, the reconstruction function (from [AAR02-2]) operates as follows:

$$X' = \begin{cases} Y & , \text{if } Y_q = X'_q \\ X'_q & , \text{if } Y_q < X'_q \\ X'_q + 2^{8-M_q} - 1 & , \text{if } Y_q > X'_q \end{cases} \tag{4.25}$$

In other words, if the side information lies within the same bin determined by X'_q, then X' takes the value of the side information. Otherwise, the reconstruction is clipped to the boundary of the bin closest to the side information. An example of the three different cases in the reconstruction process is illustrated in Figure 4.5 for the case where $M_q = 2$.

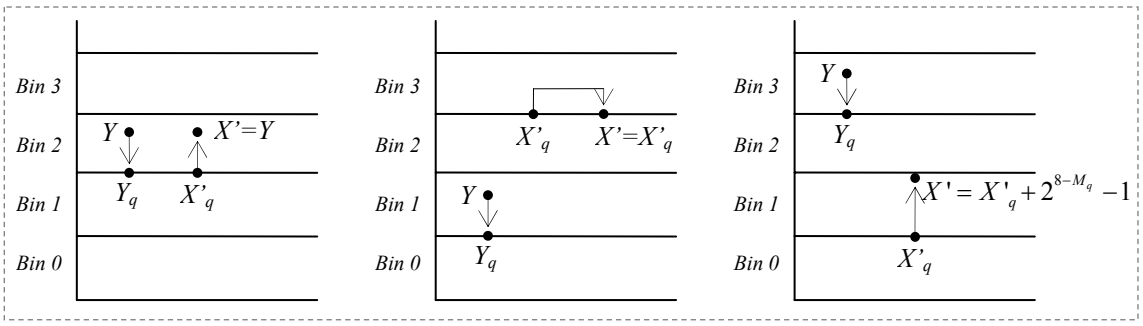


Figure 4.5 Example of the different cases in the reconstruction process.

This reconstruction function has the advantage of limiting the magnitude of the reconstruction distortion (in case $X'_q = X_q$) to a maximum value determined by M_q. In fact, the maximum value e that the absolute error between X' and X can reach is: $e = 2^{8-M_q} - 1$.

4.4 Theoretical Compression Bound of the distributed coding system

4.4.1 Error-free transmission

In case of error-free transmission, the Slepian-Wolf theorem states that a WZ frame can be compressed to a rate close to the conditional entropy defined as:

$$H(X \mid Y) = -\sum_{a=0}^{2^{M_q}-1} \sum_{b=0}^{2^{M_q}-1} P(Y=b) P(X=a \mid Y=b) \log_2 \left(P(X=a \mid Y=b) \right). \qquad (4.26)$$

Supposing an equiprobable source, and by substituting equations (4.7) and (4.10) into Eq. (4.26), the above expression can be written as:

$$H(X \mid Y) = -\sum_{a=0}^{2^{M_q}-1} \sum_{b=0}^{2^{M_q}-1} c \frac{\alpha}{2} \frac{\exp\left(-\alpha \left| d_{a-b} \right|\right)}{L_{d_{a-b}}} \log_2 \left(2^{M_q - 1} c \frac{\alpha \exp\left(-\alpha \left| d_{a-b} \right|\right)}{L_{d_{a-b}}} \right). \qquad (4.27)$$

4.4.2 Transmission over a binary symmetric channel

In case of a transmission over a BSC with crossover probability q, the compression bound becomes [AAR02-1]:

$$H_{BSC}(X \mid Y) = \frac{H(X \mid Y)}{C(q)}, \qquad (4.28)$$

where $H(X \mid Y)$ is obtained from Eq. (4.27) and $C(q)$ is the capacity of the BSC defined as:

$$C(q) = 1 + q \log_2(q) + (1-q) \log_2(1-q). \qquad (4.29)$$

4.4.3 Transmission over an AWGN channel

To derive the theoretical compression bound of the practical pixel-domain Wyner-Ziv coding system of Figure 4.1, we consider (in Figure 4.6) a discrete source Z being transmitted over an AWGN channel to yield a received discrete sequence X. In addition, side information Y is available at the decoder in such a way that the residual difference between Z and Y exhibits a

Laplacian distribution. Since variables *X*, *Y* and *Z* are quantized representations of the video frame pixels, we further assume the presence, in the proposed model, of a hard decision device at the channel output. In this section, $\{i_1, i_2, ..., i_{M_q}\}$ will denote the binary representation of a quantized pixel *i*, i_1 being the most significant bit.

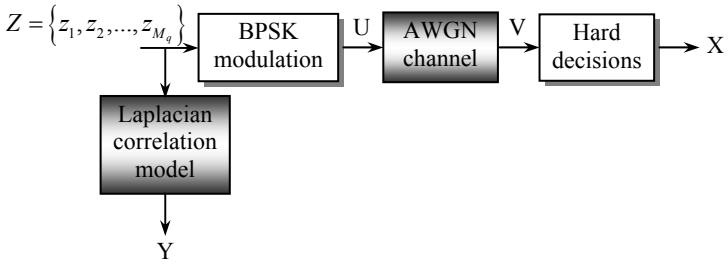


Figure 4.6 Theoretical model for the derivation of the system compression limit in the presence of AWGN noise.

In the absence of AWGN, the system compression limit would be the conditional entropy $H(Z|Y)$ as expressed in Eq. (4.27). Due to the presence of AWGN, the uncertainty about the transmitted source increases and the new compression limit becomes $H_{AWGN}(X|Y)$, defined as in Eq. (4.26).

Let *N* be the additive noise vector, $N = \{n_1, n_2, ..., n_{M_q}\}$, where n_k has a normal distribution with zero mean and variance σ^2. The conditional probability of *X* knowing Y can be calculated by:

$$\begin{aligned} P(X = i | Y = j) &= \sum_{k=0}^{2^{M_q}-1} P(Z = k, X = i | Y = j) \\ &= \sum_{k=0}^{2^{M_q}-1} P(Z = k | Y = j) P(X = i | Z = k, Y = j). \end{aligned} \tag{4.30}$$

Since the additive noise is independent from the source bits, the above expression can be written as:

$$P(X = i | Y = j) = \sum_{k=0}^{2^{M_q}-1} P(Z = k | Y = j) P(X = i | Z = k), \tag{4.31}$$

where $P(Z|Y)$ is defined in Eq. (4.10). The noise samples being independent, we can write:

$$P\left(X=i \mid Z=k\right)=\prod_{r=1}^{M_q} P\left(x_r=i_i \mid z_r=k_r\right) \tag{4.32}$$

On the other hand,

$$\begin{cases} P\left(x_r=1 \mid z_r=0\right)=P\left(n_r>1\right)=P_N \\ P\left(x_r=0 \mid z_r=0\right)=P\left(n_r<1\right)=1-P_N, \end{cases} \tag{4.33}$$

where P_N is defined in Eq. (3.6).

Similarly,

$$\begin{cases} P\left(x_r=0 \mid z_r=1\right)=P\left(n_r<-1\right)=P_N \\ P\left(x_r=1 \mid z_r=1\right)=P\left(n_r>-1\right)=1-P_N. \end{cases} \tag{4.34}$$

Therefore,

$$\begin{aligned} P\left(x_r=i_r \mid z_r=k_r\right) &= \begin{cases} P_N & , \text{if } i_r \neq k_r \\ 1-P_N, & \text{if } i_r=k_r \end{cases} \\ &= \left(1-P_N\right)\delta_{i_r-k_r}+P_N\left(1-\delta_{i_r-k_r}\right), \end{aligned} \tag{4.35}$$

where $\delta_{i_r-k_r}$ is defined as:

$$\delta_{i_r-k_r}=\begin{cases} 1, & \text{if } i_r=k_r \\ 0, & \text{if } i_r \neq k_r. \end{cases} \tag{4.36}$$

Finally, by substituting the equations (4.31), (4.32) and (4.35) into the expression of $H_{AWGN}(X \mid Y)$, we obtain:

$$\begin{aligned} H_{AWGN}\left(X \mid Y\right)=-\sum_{i=0}^{2^{Mq}-1}\sum_{j=0}^{2^{Mq}-1}\sum_{k=0}^{2^{Mq}-1} \\ \left\{\begin{array}{l} \left\{ c\frac{\alpha}{2}\frac{\exp\left(-\alpha\left|d_{i-j}\right|\right)}{L_{d_{i-j}}}\prod_{r=1}^{M_q}\left[\left(1-P_N\right)\delta_{i_r-k_r}+P_N\left(1-\delta_{i_r-k_r}\right)\right]\right\} \\ \cdot\log_2\left(2^{M_q-1}c\frac{\alpha\exp\left(-\alpha\left|d_{i-j}\right|\right)}{L_{d_{i-j}}}\prod_{r=1}^{M_q}\left[\left(1-P_N\right)\delta_{i_r-k_r}+P_N\left(1-\delta_{i_r-k_r}\right)\right]\right) \end{array}\right\}. \end{aligned} \tag{4.37}$$

An estimate of Eq. (4.37) can be obtained with a simple expression as in Eq. (4.28), by mapping the real AWGN channel to a virtual BSC using an equivalence of the stability functions of the two channels [CHU00]. This mapping was adopted in [LIV02] for the design of source-channel codes. In this case, the relationship between the BSC crossover probability p and the average symbol energy per noise density ratio (E_s/N_0) can be written as:

$$p = \frac{1}{2}\left(1 - \sqrt{1 - \exp\left(-2\frac{E_s}{N_0}\right)}\right). \tag{4.38}$$

Therefore,

$$H_{AWGN}(X \mid Y) \approx \frac{H(Z \mid Y)}{C(p)}. \tag{4.39}$$

4.4.4 Application of the theoretical compression bound in broadcast DVC

As mentioned earlier, in the case of a broadcasting application, the system must rely on a pre-determined amount of parity bits, instead of using the feedback channel repeatedly. We can estimate the compression ratios of the video frames using the theoretical compression limits derived in the previous sections. For a fixed transmission rate (for example, the average bit rate D_t obtained by the system with feedback), the theoretical limits can be used by the encoder to determine a quasi-optimal partitioning scheme of the total amount of parity bits between the different frames. In this case, the transmission rate for a particular frame f will be:

$$D_f = D_t \cdot H_f / H_t,$$

where H_f is the lower compression bound for frame f and H_t is the average of the compression limits for all the frames in the sequence.

Note that in this section, we only mention a simple example for applying our analytical study in broadcast DVC. More advanced and practical applications will be presented in the next chapter.

4.5 Simulation results

In order to validate our previous calculations through practical results, we first simulated the system in Figure 4.1 by considering 101 frames from the Carphone sequence and 361 frames from Foreman. The GOP size is fixed to 2 (each Intra-coded key frame is followed by one WZ frame), which means that the number of encoded WZ frames are respectively 50 and 180 for Carphone and Foreman. The interpolation technique used to generate the side information is AVI, the transmission channel is an AWGN, and the compression rate ρ is varied between 0 (no parity bits transmitted) and 0.5 (all parity bits transmitted), $\rho \in \{k/32; k = 0,1,2,...,16\}$.

In Figure 4.7, Figure 4.8 and Figure 4.9, we represent the achievable compression rate (R: xdB) obtained by simulating the source-coding system of Figure 4.1 with feedback, as well as the theoretical lower bound (H: xdB) obtained (as developed in section 4.4) for different values of E_S/N_0 and M_q. The curves labeled 'R' and 'H' are obtained in the absence of noise, whereas the label 'H: xdB (M)' designates the theoretical compression bound estimated by the channel mapping method (Eq.(4.39)). Note that in all the figures, the theoretical bounds are normalized (divided by M_q) so that the unit on the vertical axis becomes similar to the unit of the compression rate (bit per input bit).

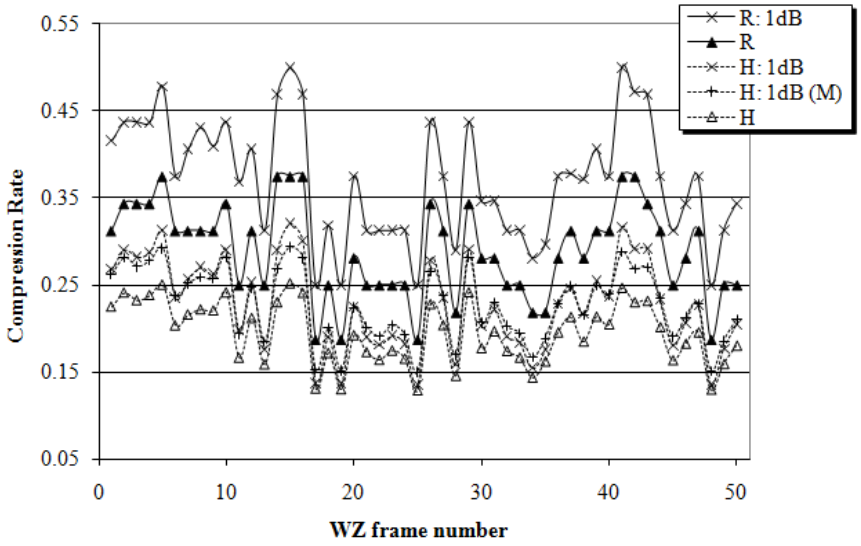


Figure 4.7 Achievable compression rates and theoretical limits for $M_q = 4$ (Carphone).

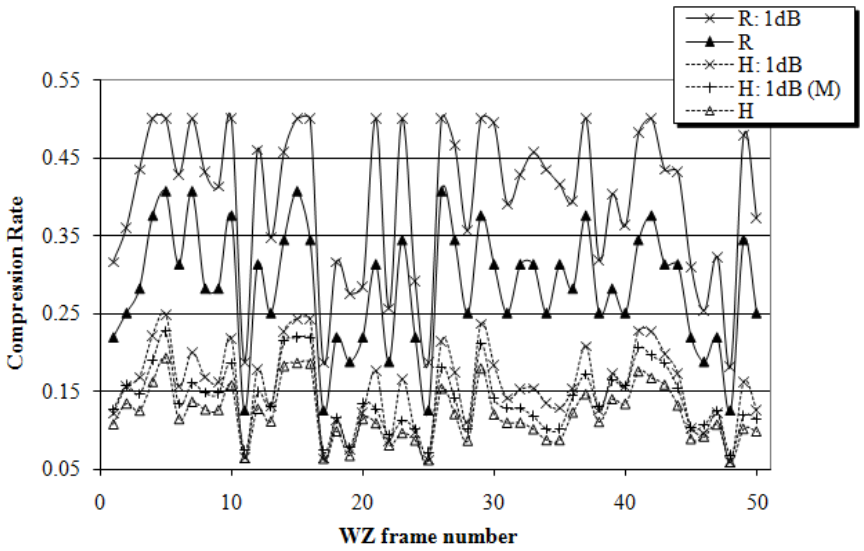


Figure 4.8 Achievable compression rates and theoretical limits for M_q = 1 (Carphone)

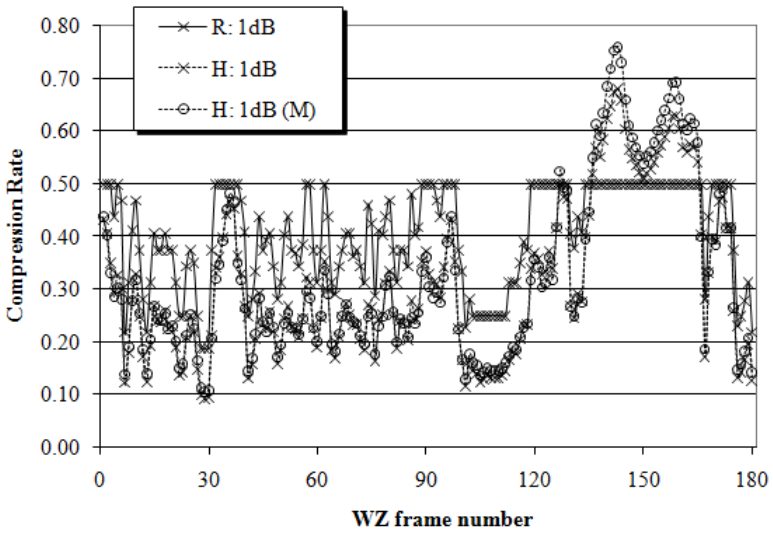


Figure 4.9 Achievable rates and theoretical limits for M_q = 4 (Foreman)

The results obtained for the Carphone sequence with M_q = 4 show a gap in the achieved compression towards the theoretical limit between 0.06 and 0.14 for the case of noiseless transmission. In the case where E_S/N_0 = 1 dB, the gap range increases to [0.11 ; 0.18]. The high values of the gap, of the achievable compressions and of the theoretical limits correspond to the low values of the parameter α measured for M_q = 4 (Figure 4.10). In fact, a low value of α indicates a high level of motion in the frame, making the corresponding side information less reliable for the turbo-decoding process. In this case, a greater amount of parity information is required from the encoder. Starting from E_S/N_0 = 3 dB, the theoretical limits for both noisy and noiseless transmission become almost similar, whereas the practical

system performance becomes almost identical to the noiseless case for $E_S/N_0 \geq 5$ dB. Similar effects were observed for $M_q = 2$ and $M_q = 1$.

However, at $E_S/N_0 = 1$ dB, the gap range increases to [0.14 ; 0.25] for $M_q = 2$ and [0.12 ; 0.33] for $M_q = 1$ (Figure 4.8). The high values observed in the practical compression levels, for both cases of noisy and noiseless transmissions, are due to the fact that when the number of quantization levels is low, the correlation between the quantized pixels of the even frames and those in the interpolated frames decreases dramatically. This leads to important fluctuations in the achievable compression levels from a frame to another and explains the wider ranges in the performance gaps compared to those obtained for $M_q = 4$.

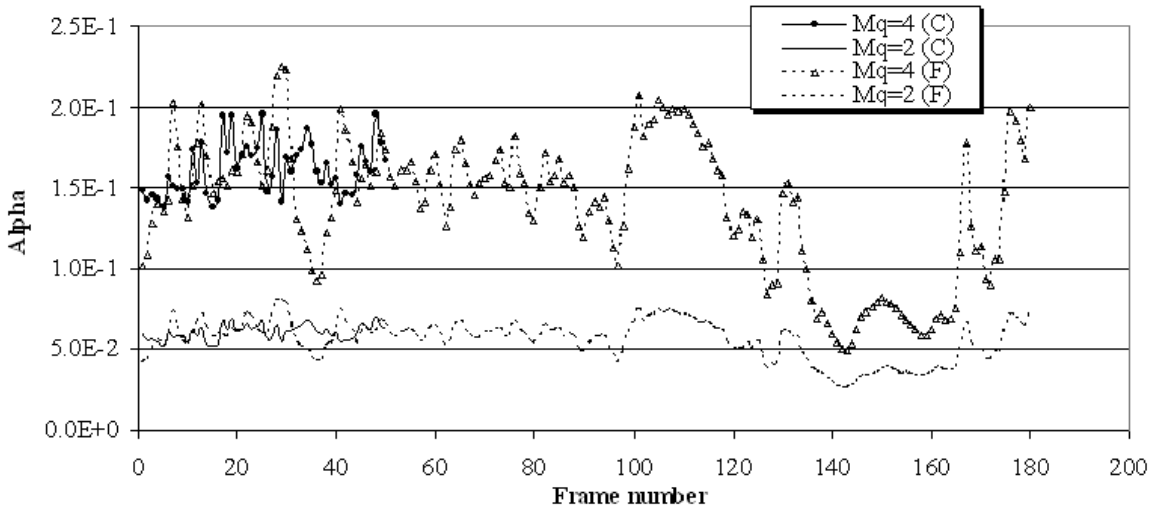


Figure 4.10 The alpha parameter of the Laplacian distribution for the Carphone (C) and Foreman (F) sequences.

In the case of the Foreman sequence, the level of motion throughout the sequence is much more important than for the case of Carphone (note the very low values of the alpha parameter for a certain number of frames in Figure 4.10). In fact, as seen in Figure 4.9, the maximum gap in the system compression performance is very important and reaches almost 0.2 for $M_q = 4$. Furthermore, the entropy limit is sometimes higher than the system's maximum compression limit (0.5), as in frames 136 to 166. In this case, the decoding bit error rate will saturate at higher values than the 10^{-3} target, leading to a more important distortion than for the rest of the sequence.

On the other hand, we clearly notice that, in general, the theoretical entropy limits obtained by Eq. (4.37) are very close to the approximate calculation in Eq. (4.39). In fact, the latter is

less accurate when the compression limits are high, especially for low values of E_S/N_0, i.e. for high values of the mapped BSC crossover probability. However, the closeness between the results of the two estimation methods proves the validity of our theoretical model. Moreover, we notice that the variations of the compression ratio throughout a sequence, in the presence of a feedback channel, are comparable to the variations of theoretical bounds. This similarity in the behavior of the practical and theoretical curves will be exploited by our proposed rate allocation technique for feedback suppression in broadcasting applications.

Finally, in Figure 4.11 and Figure 4.12, we show the average PSNR as a function of the average transmission bit rate (RD curves), obtained for different values of E_S/N_0. The bit rates correspond to 15 WZ frames per second. The theoretical limits (labeled 'TL') are estimated by considering only the influence of quantization, i.e. we assume perfect recovery of the M_q-bit frame at the input of the reconstruction block. For the Carphone sequence quantized at $M_q = 4$, we find a gap between the theoretical and achieved bit rate from 145 kbps (for a noiseless transmission) to 218 kbps (for E_S/N_0 = 1 dB). For M_q = 2, the gap is almost 150 kbps at E_S/N_0 = 1 dB and it decreases to 91 kbps for M_q = 1. The loss in bit rate in the practical compression system due to noise, measured for M_q = 4, is 13 kbps for E_S/N_0 = 5 dB, 46 kbps for E_S/N_0 = 3 dB, and reaches 124 kbps for E_S/N_0 = 1 dB. In the case of a broadcasting application (curves labeled 'B'), for the same data rate, a loss in the PSNR between 0.85 and 1.1 dB is noticed for M_q = 2 and 4, towards the system with a feedback channel. By observing the performance of the broadcasting system at low data rates, we conclude that, in the presence of high noise levels, it is preferable for the decoder to rely only on the side information, since the transmitted parity information does not permit a noticeable enhancement in the PSNR.

In the case of the Foreman sequence, the performance gap towards the theoretical limits is between 100 and 140 kbps for M_q = 4. However, a loss in the PSNR between 2 and 3 dB is noticed due to the high motion in a great part of the sequence. This important loss in performance will be alleviated in the next chapters by adopting the MCI technique for side information generation in the receivers, instead of the simple AVI. We also estimated the system performance for the case where the frames corresponding to a theoretical compression limit higher than 0.5 are dropped at the transmitter and replaced by their corresponding side information in the receiver (curves labeled 'D'). This technique permits a considerable gain in the RD performance: for M_q = 4, a gain in the bit rate between 93 and 127 kbps is observed with a slight increase in the decoding PSNR. Furthermore, when a broadcasting application is

considered (curves labeled 'D-B'), the loss in PSNR is only 0.3 dB at E_S/N_0 = 1 dB and reaches 1 dB at E_S/N_0 = 5 dB. For the Foreman sequence, this system permits a transmission at data rates very close to the theoretical bound, but with a loss in PSNR that can reach 3.4 dB.

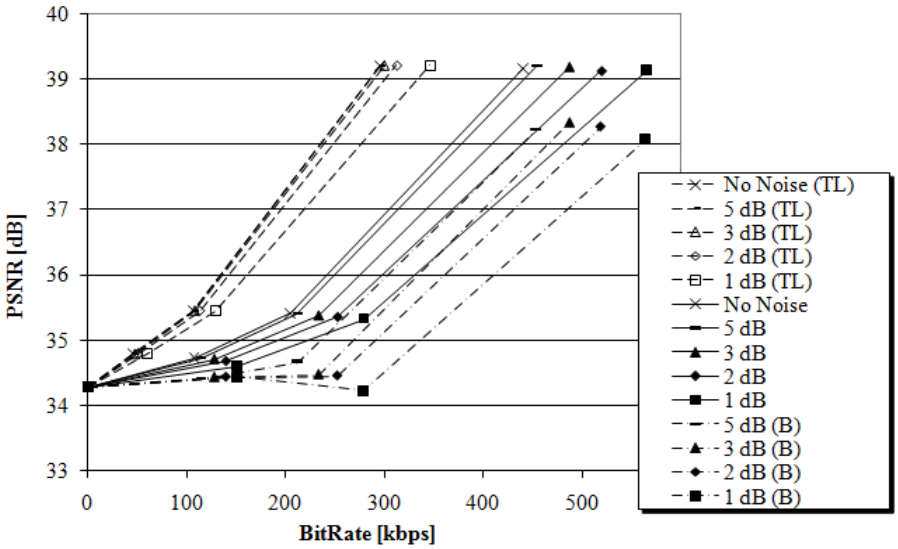


Figure 4.11 RD curves for the Carphone sequence.

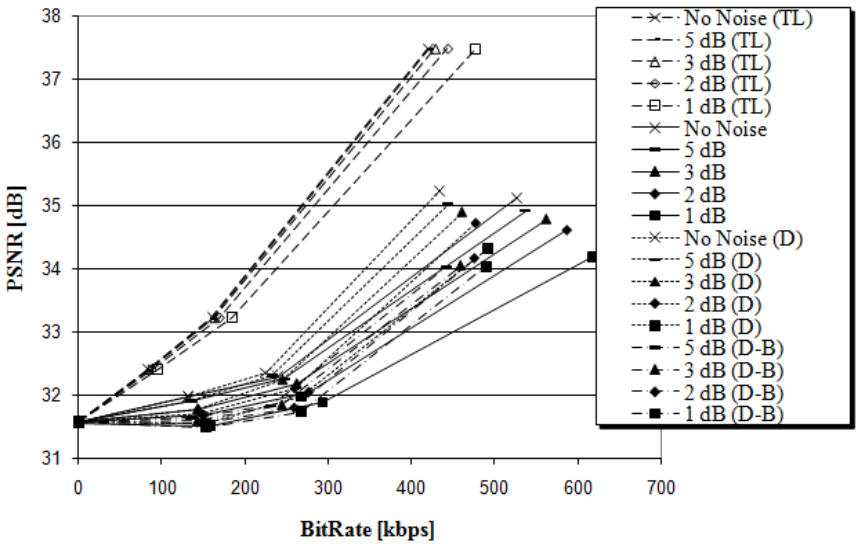


Figure 4.12 RD curves for the Foreman sequence.

4.6 Conclusion

In this chapter, we detailed the different operational blocks of a pixel-domain distributed video codec based on quadri-binary turbo-codes. We also presented an information-theoretic

approach for estimating the system compression limits for the case of a transmission over error-prone channels, as well as for error-free transmission. Simulation results showed a behavior of the practical compression system comparable to the theoretical results; the achievable compression limit for a frame is highly dependent on its content and on the channel conditions. Using the analytical results, the compression level can be predicted for each frame and used by the encoder in a broadcasting system, with a minor loss in the decoding PSNR, compared to the classical feedback-based coding system.

In the next chapter, we will consider practical applications of the WZ compression bound studied in this chapter, mainly for multiuser scenarios and automatic GOP size selection.

Appendix A:
Proof for the relationship between L_{d_Δ} and Δ

Let M represent the number of quantization bits per pixel, and $L+1$ the number of quantization bins: $L = 2^M - 1$.

Let L_{d_Δ} be the number of integer couples (i, j) that yield the residual difference $\Delta = i - j$, where $0 \le i \le L$, $0 \le j \le L$, and $d_\Delta = 2^{8-M}\,\Delta$. Therefore, Δ can take any integer value between $-L$ and L.

In the following, the symbol # will be used to denote "the number of". For example:

L_{d_0} = # $(i, j; i = j)$ = # possible values for i or $j = L + 1 = 2^M$.

L_{d_1} = # $(i, j; i = j + 1)$. Therefore:

For $j = 0$, $i = 1$.

$j = 1$, $i = 2$.

$j = 2$, $i = 3$.

…

$j = L - 1$, $i = L$.

As a result, $L_{d_1} = L = 2^M - 1$. By interchanging i and j we obtain $L_{d_{-1}} = L = 2^M - 1 = L_{d_1}$.

Similarly, L_{d_2} = # $(i, j; i = j + 2)$. Therefore:

For $j = 0$, $i = 2$.

$j = 1$, $i = 3$.

$j = 2$, $i = 4$.

…

$j = L - 2$, $i = L$.

As a result, $L_{d_2} = L - 1 = 2^M - 2$ and $L_{d_{-2}} = L_{d_2}$.

A similar count for all possible values of d_Δ yields: $L_{d_\Delta} = 2^M - |\Delta|$.

Chapter 5

Applications of Feedback Channel Suppression in Wyner-Ziv Video Coding

5.1 Introduction

In the previous chapter, we presented an information theoretic approach for estimating the lower compression bound of Wyner-Ziv frames. Simulation results showed that the behavior of the practical system compression bounds is comparable to the analytical calculations, therefore allowing the suppression of the feedback channel in broadcasting applications. In this chapter, we first apply our previous study in the context of a multiuser system: a dynamic rate allocation algorithm is developed for this purpose and then enhanced using an adaptive quantization and a frame dropping mechanism. Also based on our analytical approach for the estimation of compression bounds, we then develop novel algorithms for dynamically varying the GOP size in distributed video coding.

5.2 Multiuser Application with Adaptive Rate Allocation and Quantization

As discussed in Chapter 4, several attempts have been made in the literature to eliminate or reduce the use of the return channel. Artigas and Torres [ART05] and Morbée et al. [MOR07] proposed techniques that rely on performance tables used by the encoder to predict the compression level of each particular frame. Kubasov et al. proposed in [KUB07-2] an encoder rate control technique that reduces the use of the return channel in a feedback-based DVC codec. Transform domain WZ rate control algorithms were introduced in [BRI07] and [TON06], for DCT and wavelet-based WZ codecs, respectively. However, none of these studies considers the problem of rate allocation for a multiuser scenario, under rate constraints. Additionally, the influence of the channel impairments is not taken into account in the proposed solutions.

Consider the DVC codec presented in Chapter 4, with MCI as the interpolation technique used to generate the side information at the decoder. In this section, we will use the theoretical compression bound derived in the previous chapter to design a rate allocation algorithm for a multiuser application. The proposed technique takes into account the amount of motion in the captured video scene as well as the transmission channel conditions for every user, in order to allocate unequal transmission rates among the different users. On the other hand, the total transmission rate for all users does not exceed a fixed, maximum allowable rate imposed by the limited available bandwidth in such systems. Furthermore, the quantization parameter (i.e. the number of quantization levels) is dynamically varied for each frame, at every user, in such a way to optimize the decoded video quality. A frame dropping mechanism is also used in order to avoid unnecessary channel use, when the analytical estimations show that successful decoding of a given WZ frame will not be possible in the receiver, because of very high motion and/or bad channel conditions.

5.2.1 Dynamic rate allocation

Consider a network of N video users as shown in Figure 5.1. In practical applications, these users can be camera-equipped mobile phones, each capturing a different scene, or wireless surveillance cameras capturing the same scene from different viewpoints. Each user transmits the video data to a central base station through a different wireless channel. Let R denote the total allowable transmission rate for all users, expressed in bits per second (bps), $\rho_{f,n}$ the compression rate for frame f at node n defined as in Section 4.3.4 (i.e. the ratio of the number of parity bits at the turbo-encoder output, after puncturing, over the number of systematic bits), $M_{f,n}$ the quantization parameter for this frame and $A_{f,n} = M_{f,n}\rho_{f,n}$ its average number of bits per pixel.

The base station performs rate control based on each user's transmission conditions and on the amount of movement in each video scene, as will be detailed next.

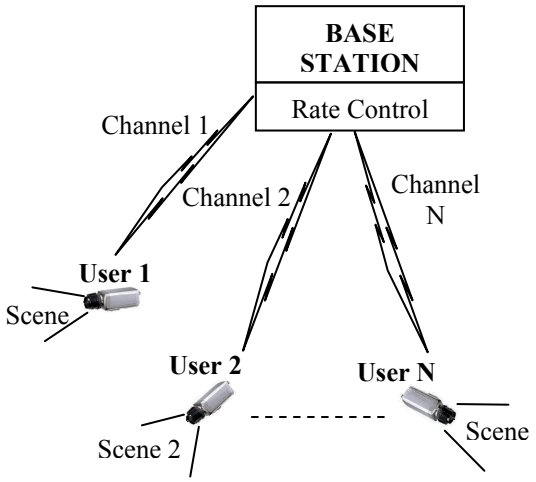


Figure 5.1 Network of Wireless Video Sensors.

We model the transmission channel between each user and the base station by a binary symmetric channel (BSC) with a transition probability q. In fact, a different channel model (i.e. additive white Gaussian noise, Rayleigh fading, etc…) can also be used in our system by mapping the channel parameters to a BSC crossover probability, as detailed in [CHU00]. In this case, the theoretical compression bound for frame f at the user n can be obtained from Eq. (4.28) and expressed as:

$$H_{f,n} = -\frac{1}{C(q)} \sum_{a=0}^{\left(2^{M_{f,n}}-1\right)} \sum_{b=0}^{\left(2^{M_{f,n}}-1\right)} c\frac{\alpha}{2}\frac{\exp\left(-\alpha\left|d_{a-b}\right|\right)}{L_{d_{a-b}}} \log_2\left(2^{M_{f,n}-1} c\frac{\alpha\exp\left(-\alpha\left|d_{a-b}\right|\right)}{L_{d_{a-b}}}\right), \quad (5.1)$$

where $C(q)$ is the capacity of the BSC.

A traditional (TRD) system would allocate R/N (bps) for each user. This un-optimized rate allocation technique would not be fair if the users were transmitting different video contents to the central station over different channel conditions. In fact, a rate of R/N may not be sufficient to correctly decode some of the WZ frames and may be more than enough for some others. By first calculating Eq. (5.1) for each sensor, the system can determine the compression rate for each frame based on its content (i.e. the parameter α) and on the sensor transmission conditions (channel crossover probability q). Therefore, the system will allocate for sensor n, in a proportionally fair attribution, the rate:

$$R_{f,n} = \frac{H_{f,n}}{\sum_{n=1}^{N} H_{f,n}} R\,. \quad (5.2)$$

If the channel state does not vary for K consecutive WZ frame and the system has sufficient buffering capacity, the above-mentioned rate allocation technique may further be improved by allocating:

$$R_{f_i,n} = \frac{H_{f_i,n}}{\sum_{n=1}^{N}\sum_{j=1}^{K} H_{f_j,n}} R, \tag{5.3}$$

where f_i represents the i^{th} frame within the selected group of K consecutive WZ frames. It is clear that in this second case, the instantaneous total transmission rate is variable within the group of K frames, with a global average R. For K=1, Eq. (5.3) reduces to Eq. (5.2).

5.2.2 Adaptive quantization and frame dropping mechanism

The frame rate $R_{f,n}$, estimated using Equation (5.2) or (5.3), can be achieved using different combinations of the parameters $M_{f,n}$ and $\rho_{f,n}$. Therefore, it is important to determine, for each frame f at every user n, the couple $(M_{f,n}, \rho_{f,n})$ that optimizes the average system performance. In other words,

$$\left\{\left(M_{f,n}, \rho_{f,n}\right); n = 1,..., N\right\} = \underset{\substack{M_{f,n}\in\{1,2,4\} \\ 0\le\rho_{f,n}\le 1/2}}{\arg\max}\left(\frac{1}{N}\sum_{n=1}^{N} PSNR_{f,n}\right), \tag{5.4}$$

where $PSNR_{f,n}$ is the Peak Signal-to-Noise Ratio obtained after the decoding and reconstruction of the frame f encoded at the user n. It depends not only on the parameters $\rho_{f,n}$ and $M_{f,n}$, but also on the transmission conditions, the accuracy of the side information and the rate constraint:

$$\sum_{n=1}^{N} R_{f,n} \le R. \tag{5.5}$$

As a result, it is difficult to determine analytical solutions (i.e. the set $\left\{\left(M_{f,n}, \rho_{f,n}\right); n = 1,..., N\right\}$) for the combinatorial problem defined by equations (5.4) and (5.5). For this reason, we proceed with the optimization process in two stages. First, we determine the transmission rate for each user, and then we choose the couples $\left(M_{f,n}, \rho_{f,n}\right)$ that yield the best reconstructed output at the specified bit rates.

In the first stage, since the optimal quantization parameter has not been determined yet, $R_{f,n}$ is computed using Eq. (5.2) or Eq. (5.3), for $M_{f,n} = 8$, assuming 8-bit raw video data before compression.

The average number of bits per pixel is related to $R_{f,n}$ by:

$$A_{f,n} = \frac{R_{f,n}}{m \cdot n \cdot r}, \tag{5.6}$$

where (*m*,*n*) represent the dimensions of a video frame and *r* the WZ frame rate, in frames per second (fps).

In the second stage, the base station needs to determine, for each frame *f* at every node *n*, the couple $\left(M_{f,n}, \rho_{f,n}\right)$ that yields the best video output after reconstruction, at the specified rate $R_{f,n}$. In fact, this is equivalent to optimizing the individual rate-distortion performance for each user since, for a given bit rate, the parameters are chosen in such a way to maximize $PSNR_{f,n}$.

After a thorough analysis of the system performance observed for different values of $M_{f,n}$, we noticed that in most cases, for a given target bit rate, choosing the lowest allowable value of $M_{f,n}$ yields the best video quality at the decoder output. Indeed, by reducing the number of quantization levels, the system is able to transmit a greater amount of parity bits to protect the quantized bit stream from channel errors, especially when *q* increases. However, in some cases, the assigned bit rate is sufficient enough to permit efficient error protection when a greater value of $M_{f,n}$ is selected, thus, a better reconstructed output can be obtained. In all cases, we noticed that the system behavior for different configurations of the couples $\left(M_{f,n}, \rho_{f,n}\right)$ is directly related to the ratio between $A_{f,n}$ and the theoretical compression bound defined as:

$$C_{f,n}(M_{f,n}) = \frac{A_{f,n}}{H_{f,n}(M_{f,n})} \tag{5.7}$$

Therefore, we define the thresholds T_1, T_2 and T_4 which indicate the average value of the ratio $C_{f,n}(M_{f,n})$ that permits a correct decoding of a transmitted frame for $M_{f,n}$ = 1, 2 and 4

respectively. These thresholds are determined experimentally by observing the system performance for different values of the ratio $C_{f,n}(M_{f,n})$, as will be detailed later. Our proposed algorithm then proceeds with the dynamic quantization (Figure 5.2) as follows:

Step 1: Initially, set $M_{f,n}$ to the lowest value that permits to reach $A_{f,n}$. This will allow for the maximum error protection for a given $A_{f,n}$.

Step 2: Calculate $C_{f,n}(1)$, $C_{f,n}(2)$ and $C_{f,n}(4)$.

Step 3: If $M_{f,n}=4$ and $C_{f,n}(4)\le T_4$, set $M_{f,n}=2$ and $\rho_{f,n}=1/2$.

In other words, if $A_{f,n}$ could not be reached for $M_{f,n}<4$ (i.e. $A_{f,n}>1$) and the amount of error-protection transmitted with $M_{f,n}=4$ does not yield an acceptable decoding error rate, set $M_{f,n}$ to the next lower value and transmit all parity bits. In this case, the given frame is transmitted at a rate lower than the target bit rate since the target $A_{f,n}$ could not be reached exactly.

Similarly, if $M_{f,n}=2$ and $C_{f,n}(2)\le T_2$, set $M_{f,n}=1$ and $\rho_{f,n}=1/2$.

Step 4: If $M_{f,n}=1$ and $C_{f,n}(1)\le T_1$, drop the frame (set $M_{f,n}=0$).

In this case, the amount of transmitted bits will not permit efficient decoding, even if the number of quantization levels is reduced to its minimum. At the decoder, the dropped frame is replaced by the corresponding side information. Note that other error concealment techniques can also be envisaged in the receiver.

Step 5: If $M_{f,n}=1$ and $C_{f,n}(2)>T_2$, set $M_{f,n}=2$.

In other words, if $A_{f,n}$ is reachable with $M_{f,n}$ = 1 and 2, and it is possible to send a sufficient amount of parity bits to correctly decode the frame with $M_{f,n}=2$, set $M_{f,n}=2$ since it yields a better reconstructed output.

Similarly, if $M_{f,n}=2$ and $C_{f,n}(4)>T_4$, set $M_{f,n}=4$.

Step 6: If the frame was not dropped and $\rho_{f,n}$ was not already set to ½, set $\rho_{f,n} = A_{f,n} / M_{f,n}$.

Step 7: Transmit the couple $\left(M_{f,n}, \rho_{f,n}\right)$ to the corresponding user.

As it can be seen from the proposed algorithm, control information is sent only once from the base station to the users. Moreover, instantaneous decoding (as discussed in section 4.2) at the receiver is no more required, and only one decoding run is performed for each frame. As a result, all the disadvantages related to the return channel in traditional Wyner-Ziv applications are eliminated.

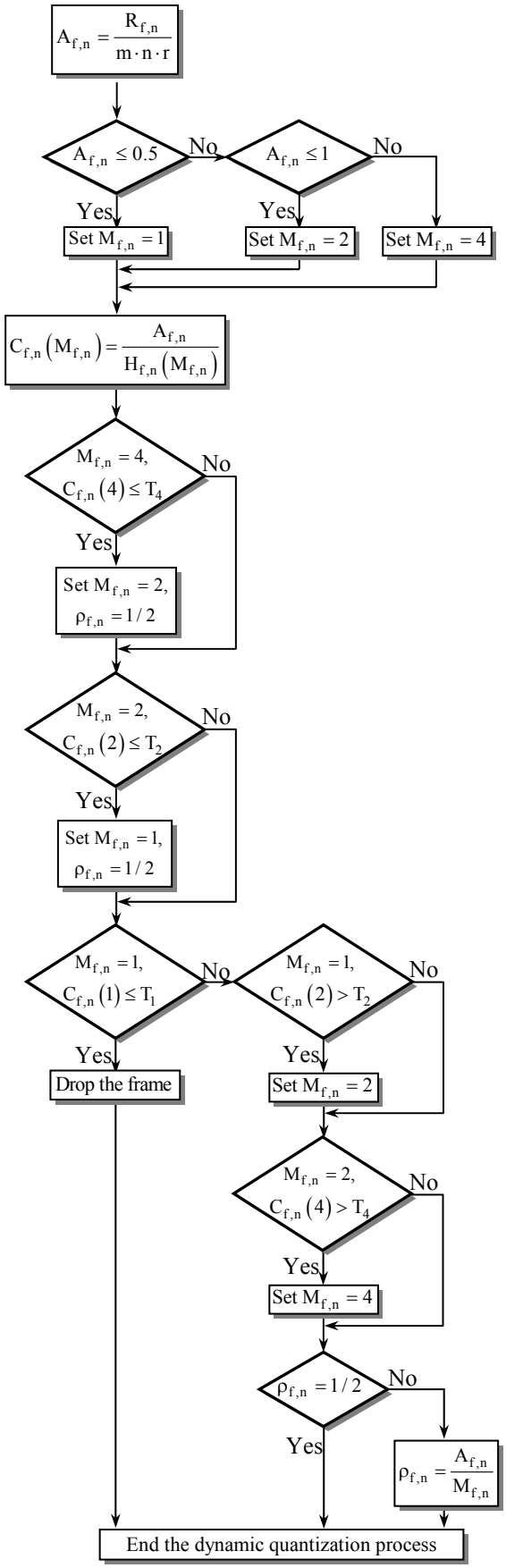


Figure 5.2 Adaptive quantization algorithm with a frame dropping mechanism.

5.2.3 Influence of coded key-frames: a rate-distortion analysis

Up to this point, we considered that key frames are perfectly recoverable at the decoder. However, this is not the case in practical applications. In this section, we consider the influence of coding the key frames on the WZ codec performance. In order to study the DVC rate-distortion behavior, we first analyze the WZ frames decoding BER as a function of the ratio $C_{f,n}$ as shown Figure 5.3, which was obtained by simulating the transmission of 22500 frames from different video sequences with variable channel conditions, for each value of *M*. It is known that for an acceptable system performance, an average BER less than 10^{-3} is desired for the decoded WZ frames. It can be seen, from Figure 5.3, that this condition is satisfied when the ratio $C_{f,n}$ is greater than 3 for $M_{f,n} = 4$, greater than 4 for $M_{f,n} = 2$, and greater than 6 for $M_{f,n} = 1$. As a result, for a BER close to 10^{-3}, the necessary $A_{f,n}$ can be estimated by multiplying the lower compression bound (calculated using Eq. (5.1)) by the corresponding coefficient T_M ($T_4 = 3$, $T_2 = 4$ and $T_1 = 6$).

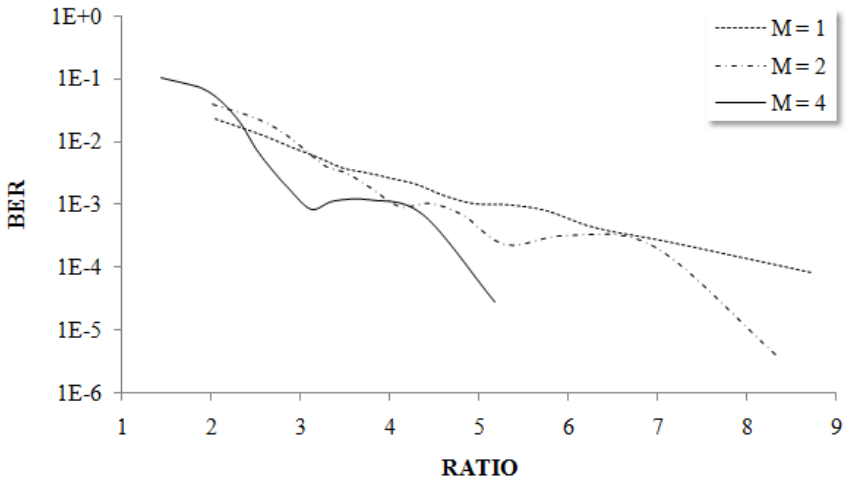


Figure 5.3 Average BER as a function of the ratio $C_{f,n}$.

The next step in the RD analysis is determining the best quantization parameter (QP) of the H.264 intra-coded key frames transmitted in the absence of noise. In fact, when QP increases, a stronger compression is performed on the key frames, which results in lower bit rate requirements, as shown in the plain curves of Figure 5.4. However, a stronger compression of key frames yields less reliable side information. As a result, more parity bits are necessary to successfully decode WZ frames and thus, the WZ bit rate increases (dotted curves in Figure 5.4).

In the following, the WZ quantization parameter will be denoted M, instead of $M_{f,n}$, for simplicity.

Figure 5.5 shows the total bit rate (key + WZ frames) required for different video sequences. As expected from the observations of Figure 5.4, when QP increases, the total bit rate decreases until it reaches a minimum, and then starts increasing. At the same time, the PSNR decreases all the time, as shown in Figure 5.6. In other words, a certain target total rate can be generally reached with two different values of QP, while the best PSNR is obtained for the lowest value of QP. It can be seen from Figure 5.5 that the rate starts increasing for values of QP greater than 30. Therefore, values of QP less than 30 are more suitable for the DVC codec.

In general, a near-constant PSNR is desired in the received video, since great fluctuations in the PSNR values result in an annoying visual effect. It can be observed in Figure 5.6 that the PSNR of WZ frames approaches that of key frames for values of QP between 20 and 30. In fact, similar results were observed with different values of M.

Therefore, it can be concluded that in a practical implementation of the pixel-domain Wyner-Ziv video codec of Chapter 4, QP must not exceed 30 for efficient compression, and must not go below 20 for quasi-constant PSNR.

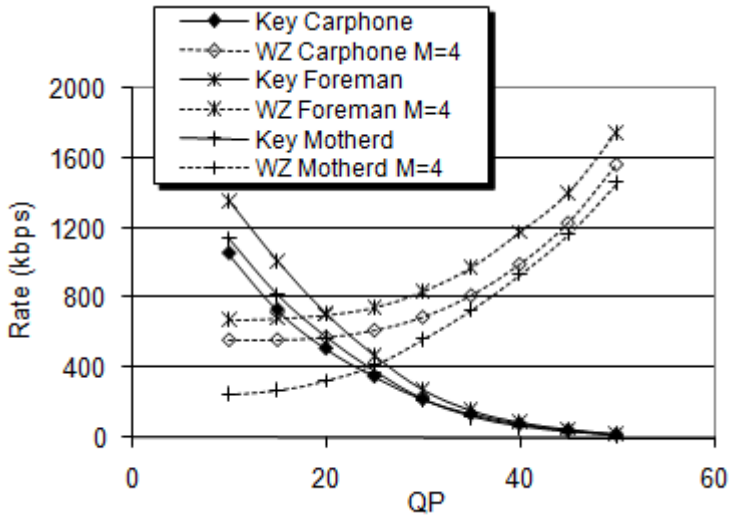


Figure 5.4 Average source coding rate of key and WZ frames with $M = 4$, for Carphone, Foreman, and Mother-Daughter sequences at 30 fps.

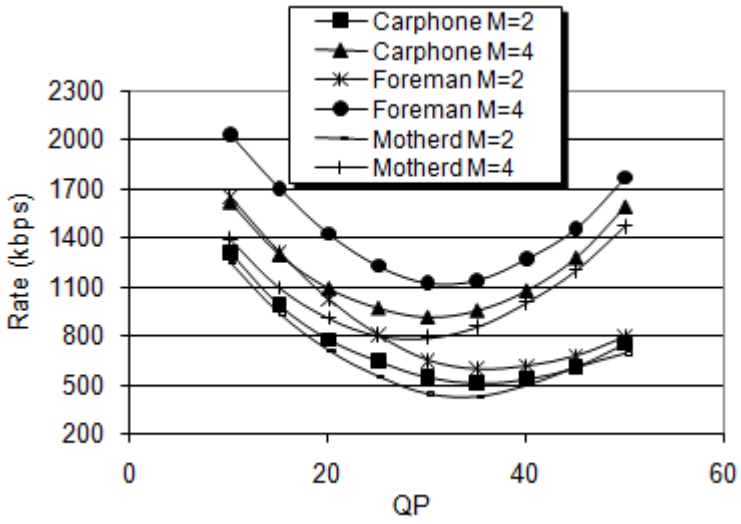


Figure 5.5 Average source coding rate for Carphone, Foreman, and Mother-Daughter sequences at 30 fps, including both key frames and WZ frames.

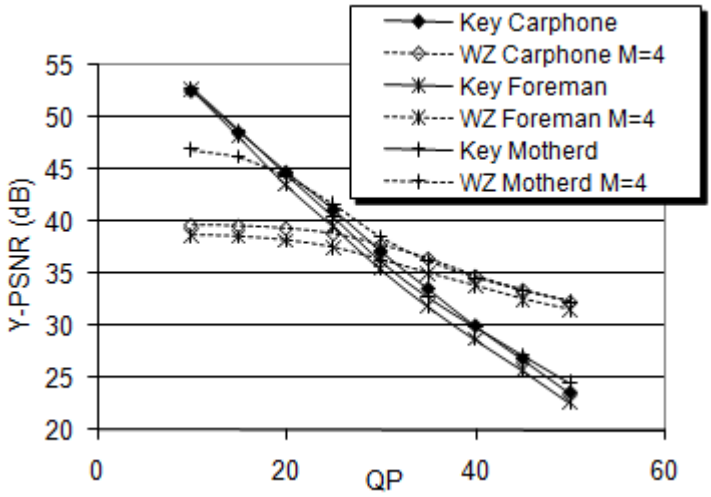


Figure 5.6 Average PSNR of the luminance component for Carphone, Foreman, and Mother-Daughter sequences at 30 fps.

5.2.4 Simulation results of the multiuser scenario

In our simulation setup, we consider a set of three mobile users (N = 3) capturing different scenes and transmitting the resulting video to a central base station. These scenes are assumed to be the Foreman, Carphone, and Mother-Daughter QCIF video sequences. They are sampled at a rate of 15 WZ fps, which corresponds to an overall sampling rate of 30 fps. For example, this can be seen as a network of surveillance cameras in a building, each located in a different floor, or a network of mobile users positioned at different locations in a cell. We

consider the first 100 frames of each sequence repeated in 50 simulations. The side information is generated by MCI as described in the previous chapter. The time-varying nature of the transmission channel between a video user and the base station is modeled by a uniform random variation of the crossover probability q between 0.001 and 0.02, independently for each user.

In Figure 5.7 and Figure 5.8, we first show the RD curves obtained with a traditional (TRD) system where all users are assigned an equal bandwidth, for K = 1 (rate allocation is performed using Eq. (5.2)) and K = 50 (Eq. (5.3)), respectively. The quantization parameter M is fixed and is the same for all users. The results are presented in terms of the PSNR averaged over the three video scenes as a function of the total WZ bit rate occupied by all the users. The rate points correspond to the values of $\rho \in \{0.21875, 0.28125, 0.34375, 0.5\}$ for $M = 1$ and 2, and $\rho \in \{0.21875, 0.28125, 0.34375\}$ for $M = 4$. We also show the results obtained with our adaptive rate allocation (ARA) technique, but with a constant quantization parameter. The target total bit rates are specified to be equal to those of the TRD. We notice that at low bit rates, ARA does not significantly improve the average system performance. This is expected since the available total bit rate R is not sufficient for the N = 3 users to simultaneously transmit video data to the central station. When R increases, ARA yields a better PSNR than TRD. At a shared rate of 1.28 Mbps with M = 4, the gain reaches 1.6 dB when K = 50 and 1 dB when K = 1. Besides, we can clearly see that when the rate regions overlap for different values of M, the best performance, in both ARA and TRD systems, is obtained when the lowest value of M is used. For example, for K = 1 at 570 kbps, the use of a 1-bit quantizer yields a performance gain of nearly 1 dB, compared to the case with a 2-bit quantizer. A similar effect is noticed at 1 Mbps, where the system with a 2-bit quantizer outperforms the one with a 4-bit quantizer by 0.8 dB. Similar results can be observed for K = 50. The only gain observed, towards the case with K = 1, is for R = 1.28 Mbps. Therefore, in order to avoid high buffering requirements in the encoders, we consider that K = 1 in the following study.

In some cases, the performance of a TRD system can be better than ARA. In Figure 5.9, we present the individual PSNR of the different video sequences, for the case where M = 4. In fact, a similar behavior is observed for different values of M. We notice that at a total bit rate of 1 Mbps, the performance loss for the Mother-Daughter (D) sequence is relatively high, which degrades the average ARA system performance, compared to the TRD system, as also shown in Figure 5.7. The remedy for this problem is to reduce the number of quantization

levels for the transmission at this bit rate, as was explained earlier. We also notice that the ARA technique significantly improves the PSNR of the Carphone (C) and Foreman (F) sequences at the expense of a reduced performance for Mother-Daughter. This can be explained by the fact that the Mother-Daughter sequence contains low motion scenes, while Carphone and Foreman are characterized by average and high motion scenes. In fact, the ARA technique allocates the lowest rates for videos with low motion and/or experiencing good channel conditions, while the highest rates are assigned to video users capturing high-motion scenes and/or suffering from a bad channel. This results in an improved average system performance, especially at medium and high total transmission rates.

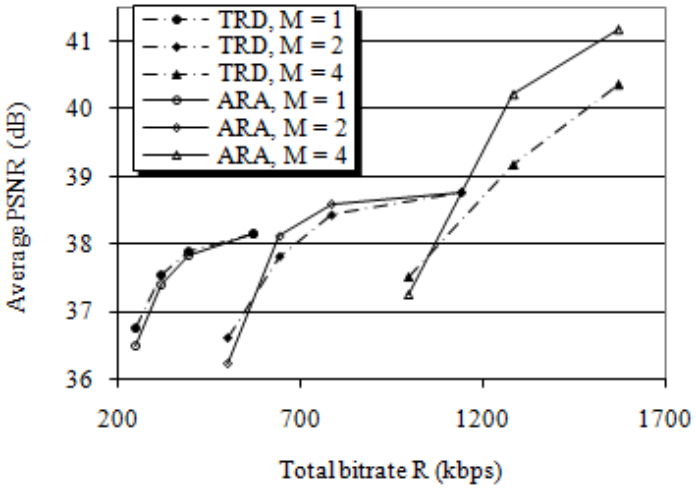


Figure 5.7 RD Curves obtained with a TRD system and with the ARA technique, $K = 1$.

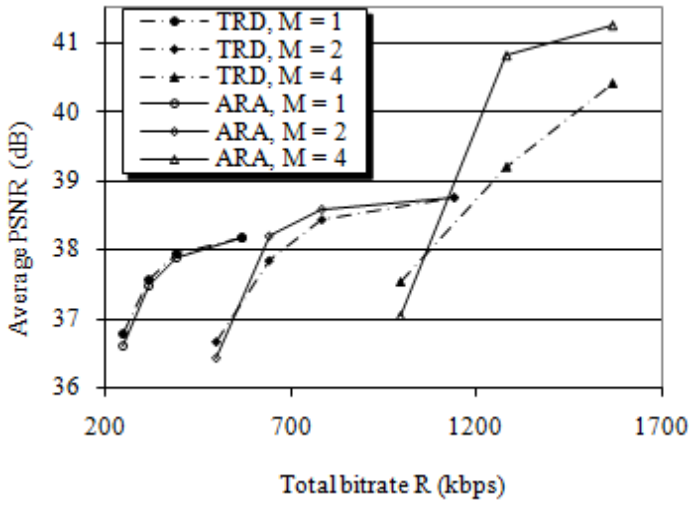


Figure 5.8 RD Curves obtained with a TRD system and with the ARA technique, $K = 50$.

The analysis of the individual PSNR as a function of the total bit rate (as in Figure 5.9) can sometimes be misleading. In Figure 5.10, the PSNR of each video sequence is represented as a function of the effective WZ bit rate assigned for each user, for the case where $M = 4$. Figure 5.10 clearly shows that the RD curves obtained with the ARA technique are above those obtained with a TRD system. This implies that the ARA technique not only improves the average system performance, but the individual RD performance for each user as well, especially at medium and high bit rates.

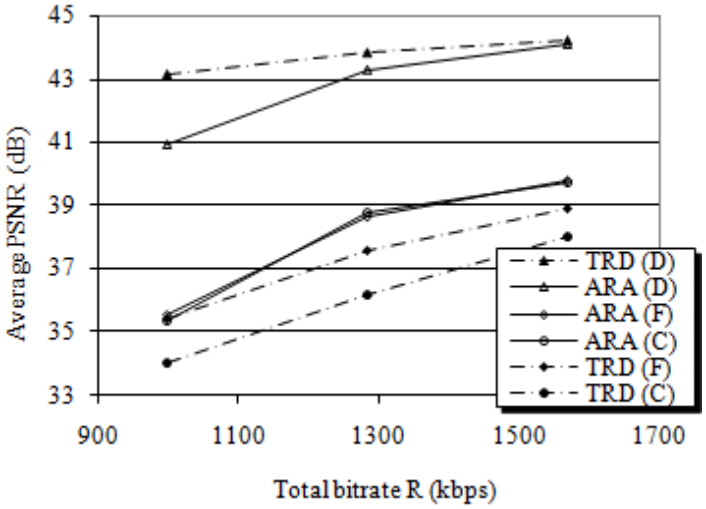


Figure 5.9 Individual PSNR as a function of the total bit rate, for the Carphone (C), Foreman (F) and Mother-Dauther (D) sequences, with $M = 4$.

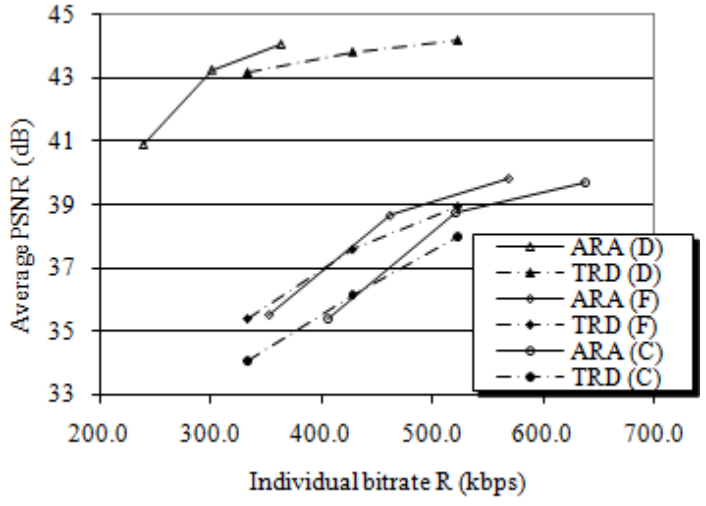


Figure 5.10 Individual RD curves for the Carphone (C), Foreman (F) and Mother-Dauther (D) sequences obtained with $M = 4$.

This effect can be further explained by Table 5.1 which shows the average bit rates (in kbps) assigned for each user with the ARA technique. In the TRD system, all users operate at the same bit rate. Indeed, we notice that in all cases, for a certain total bit rate, the ARA technique assigns the Mother-Daughter sequence the lowest bit rate compared to the two other sequences. For example, at a total rate of 1.5 Mbps, the bit rate assigned for Mother-Daughter is almost half the one assigned for Carphone. Moreover, the bit rate assigned for the Mother-Daughter sequence by the ARA technique is less than the one assigned by the TRD system. This allows the assignment of higher bit rates to the two other sequences and leads to an improvement in the overall system performance.

	Total	***TRD***	***ARA (C)***	***ARA (F)***	***ARA (D)***
M=1	249.5	83.16	80.1	95.5	73.9
	320.8	106.92	103.2	124.4	93.2
	392	130.68	127.6	150.5	113.9
M=2	499	166.32	212.9	168.3	117.7
	641.5	213.84	274.2	219.9	147.4
	784.1	261.36	331.1	274.4	178.6
M=4	997.9	332.64	405.4	353.4	239.2
	1283	427.68	520.3	461.9	300.8
	1568.2	522.72	636.5	568.5	363.1

Table 5.1 Average individual bit rates in kbps for the Carphone (C), Foreman (F) and Mother-Dauther (D) sequences.

In order to apply the dynamic quantization algorithm and the frame dropping mechanism, suitable values for the thresholds T_1, T_2, and T_4 are needed. In Section 5.2.3, the values T_4 = 3, T_2 = 4 and T_1 = 6 were chosen in such a way to yield a decoding BER close to 10^{-3}. In fact, it should be noted that when M = 1 (or 2), only the first (two) MSB in each pixel is (are) coded and transmitted, which greatly affects the performance of the reconstruction function. However, for the case where M = 4, a higher BER can be tolerated, since the two least significant bits in the quantized pixel are less important than the two most significant bits. Therefore, we can reduce T_4 to 2.4 in order to favor decisions to M = 4 at high bitrates in case of good channel conditions and low motion frames. In fact, as it can be deduced from the algorithm in section 5.2.2, if T_4 is reduced, the choice $C_{f,n}(4) > T_4$ is more likely to occur in those transmission conditions. In the sequel, we will show the system performance for these selected values of the thresholds and compare them to the ones obtained with other values.

In Figure 5.11, it can be seen that the problem of overlapping rate regions is avoided by applying dynamic quantization (curves labeled 'ARAQ' and 'ARAQ-D'), in addition to adaptive rate allocation. When no frame dropping is performed (ARAQ: step 4 in the dynamic quantization algorithm is skipped), a performance gain towards the TRD system is observed only at high rates, for $T_2 = T_4 = 2.7$. The reason for this behavior is that, at low rates, M takes the values 1 or 2 most of the time, and $T_2 = 2.7$ does not guarantee a good BER when $M = 2$, as shown in Figure 5.3, whereas for $M = 4$ (mostly at high rates), $T_4 = 2.7$ yields acceptable performance. Compared to the TRD system, very close performances are obtained at low rates by setting $T_2 = 4$ and $T_4 = 2.4$, while an important enhancement is noticed at medium and high rates. In fact, at very low rates, the available bandwidth to be allocated for the different users is barely sufficient to protect the transmitted bit streams. At a rate near 200 kbps, we notice a performance loss of 0.5 dB, whereas a gain of 1.5 dB is observed with the frame dropping mechanism (ARAQ-D curves) for $T_1 = 6$. When the bit rate increases, less frames are dropped. Starting from 600 kbps, the RD curves are the same with or without frame dropping. This is expected since frames are only dropped when the assigned bit rate is not sufficient to guarantee a correct decoding of the transmitted WZ frame. On the other hand, at 500 kbps, our proposed algorithm yields similar performance compared to the TRD system with $M = 1$, but a gain of 1.4 dB is observed towards the case where $M = 2$. At 1 Mbps, the gain reaches 0.6 dB compared to $M = 2$ and 1.5 dB to $M = 4$.

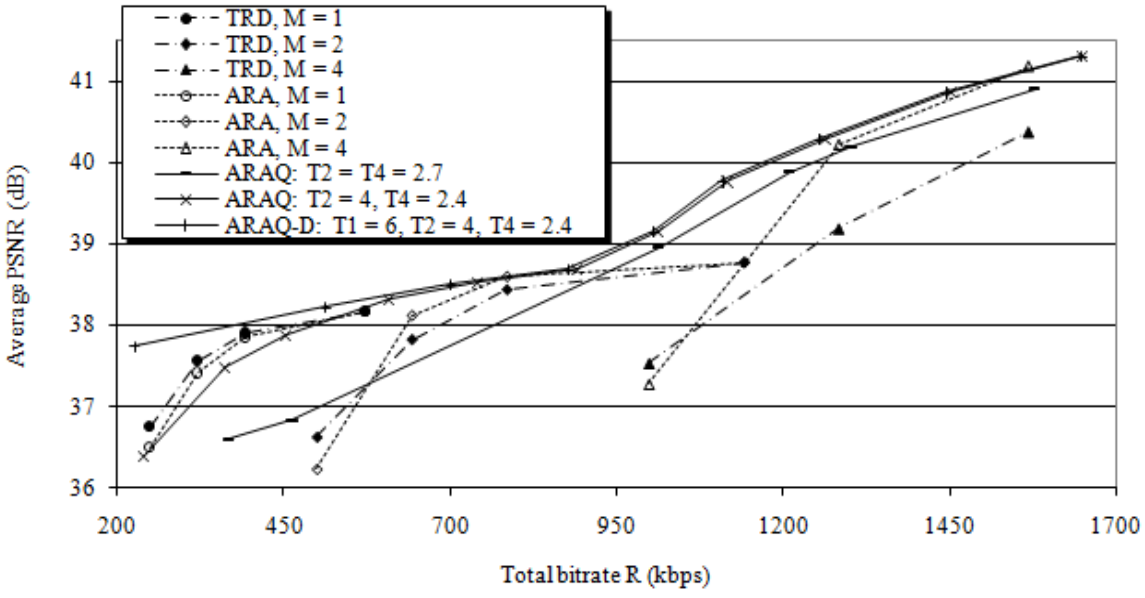


Figure 5.11 RD Curves obtained with a TRD system and with the ARAQ and ARAQ-D techniques.

It is important to note that when M is fixed (TRD and ARA systems), the transmission bit rate for the WZ codec is limited to a narrow range. For example, a traditional system with $N = 3$ cannot transmit at a total rate greater than 570 kbps when $M = 1$, and 1140 kbps when $M = 2$ (assuming QCIF video sequences sampled at 15 WZ-fps). Our proposed algorithm with the ARAQ and ARAQ-D techniques allows the system to transmit video data at a wider range of transmission rates and with an optimized decoding quality.

Figure 5.12, Figure 5.13 and Figure 5.14 show the individual RD performance for the Carphone (C), Foreman (F) and Mother-Daughter (D) sequences respectively, obtained with our ARAQ-D technique with $T_1 = 6$, $T_2 = 4$ and $T_4 = 2.4$. Compared to the TRD system, it can be seen, as explained previously, that the RD performance for each user is improved, especially at high rates. At an individual rate of 430 kbps, the gain reaches 1.5 dB for Carphone, 1.2 dB for Foreman, and 0.6 dB for Mother-Daughter. Again, it can be seen that Mother-Daughter is assigned the lowest bit rates. The maximum bit rate allocated to the Mother-Daughter sequence is 430 kbps, while Carphone and Foreman are assigned a maximum of 615 kbps and 610 kbps respectively.

In Figure 5.15, we show a snapshot from the Carphone video sequence (7^{th} WZ frame) obtained with both the TRD (Figure 5.15-b and Figure 5.15-c) and ARAQ-D (Figure 5.15-d and Figure 5.15-e) systems, for the same transmission conditions ($R_{f,n} = 332.64$ kbps and $q = 6.4\text{E-}3$). The middle column shows the turbo-decoded images before reconstruction and the right column shows the final reconstructed outputs. In the TRD system, the quantization parameter was set to $M_{f,n} = 4$ and the compression rate to $\rho_{f,n} = 0.21875$, whereas the ARAQ-D system determined that $M_{f,n} = 2$ and $\rho_{f,n} = 0.4375$ would yield a better result, even though in both cases $A_{f,n} = 0.875$. In fact, when $M_{f,n} = 4$, more details about the transmitted image are available at the receiver, as shown in Figure 5.15-b compared to Figure 5.15-d. This is supposed to help the reconstruction function in improving the output video quality. However, the received frame was decoded with a BER = 0.09 in the TRD system, giving a noise-like visual effect (Figure 5.15-b), whereas in the ARAQ-D system, the frame was perfectly recovered (BER ≈ 0) with $M_{f,n} = 2$ (Figure 5.15-d). This is due to the fact that, in the latter case, the choice of $\rho_{f,n} = 0.4375$ leads to a better protection (by the turbo-encoder) of the transmitted bit stream. The superior performance of the ARAQ-D system can be observed by comparing Figure 5.15-c and Figure 5.15-e. A visual inspection of both

images shows that the reconstruction function was not able to completely remove the noise-like effect from the decoded image in the TRD system, whereas in the ARAQ-D system, the reconstructed image is visually better. This performance gain can be further observed in the output PSNR: 31.5 dB in Figure 5.15-c and 34 dB in Figure 5.15-e, resulting in a 2.5 dB gain. However, in both cases, the PSNR for this frame is below the average PSNR obtained at the same bit rate and presented in Figure 5.12. This is due to a higher level of motion in this particular frame, compared to the average motion level in the sequence, which further justifies the need for increasing the compression rate $\rho_{f,n}$ and decreasing the quantization parameter $M_{f,n}$.

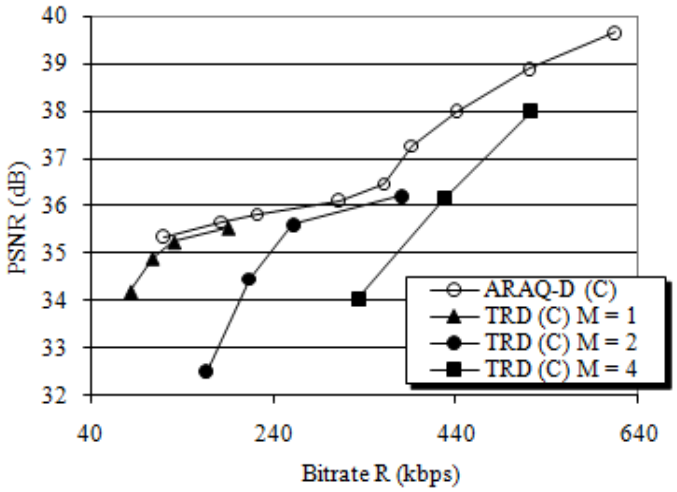


Figure 5.12 RD curves for the Carphone (C) sequence obtained with the ARAQ-D technique, compared to the performance of a TRD system.

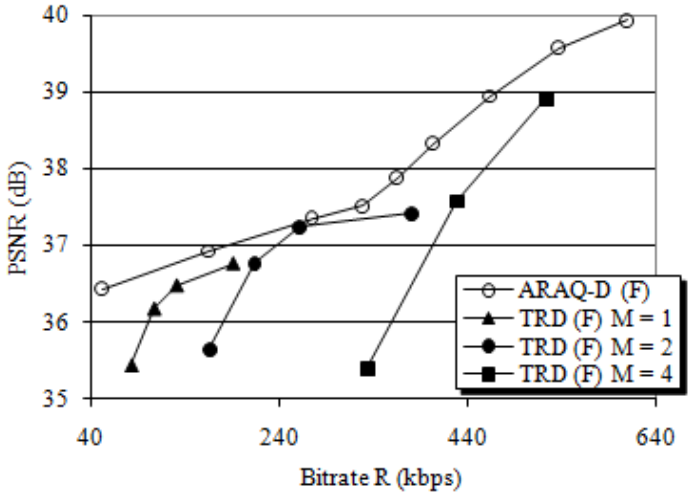


Figure 5.13 RD curves for the Foreman (F) sequence obtained with the ARAQ-D technique, compared to the performance of a TRD system.

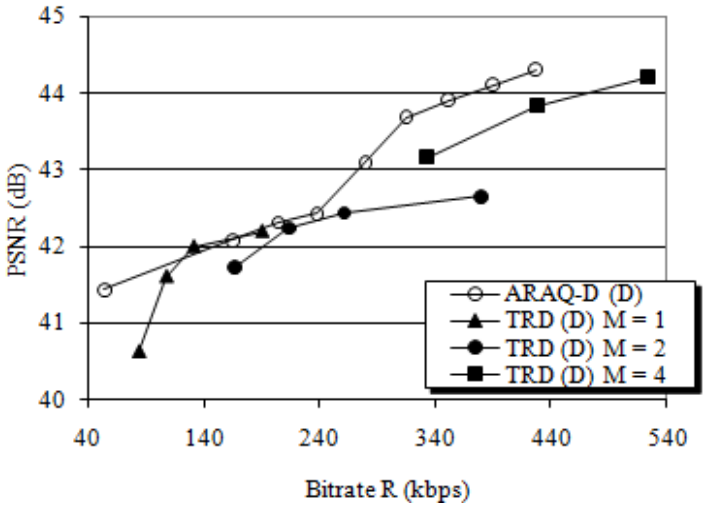


Figure 5.14 RD curves for the Mother-Daughter (D) sequence obtained with the ARAQ-D technique, compared to the performance of a TRD system.

Figure 5.15 Snapshot from the Carphone sequence. (a) Original image. (b) TRD turbo-decoded image with $M = 4$ and $\rho = 0.21875$. (c) Reconstructed image from (b). (d) ARAQ-D turbo-decoded image with $M = 2$ and $\rho = 0.4375$. (e) Reconstructed image from (d).

In Figure 5.16, we show the 40th WZ frame from the Foreman sequence. In the ARAQ system, the original WZ frame was quantized with $M = 1$, compressed with $\rho = 0.25$, and transmitted at 95 kbps over a BSC with q = 9E-3. As in the previous example, the image was

decoded with a high BER (0.02) which caused undesirable noise in the reconstructed output (Figure 5.16-b). The ARAQ-D system determined that under these conditions, it is not possible to recover an acceptable version of the transmitted frame. Therefore, the frame was dropped in the transmitter and replaced by the corresponding side information at the receiver side (Figure 5.16-c). It can be clearly seen that the resulting image has a smoother visual effect. The PSNR of Figure 5.16-b is 30.65 dB compared to 32.61 in Figure 5.16-c, resulting in a 2 dB gain. However, in both images, we notice a deformation of the mouth (compared to Figure 5.16-a). In fact, this frame was captured from a fast mouth opening and closing scene (high motion of the lips), which yielded less accurate side information for this region. In general, when the available transmission rate permits an error-free decoding of the WZ frame, the influence of inaccurate side information on the final output is reduced to a large extent.

(a) Original image (b) ARAQ reconstructed image (c) ARAQ-D reconstructed image

Figure 5.16 Snapshot from the Foreman sequence. (a) Original image. (b) ARAQ reconstructed image with $M = 1$ and $\rho = 0.25$. (c) ARAQ-D output image obtained by replacing the frame with its corresponding side information.

In our simulations, up to this point, key frames were assumed to be perfectly recovered at the receiver. The case where key frames are subject to degradations due to lossy source coding or channel impairments can be easily taken into account in our study by slightly modifying our proposed algorithm, which will be the subject of our next discussion.

Let R_T represent the total transmission rate to be shared between all users: $R_T = R_{Key} + R_{WZ}$, where R_{Key} is the rate assigned for key frames and R_{WZ} the rate assigned for WZ frames. The source coding rate $R_{S,i}$ for the key frames of user i is determined by the quantization parameter QP of the H.264 intra-encoder. After QP has been determined using the RD analysis of Section 5.2.3, R_{Key} can be calculated by:

$$R_{Key} = \sum_{i=1}^{N} R_{S,i} / r_i \tag{5.8}$$

where r_i represents the channel coding rate assigned to a given key frame. r_i can be varied by applying different puncturing schemes to the parity output of a quadri-binary turbo-encoder similar to the one used for Wyner-Ziv coding. The necessary channel coding rate to guarantee almost error-free transmission depends on the channel conditions (BSC crossover probability) and is determined dynamically for each key frame using the turbo-code performance tables constructed offline (i.e. the BER as a function of the code rate and the crossover probability).

Now that the necessary rate for each key frame has been determined, we can proceed with the rate allocation of the WZ frames. The total WZ transmission rate is: $R_{WZ} = R_T - R_{Key}$. Therefore, the dynamic rate allocation and quantization algorithm can be applied for the WZ frames, by substituting R_{WZ} for R in Eq. (5.2) or Eq. (5.3).

Figure 5.17 shows the R-D performance of the DVC system with our rate allocation technique compared to a traditional WZ system where all users are assigned fixed quantization parameters and compression rates for the WZ frames. As for the key frames, the same technique for source and channel coding is performed in both systems. Therefore, the H.264 bit stream of the coded key frames (with QP=30) is almost perfectly turbo-decoded in both systems, thanks to the application of the necessary amount of protection using Eq. (5.8).

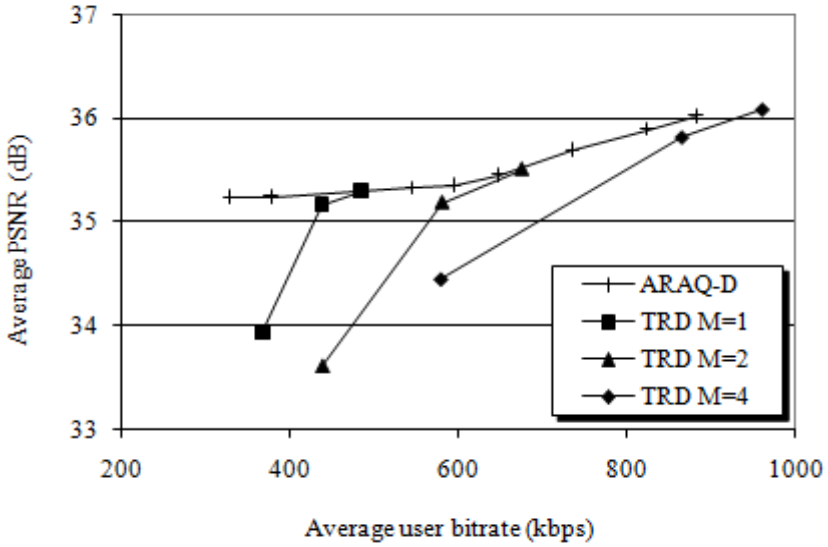


Figure 5.17 RD Curves obtained with a TRD system and with the ARAQ-D technique, for the case where key frames are subject to degradation.

The PSNR is averaged over the N=3 users and shown as a function of the average bit rate per user, including both key and WZ frames. The system behavior, as shown in Figure 5.17, is comparable to the case where the key frames are perfectly decodable (Figure 5.11). Therefore, the previous discussion (for the case of perfectly decoded key frames) still holds. It is important to note that, at very low rates, key frames occupy most of the channel resources, and not enough error protection can be provided for the WZ frames. In this case, the proposed frame dropping strategy allows the system to rely on the received side information, which results in a performance gain of 1.3 dB at 365 kbps.

5.3 Dynamic GOP Size Selection

In addition to the need for a feedback channel, another drawback of current DVC systems is that the quality of the generated side information greatly affects the system's performance. Even though the interpolation algorithm used at the decoder strongly influences the side information quality, key frames are essential components in the interpolation process and thus, having high quality key frames is crucial. Therefore, a very high PSNR is desired (for the key frames) in order to allow a successful decoding of the WZ frames at feasible WZ bit rates. This condition can result in a very high bit rate requirement, which is not possible in limited bandwidth applications. Additionally, when the key frames are too distant apart, the quality of the side information is degraded. As a result, most research on DVC considers a GOP size of 2, i.e. each key frame is followed by one WZ frame. Several attempts have been made to increase the GOP size in DVC. In [AAR03], Aaron et al. impose the use of high–quality key frames with fixed GOP sizes ranging from 2 to 5. As the GOP size increases, the system's performance decreases. However, lower rates can be reached with greater GOP sizes due to the high bit rate requirements of the key frames. In [ASC06], Ascenso et al. present a content-adaptive GOP size selection algorithm. The number of frames in a GOP is determined dynamically depending on motion activity. However both studies rely on a feedback channel for the decoding of WZ frames, and on H.263+ for key-frame encoding. Since H.264/AVC greatly outperforms H.263+, it is expected that H.264 intra-coding will outperform both Wyner-Ziv systems too.

In this section, we develop novel algorithms for dynamically varying the GOP size in distributed video coding. Our method relies on H.264 for the encoding of key frames, and on our previously developed WZ rate estimation technique presented in Section 5.2 , where a

feedback channel is not needed for the decoding of WZ frames. Automatic mode selection allows the system to switch to H.264 intra-coding mode in regions where H.264 outperforms WZ video coding. Furthermore, based on our study in Section 5.2 , our algorithms can be easily extended to take into account channel impairments and multiuser scenarios.

5.3.1 Adaptive algorithms for GOP size control

In a video sequence, when there is low motion, consecutive frames are highly correlated. The aim of varying the GOP size is to allow the system to better exploit this property, by reducing the number of intra-coded key frames in regions where WZ frames would yield a better RD performance. In regions where intra-coding outperforms WZ coding (because of high motion), the GOP structure is reduced to one (H.264 intra-coded) frame per GOP. This automatic mode selection allows the WZ encoder to make use of H.264 coding efficiency to better improve the system's RD performance.

Let S_{max} represent the maximum allowable GOP size. For each GOP, let R_0 represent the average bit rate assigned for the first frame (intra-coded key frame) in the GOP, and $PSNR_0$ its PSNR. S_{max} can be chosen depending on the system's delay constraints. For a GOP of size S, let F_0 denote the key frame, F_1, F_2, … , F_{S-1} the WZ frames, and F_S the key frame of the next GOP.

To perform GOP length decision, our proposed algorithm operates as follows:

Initially, set $S=1$.
While $S \leq S_{max}$ *do*:
If $S == 1$, *go to step (e), otherwise:*

a) *Interpolate between* F_0 *and* F_S
The interpolated frame serves as an estimate of the side information available at the decoder during the decoding process of the WZ frame $F_{\lfloor S/2 \rfloor}$, located at half-distance between F_0 and F_S. Since motion estimation is not allowed at the encoder for complexity reasons, average interpolation (AVI) can be used to estimate the side information that will be available at the decoder.

b) *Estimate the average bit rate* $R_{\lfloor S/2 \rfloor}$
Given the WZ frame $F_{\lfloor S/2 \rfloor}$ and its corresponding side information (estimate), the encoder determines its lower compression bound (which depends on its motion

level), and consequently, its compression rate, as explained in Section 5.2.3. The computation of $R_{\lfloor S/2 \rfloor}$ becomes straightforward.

c) *Compute* $PSNR_{\lfloor S/2 \rfloor}$

Given the WZ frame $F_{\lfloor S/2 \rfloor}$ and its corresponding side information (estimate), the encoder can determine an estimate $F'_{\lfloor S/2 \rfloor}$ of the decoded frame that will be obtained at the receiver by first quantizing the WZ frame, and then reconstructing an 8-bit version using the available side information. The PSNR is then computed using $F_{\lfloor S/2 \rfloor}$ and $F'_{\lfloor S/2 \rfloor}$.

d) *Repeat steps (a) to (c) until rate and PSNR estimates are obtained for all the frames of the GOP*.

However, instead of interpolating between F_0 and F_S in step (a), the frames F_0 and $F'_{\lfloor S/2 \rfloor}$ are first used to generate an SI estimate for the frame located at half-distance between the two, and the same process in steps (b) to (d) is repeated. Then, a similar procedure is performed in the second half of the GOP, using the frames $F'_{\lfloor S/2 \rfloor}$ and F_S. Frame $F'_{\lfloor S/2 \rfloor}$ is used instead of $F_{\lfloor S/2 \rfloor}$ because the former better estimates the frame that will be available at the decoder side since the latter is not known by the decoder.

e) *Estimate the average rate and PSNR obtained with a GOP of size S, respectively defined as*:

$$R_{av}^{S} = \frac{1}{S}\sum_{j=0}^{S-1} R_j \text{ and } PSNR_{av}^{S} = \frac{1}{S}\sum_{j=0}^{S-1} PSNR_j$$

f) *Determine* $\lambda_{av}^{S} = PSNR_{av}^{S} / R_{av}^{S}$

This represents the average PSNR per average unit bit rate estimated for a GOP of size *S*.

g) $S = S + 1$

The best RD performance is obtained by maximizing the average PSNR per unit bit rate. As a result, the system decides the GOP length *L* as:

$$L = \arg\max_{k=1,2,\ldots,S_{\max}} \left(\lambda_{av}^{k} \right) \tag{5.9}$$

In other words, if the system determines that the average PSNR per unit bit rate obtained by WZ coding, for different GOP lengths ($S > 1$), is lower than the one obtained with H.264 intra-coding ($S = 1$), the system switches to H.264 intra-coding mode ($L = 1$) and an H.264 I-frame is then transmitted. Otherwise ($L > 1$), an H.264 intra-coded key frame is transmitted, followed by L-1 WZ frames. Furthermore, since MCI yields better side information compared to AVI, in general, the decoder is expected to perform better than estimated at the encoder side. This procedure is repeated at the beginning of every GOP and thus, the GOP length is dynamically varied along the sequence, in order to optimize the overall performance.

Figure 5.18 presents the algorithm above as a flow-chart diagram, and Figure 5.19 shows the pseudocode of the recursive procedure `R_PSNR_estimations(i,len)` used to estimate the rate and PSNR for all the frames in a GOP of size S, where i is the time index of the first frame in the GOP, and *len* is the time interval between the frame at index i and the next frame used during the interpolation process (initially, $len = S$).

In general, a constant quality is desired along the sequence, since big fluctuations in PSNR yield undesirable visual effects. For this reason, the encoder determines the rate for the H.264 intra-coded frames in such a way to obtain a near-constant PSNR in the GOP. This can be done by one of several possible techniques:

- Using pre-defined performance tables that determine H.264 rate-distortion relationships.
- Using an analytical model for H.264 rate-distortion performance. This allows the system to avoid extensive table search as in the previous technique. However, it is important in this case to have an accurate generalized RD model [ORT98].
- Trial and error: the system tries several coding rates for each H.264 intra-coded frame, and determines the PSNR for each case. Then, the rate that yields a PSNR closer to the one of neighboring frames is chosen. This method is more accurate than the previous ones. However, it can result in significant delay and increased encoder complexity.

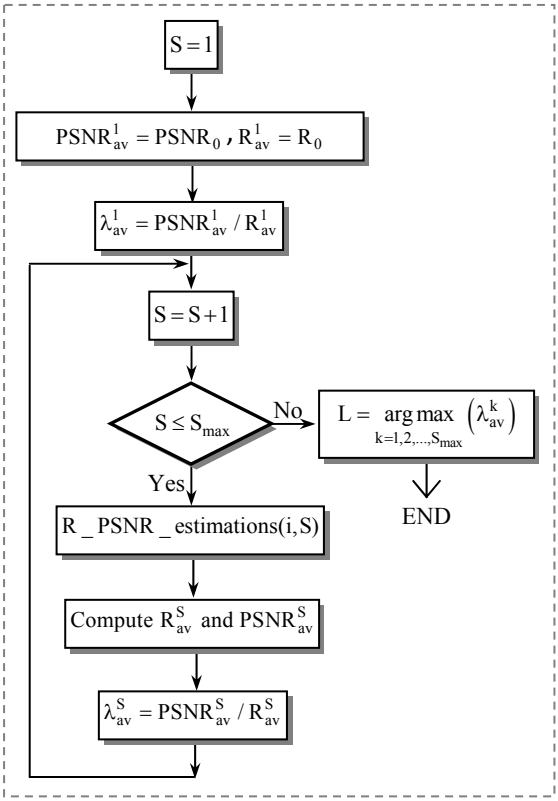


Figure 5.18 Flow-chart diagram of the proposed GOP size control algorithm.

In fact, the GOP size control algorithm can be further simplified by assuming a constant PSNR for all the frames. In this case, instead of maximizing the average PSNR per unit bit rate, Eq. (5.9) reduces to minimizing the average bit rate per frame over all possible GOP sizes. As a result, the simplified GOP size control algorithm operates as follows:

Initially, set S=1.

While $S \leq S_{max}$ *do*:

If S == 1, go to step (d), otherwise:

a) *Interpolate from* F_0 *and* F_S

b) *Estimate the average bit rate* $R_{\lfloor S/2 \rfloor}$

c) *Repeat steps (a) and (b) until a rate estimate is obtained for all the frames in the GOP (replace frames* F_0 *and* F_S *in step (a) with the corresponding frames as previously explained in step (d) of the initial algorithm).*

d) *Determine* $R_{av}^S = \frac{1}{S}\sum_{j=0}^{S-1} R_j$

e) $S = S + 1$

Finally, the system decides the GOP length L as:

$$L = \arg\min_{k=1,2,\ldots,S_{\max}} \left(R_{av}^{k} \right) \tag{5.10}$$

This allows the system to avoid estimating the PSNR for each frame, and the average PSNR per unit bit rate for each GOP size.

```
procedure R_PSNR_estimations(i, len) {
  If len < 2
    Return                    // End of the recursive function calls
  Else
    a = i, b = i+len          // a and b are the time indices of the frames used during the interpolation process.
    d = ⌊(b – a)/2⌋           // time interval from a to the frame at mid distance between a and b.
    SI(a+d) = Interpolate(a,b)          // Perform average interpolation between frames at time indices a and b.
    Q(a+d) = QuantizeFrame(F(a+d))
    R(a+d) = EstimateBitrate(Q(a+d))    // estimate the bit rate as explained in Section 2.
    F'(a+d) = Reconstruct(Q(a+d), SI(a+d))   // reconstruct the quantized WZ frame given the estimated
                                             // side information.
    PSNR(a+d) = ComputePSNR(F'(a+d), F(a+d))  // Compute the PSNR of the reconstructed frame.
    R_PSNR_estimations(i, d);          // Recursive function call using the first half of the GOP.
    R_PSNR_estimations(i+d, len-d)  // Recursive function call using the second half of the GOP.
  End If
}
```

Figure 5.19 Pseudocode of the recursive procedure R_PSNR_estimations() used to estimate the rate and PSNR for all the frames in a GOP.

5.3.2 Analysis of the computational load for the proposed GOP size control algorithms

While the aim of DVC is mainly permitting the design of low-complexity encoders, our GOP size selection algorithms incur additional encoding complexity. In this section, we present an analysis of the proposed algorithms computational load, and we compare our dynamic algorithms with the one presented in [ASC06].

Table 5.2 presents an estimation of the necessary number of additional operations (OPs) incurred by each iteration (for each frame in the GOP, for every GOP size k, $1 \leq k \leq S_{max}$) in

the initial (non-simplified) GOP size control algorithm. PxQ represent the frame dimensions, and M the quantization parameter for WZ frames.

Consider for example the number of operations performed to compute the *PSNR*. The calculation of the *PSNR* consists of first computing the mean square error (*MSE*), taking its inverse, multiplying it with a constant, computing the log of the result, and finally multiplying it by 10. The square error between two pixel values requires one addition operation and one multiplication. To compute the *MSE*, PQ square errors are first computed (which results in PQ additions and PQ multiplications) and summed together (PQ-1 additions). The final result is then divided by PQ. As a result, 2PQ-1 additions, PQ+4 multiplications, and a log operation are performed in order to obtain the *PSNR*. A similar analysis was performed on the other operations involved in the GOP size control algorithm, and the results are summarized in Table 5.2.

The total number of operations is roughly obtained by summing the elements of the last row in the table, which results in $12PQ + (8 \times 2^{2M}) + 3$ operations. Since, in our codec, the maximum value for M is 4, and by assuming QCIF video sequences (PxQ = 144x176), the total number of operations becomes 306,179 OPs. In the simplified algorithm, the computation of the *PSNR* is not performed. Thus, a reduction of 3PQ+3 operations is obtained, which yields a total number of 230,144 OPs.

A similar study was performed on the algorithm presented in [ASC06], where four different metrics were used: the difference of histograms (DH), the histogram of difference (HD), the block histogram difference (BHD) and the block variance difference (BVD). Given the parameters specified in [ASC06], we obtain 50,784 OPs for DH, 50,736 OPs for HD, 60,191 OPs for BHD and 161,567 OPs for BVD, which results in a total of 323,278 OPs.

Even though the computational load of our initial algorithm and the one of the algorithm in [ASC06] are almost similar, it is important to note that our algorithm presents the additional property of estimating the necessary bit rate without the need for a feedback channel, and the possibility to take into account channel impairments based on our information theoretic study presented in Chapter 4, and earlier in this chapter. On the other hand, a reduced complexity of approximately 25% can be obtained with our simplified algorithm, without a significant loss in RD performance as will be shown in the next section.

OPERATION	AVI	Reconstruction	Quantization (WZ and SI)	Estimation of the α parameter	PSNR estimation	Rate (R) estimation
Additions	PQ	PQ		$2PQ$-1	$2PQ$-1	1 x 2^{2M} -1
Multiplications	PQ			PQ+1	PQ+4	5 x 2^{2M} +1
Log or Exp					1	2 x 2^{2M}
Comparisons		PQ				
Logical AND			$2PQ$			
TOTAL	$2PQ$	$2PQ$	$2PQ$	$3PQ$	$3PQ$+3	8 x 2^{2M}

Table 5.2 Number of additional operations per frame for each GOP size k, $1 \leq k \leq S_{max}$, incurred by the proposed GOP size control algorithm.

5.3.3 Simulation results

In our simulations, we consider three different QCIF video sequences with different levels of motion: Foreman, Grandmother, and Salesman, sampled at a rate of 30 fps. The first 100 frames from each sequence are first encoded using a WZ codec with fixed GOP sizes ranging from 1 to 5. For the case where the GOP size is 1, all frames are H.264 intra-coded, whereas for the other cases, only the first frame (key frame) from each GOP is H.264 intra-coded, while the remaining ones are WZ-coded. H.264 coding is performed using JM FRExt reference software, version 13.2, with baseline profile. The results are then compared with the case where a WZ codec with a dynamically varying GOP size is used. The GOP size is determined using our proposed algorithms as explained in Section 5.3.1, with S_{max} set to 5.

In Figure 5.20, we show the rate and PSNR variations along the Grandmother sequence for $M = 4$, and the quantization parameter of the H.264 intra-frames QP = 25, obtained using a WZ codec with a fixed GOP size set to 3, with and without a feedback channel (FC). It can be noticed that the rate estimated without FC exceeds the rate obtained using FC most of the time. As a result, WZ frames are correctly decoded in both cases and the reconstructed output is the same. However, in rare situations (e.g. in frame 41), the encoder under-estimates the rate needed for correctly decoding a WZ frame, which yields a degraded quality at the decoder output. For an average bit rate of 697 kbps obtained with the feedback channel, an average rate excess of 50 kbps is observed in the case where the feedback channel is suppressed. In other words, for applications where the feedback channel is not suitable (for example, in video broadcasting), the cost to be paid (in terms of bit rate) for the suppression of the return channel is approximately 7%.

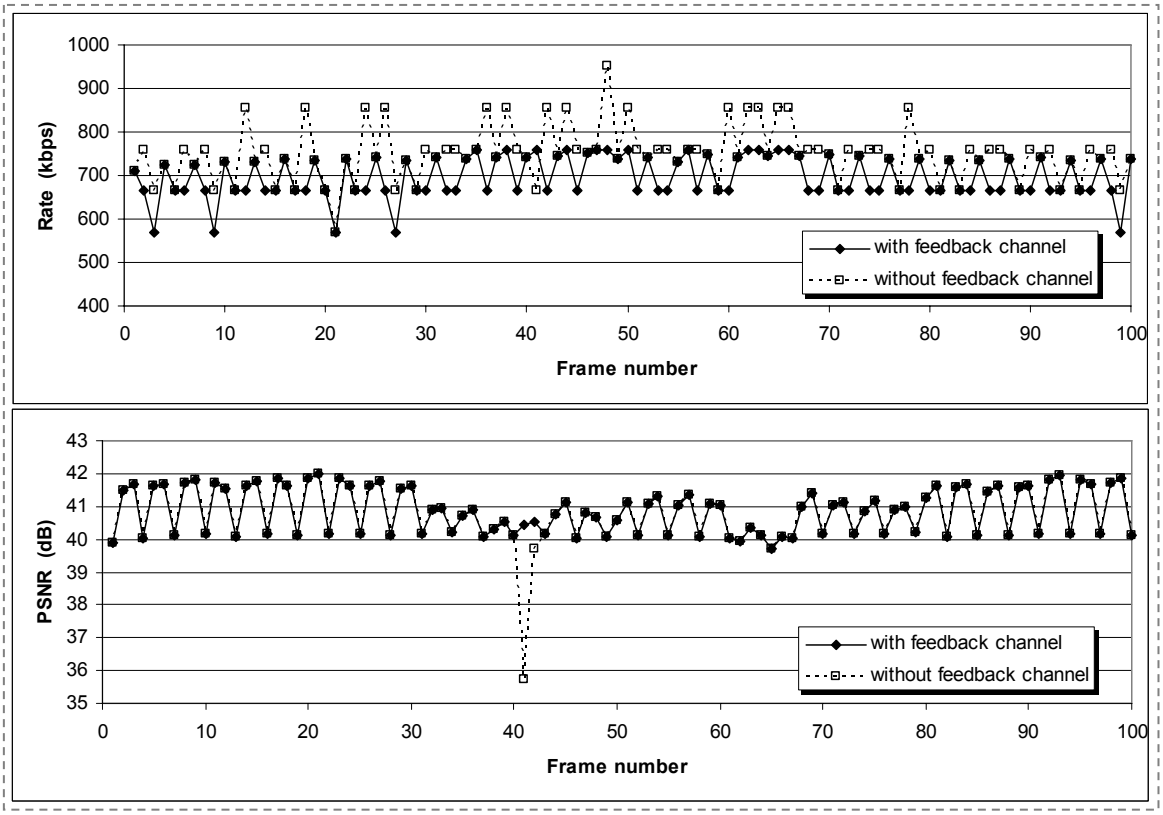


Figure 5.20 Rate (top) and PSNR (bottom) variations along the Grandmother sequence using a WZ codec with a GOP size = 3, M = 4, and QP = 25.

In Figure 5.21, we show the average RD curves obtained for the three sequences using both the initial (curves labeled 'I' in the figure) and simplified (curves labeled 'S' in the figure) algorithms. The rate and PSNR are averaged over all the sequence (key and WZ frames). Different rate points are obtained by varying the quantization parameter M for the WZ frames. As for the quantization parameter (QP) of the H.264 intra frames, it is chosen in such a way to permit a near-constant decoding quality in the output video sequence as in [ASC06]. It can be clearly seen that, for the Salesman and Foreman sequences, both curves overlap (both algorithms have similar performance), whereas a negligible loss that does not exceed 0.45 dB is observed with Grandmother by using the simplified algorithm.

Figure 5.22, Figure 5.23 and Figure 5.24 show the average RD performance for the Foreman, Grandmother, and Salesman sequences respectively, obtained with the initial (non-simplified) algorithm. In Figure 5.22 (Foreman), we notice that for the case of a fixed GOP size, the performance decreases as the GOP size increases. The best performance is thus obtained when all frames are intra-coded. This is due to the high motion in this sequence, which yields less accurate side information when the key frames are further apart. A similar effect has

been noticed in [AAR03] where key frames were encoded using an H.263+ video codec. However, when the GOP size is dynamically varied along the sequence, a gain of 10 to 12 kbps is obtained compared to H.264 intra-coding. For sequences with lower motion levels, different results are observed. It can be seen in Figure 5.23 and Figure 5.24 that the best system performance is obtained with a GOP of size 3 (for the fixed-GOP case). Our proposed system outperforms both H.264 intra coding and fixed-GOP WZ coding in most cases. For example, for the Grandmother sequence at 520 kbps, a gain of 3 dB is observed with respect to the H.264 intra codec and 0.8 dB with respect to the WZ codec with a GOP size of 3. Similarly, for the Salesman sequence at 580 kbps, our proposed algorithm outperforms the H.264 intra codec and the WZ codec with a GOP size of 3 by 1 dB and 0.1 dB respectively.

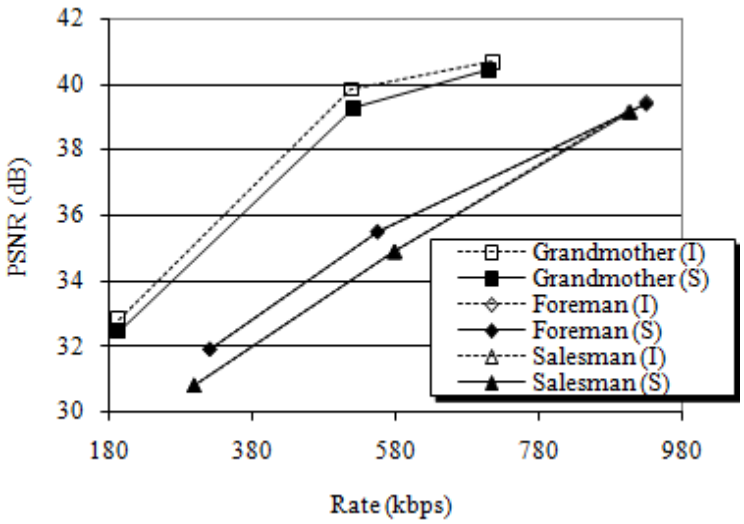


Figure 5.21 Average RD curves obtained using the initial (I) and simplified (S) adaptive GOP size control algorithms.

However, a performance loss of 0.4 dB can be observed with the Salesman sequence at 900 kbps using a dynamic GOP size, compared to the fixed-GOP (size = 3) WZ codec. In fact, this is due to a significant mismatch between the side information available at the encoder (estimated using AVI) and the one available at the decoder (obtained by MCI).

It is important to note that in practical situations, the optimal GOP size cannot be known in advance, without effectively encoding and decoding the sequence with different GOP sizes. However, with our proposed GOP size control algorithm, the system is able to determine the optimal GOP size and, at the same time, improve the average R-D performance, since the GOP size is dynamically varied along the sequence, depending on the motion level in the video scene.

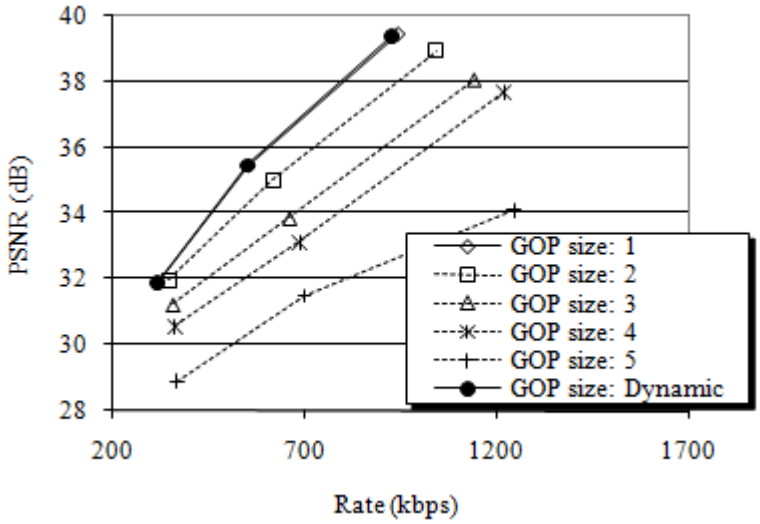


Figure 5.22 Average RD curves for the Foreman sequence using a WZ codec with fixed and variable GOP sizes.

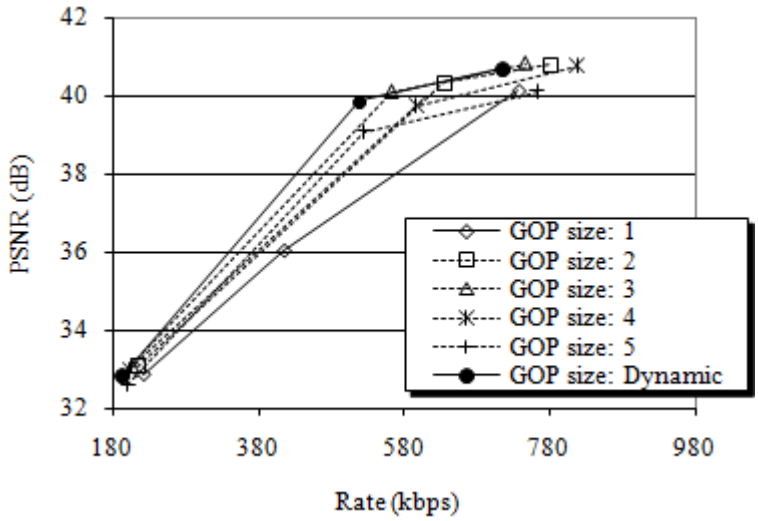


Figure 5.23 Average RD curves for the Grandmother sequence using a WZ codec with fixed and variable GOP sizes.

Figure 5.25 and Figure 5.26 show the rate and PSNR variations along the Salesman sequence for $M = 4$ and $M = 2$ respectively, for the case where the rate estimation is done at the encoder using average-interpolated side information (using AVI), and for the case where this estimation is performed at the decoder using the side information obtained by motion-compensated interpolation (MCI). In fact, the actual source coding rate is the one estimated by the encoder, whereas the real PSNR is the one obtained after the decoding process and thus, the corresponding curves are shown as solid lines. On the other hand, decoder-estimated bit rate and encoder-estimated PSNR (dotted curves) are shown only to analyze the system's

behavior at both (encoder and decoder) sides. It can be seen that, in some regions (e.g. frames 61 to 64 and frames 71 to 74 with $M = 4$), the encoder under-estimates the rate necessary for the decoding of WZ frames, which yields a very high bit error rate at the turbo-decoder output. As a result, the reconstruction function of the WZ codec cannot yield a reliable output and thus, a significant performance loss is observed in these regions, which greatly affects the average system performance as shown in Figure 5.24. However, such estimation errors rarely occur. As it can be clearly seen in Figure 5.25 and Figure 5.26, the encoder estimations are accurate most of the time when $M = 4$, and all the time when $M = 2$. Similar results were observed with the other sequences for different values of M.

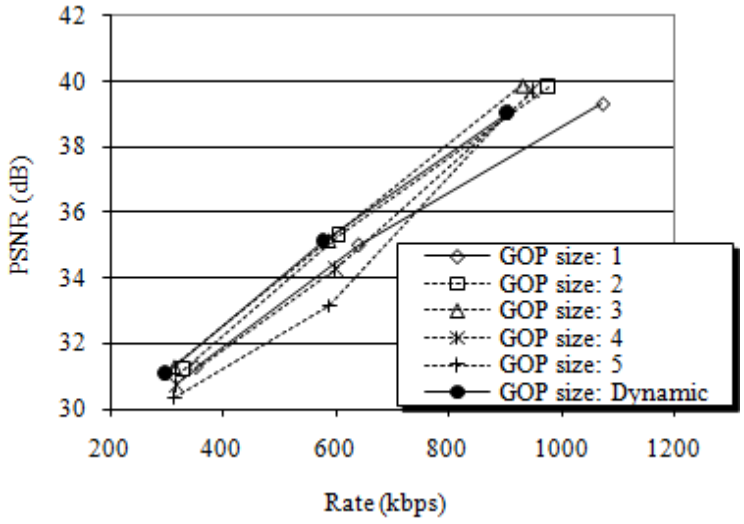


Figure 5.24 Average RD curves for the Salesman sequence using a WZ codec with fixed and variable GOP sizes.

On the other hand, we represent, in Figure 5.27, the variations of the ratio PSNR/R for fixed and adaptive GOP size coding for the Grandmother sequence, with $M = 2$. The plot shows only 80 frames for clarity of visualization. Points labeled with "I" indicate the beginning of a GOP (intra-frame) in the case of a dynamic GOP size. It can be clearly seen that, most of the time, the greatest ratio is obtained with the proposed algorithm. Only for a few frames, the best result is obtained with fixed GOP or intra-coding. This is expected since the proposed technique maximizes the ratio between the average PSNR and average bit-rate per GOP, whereas the plot is in terms of the frame number.

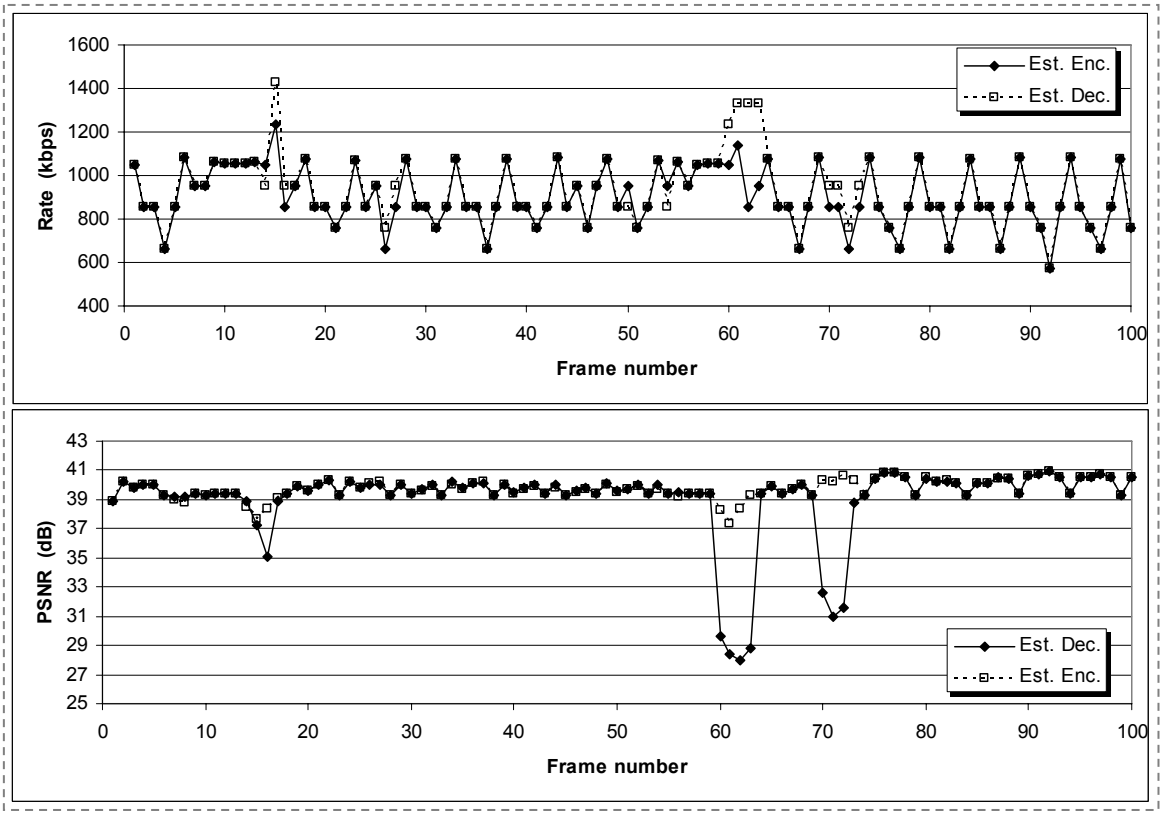


Figure 5.25 Rate (top) and PSNR (bottom) variations along the Salesman sequence using a WZ codec with the proposed GOP size control algorithm for $M = 4$.

Table 5.3 shows the percentage of GOP sizes obtained for each of the sequences using the proposed adaptive GOP size control algorithm (non simplified), for different values of the WZ quantization parameter M. For the Foreman sequence, when $M = 1$, 100% of the GOPs are of size 1. In other words, the system switches to H.264 intra coding mode all the time, and no frame in the sequence is WZ-encoded. For $M = 2$ and $M = 4$, most of the GOPs are of size 1, while the maximum GOP size does not exceed 3. This explains the reason behind the similar performance between the H.264 and our proposed WZ codec for the Foreman sequence, as shown in Figure 5.22. More GOP size variations can be noticed with the two other sequences. For the Grandmother sequence with $M = 1$ and $M = 2$, the system is always in WZ coding mode. In other words, not any GOP is of size 1 and only key frames are intra-coded, since the WZ coding outperforms H.264 intra-coding in this case, according to the encoder estimations.

The Foreman sequence is characterized by very high motion levels, especially in its second half. In such regions with high motion, H.264 intra coding usually outperforms WZ coding.

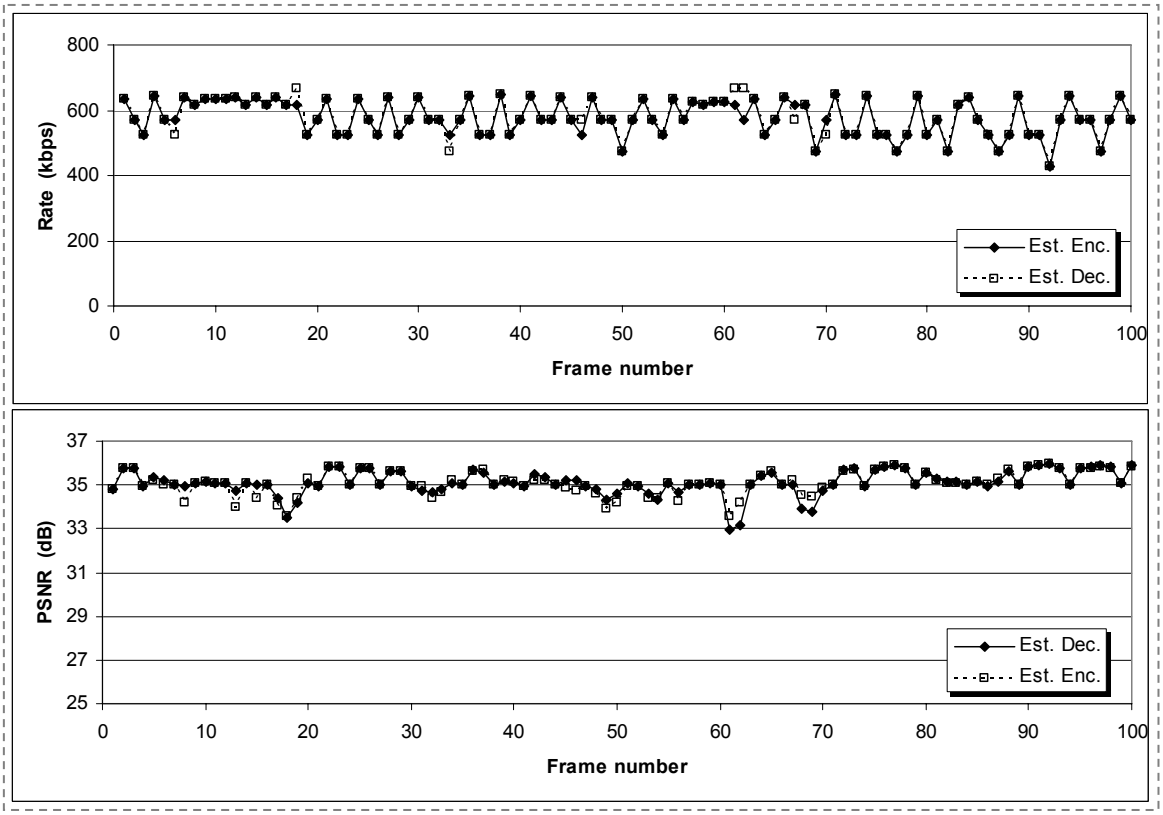


Figure 5.26 Rate (top) and PSNR (bottom) variations along the Salesman sequence using a WZ codec with the proposed GOP size control algorithm for $M = 2$.

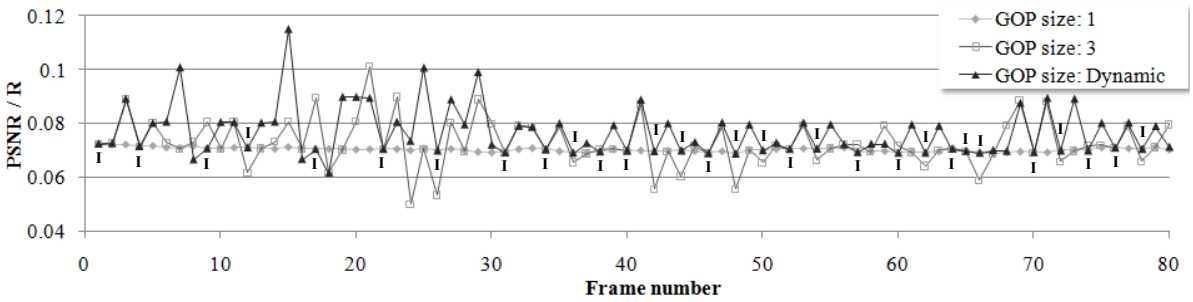


Figure 5.27 Variations of the ratio *PSNR*/*R* for the Grandmother sequence, for $M = 2$.

For this reason, in order to analyze our system's performance with high motion video, we encoded the complete Foreman sequence (400 frames) and the results are reported in Figure 5.28. This figure shows the GOP size variations along the sequence for $M = 2$ and $M = 4$. Long runs of consecutive GOP sizes equal to 1 can be noticed in high motion areas, which indicates that the system has switched to H.264 intra coding mode in these regions. As a result, the RD performance for the Foreman sequence is slightly better than the one obtained with a pure H.264 intra-coder, as was also noticed in Figure 5.22.

Gop size:			1	2	3	4	5
Foreman	M=1	%	100	0	0	0	0
	M=2		91.1	6.7	2.2	0	0
	M=4		87.4	10.3	2.3	0	0
Grandmother	M=1		0	4	20	36	40
	M=2		0	4.2	16.7	37.5	41.6
	M=4		23.1	33.3	20.5	2.6	20.5
Salesman	M=1		3.7	22.2	18.5	0	55.6
	M=2		18.2	12.1	39.4	0	30.3
	M=4		22.2	7.4	3.7	0	66.7

Table 5.3 Percentage of GOP sizes used in each sequence.

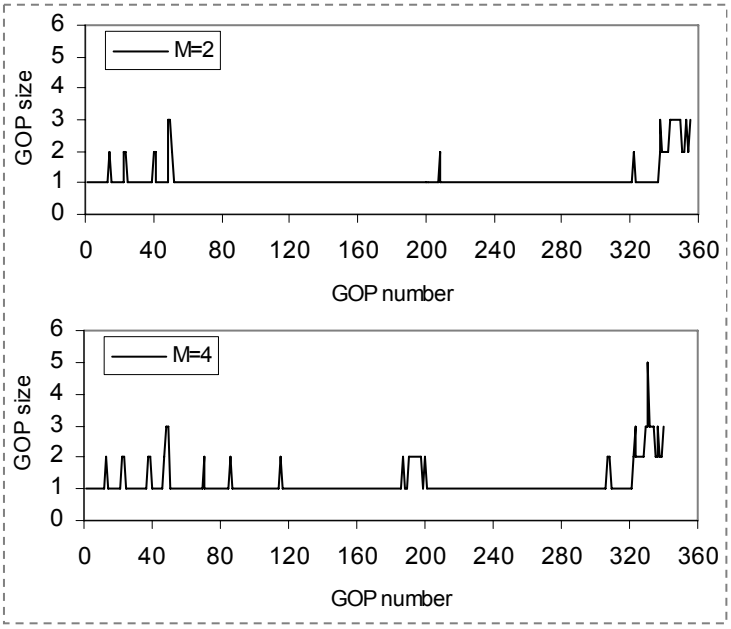


Figure 5.28 GOP size variations along the complete Foreman sequence (400 frames) for $M = 2$ (top) and $M = 4$ (bottom).

5.4 Conclusion

In this chapter, we presented two different applications based on our information-theoretic approach for feedback channel suppression. First, a novel rate allocation technique for distributed multiuser Wyner-Ziv video coding systems was proposed. Using entropy calculations, the available system bandwidth was unequally distributed among several users transmitting data to a central base station without the need for a permanent feedback channel. The proposed rate allocation technique allowed the system to adapt to the random variations of the wireless channel and to different amounts of motion captured by the different users. The quantization parameters were dynamically varied to optimize the decoding quality and a

frame dropping mechanism allowed the system to avoid unnecessary channel use. Our results obtained by simulating a network of multiple users showed a significant improvement in the average system performance, as well as in the individual RD performance of each user. Then, we presented simple algorithms that dynamically adapt the GOP size for a distributed Wyner-Ziv video codec, depending on the content of the video scene to be encoded. Based on H.264 intra-coding for key frames, the system relied on theoretical calculations to estimate the bit rate necessary to successfully decode Wyner-Ziv frames without the need for a feedback channel, which makes it suitable for broadcasting applications. Automatic mode selection allowed the system to switch between H.264 intra-coding and WZ coding modes in order to optimize the overall system performance.

In the next chapter, we will focus our study on improving the quality of the side information generated at the decoder side, where a novel algorithm will be proposed for this purpose.

Chapter 6

A Fusion-Based Approach for Improving the Side Information in DVC

6.1 Introduction

In the previous chapters, we targeted the problem of the feedback channel in distributed video coding. We presented an analytical approach for estimating the necessary compression rate without the need for feedback information from the decoder, and developed several applications based on return channel suppression. In this chapter, we aim at improving the generation of side information in Wyner-Ziv video coding. For this purpose, we first present a frame fusion approach that combines several interpolated frames in a hash-based DVC system. Then, we develop a novel method based on genetic algorithms to further improve the frame fusion technique.

6.2 Fusion of Multiple Side Information

The quality of the side information greatly affects the rate-distortion performance of a distributed video codec. On one hand, it is used as a first estimate of the transmitted WZ frame in the decoder. Therefore, improving the side information leads to a reduction of the bit rate necessary for the compression of the WZ frame. On the other hand, the SI is used for the reconstruction of the quantized symbols after source-channel decoding. As a result, improving the side information reduces the distortion at the decoder output.

Several interpolation techniques for generating accurate side information have been presented in the literature. Aaron et al. first proposed average interpolation and motion-compensated interpolation in [AAR02-2]. In [ASC05-2], Ascenso et al. presented an improved motion-compensated interpolation using spatial motion smoothing. In hash-based DVC [AAR04-2, ASC07], hash bits were used to improve the quality of the side information. Edge-based and mesh-based interpolations were proposed in [TOA07] and [KUB06], respectively. Multiple

hypothesis techniques were developed in [MIS05] and [KUB07], where two different frames were used as side information for the decoding of a single WZ frame. Each of these methods outperforms some of the others in particular situations (e.g. background motion, moving objects, etc…). Besides, several fusion-based approaches were proposed in [ART06, OUA06, OUA07] to generate side information in multiview DVC.

Our technique for the enhancement of SI generation in monoview DVC is based on the following idea: since different SI frames can be obtained using different interpolation techniques, the decoder can take advantage of this diversity by selecting the best among the different versions of the side information. This decision can be made on a frame-level basis (i.e., by selecting the best SI frame) or on a region-level basis (i.e., separately for every region within the frame). However, since only key frames are available at the decoder, some information about the missing WZ frame needs to be transmitted in order to evaluate the quality of the side information. For example, statistical information (e.g., mean, variance) about the WZ frame, or a binary edge image [TOA07], can be transmitted for this purpose. As stated in earlier chapters, the encoding complexity is a main concern in DVC. Therefore, the computational burden required to construct this additional information should be minimized. Additionally, the transmission overhead required to deliver this information to the receiver should be negligible in order not to degrade the system's RD performance.

In hash-based DVC, a discrete cosine transform (DCT) is performed on the WZ frames, and a subset of the DCT coefficients is transmitted [AAR04-2, ASC07] as a hash word to aid the decoder in improving the system's RD performance. The hash word can be used during the interpolation process in a hash-based motion compensated interpolation (HMCI), where the minimal distance measure (e.g. square error) is performed between the received hash word and the one corresponding to the candidate SI block, instead of the block matching technique used in MCI. Therefore, the received hash bits can be used, in our frame-fusion technique, to evaluate the quality of the different SI frames.

6.2.1 A simple frame-fusion approach

In our study, the construction of the hash word is inspired from [AAR04-2] and [ASC07]. After computing the DCT coefficients for a block of dimensions BxB, a subset of n coefficients ($n < B^2$) that packs most of the energy within the DCT block is kept, while the remaining coefficients are replaced with zeros. Once this has been done for all the blocks within a frame, either the resulting DCT frame is compressed using entropy coding (e.g.

Huffman coding), or only the non-zero coefficients along with their relative positions are compressed and transmitted to the decoder. *B* and *n* are tunable parameters; they can be fixed or dynamically varied throughout the sequence.

At the decoder side, several interpolation techniques (e.g. AVI, MCI and HMCI) can be used to generate different SI frames. The key frames involved in the interpolation process can also be considered as candidate SI frames. For every block within a frame, a hash word is computed for all SI candidates as was done at the encoder side for the WZ frame. In order to evaluate which of the candidates better represents the missing WZ frame, the mean square error is computed between the received hash and each of the hash words extracted from the candidate SI blocks, after replacing the missing DCT coefficients with zeros. Then, the block that minimizes the MSE is selected as the final SI block.

Even though the MSE computed in the transform domain may not be an ideal measure, the proposed technique guarantees that the energy distribution of the DCT bands in the final SI frame is as close as possible to the original WZ frame.

6.2.2 Genetic Algorithms for Frame Fusion

Genetic Algorithms (GAs) [GOL89] are well suited for searching and optimization problems. They are based on the principles of evolution and natural genetics and have rarely been used in video applications. In [CHA01], a GA approach was proposed in order to obtain an up-sampled image sequence from a low-resolution image sequence. Genetic motion search and block matching algorithms were proposed in [CHO93] and [LIN96], respectively. To our knowledge, GAs have not been used in DVC applications up to this date.

In the previous section, the proposed frame-fusion approach is performed at the block level. In other words, different blocks in the final SI are obtained from different SI candidates. However, all the pixels within the same block belong entirely to the same SI candidate frame generated at the decoder. In order to perform frame fusion at the pixel level, i.e. in order to have different pixels originating from different SI candidates within the same block, we propose to use Genetic Algorithms. For this purpose, different SI frames are first generated using previously developed interpolation techniques. Actually, any SI generation technique can be considered as an input to our algorithm. The GA then combines two or more SI frames, in a fusion-based approach, in order to improve the side information generated at the decoder.

6.2.3 Proposed technique based on GA

In order to develop a GA, a genetic representation of the solution domain and a fitness function that allows evaluating each possible solution need to be defined. Initially, for a given block in the WZ frame, each of the co-located blocks in the available SI frames represents a possible solution. A candidate solution is referred to as a *chromosome* which consists of a sequence of genes. Therefore, a chromosome is defined as a sequence of pixels (genes) arranged in a matrix to form a block. A *population* is a set of chromosomes in the solution space. The similarity between a given chromosome and the corresponding block in the WZ frame represents its *fitness*. In hash-based DVC [AAR04-2, ASC07], the received hash bits can be used for fitness evaluation in the decoder. In our proposed GA, a hash word is computed for every chromosome in the population. The fitness of a chromosome is then evaluated as:

$$F = \frac{K_1}{K_2 + K_3 D}, \tag{6.1}$$

where K_1, K_2 and K_3 are tunable scaling factors, and D is a distance measure, such as the sum of absolute differences (SAD) or the sum of squared errors (SSE), between the received WZ hash word and the one computed for the current chromosome.

As a result, our genetic algorithm can be implemented in a hash-based Wyner-Ziv video codec without any complexity increase at the encoder side.

On the other hand, as previously discussed, the proposed GA can operate at the frame level or the block level. Since the former is a particular case of the latter, we start by presenting the block-level GA. Each frame is first divided into a number of blocks. For each block, the algorithm consists of an iterative procedure represented by the flowchart diagram of Figure 6.1, where i represents the iteration number and I_{max} represents the maximum allowed number of iterations. As shown in the figure, the GA consists of the following steps:

Step 1: Initialization

An initial population is generated using several SI frames obtained from different interpolation techniques. In our system, the initial candidate frames to be fed to the GA are obtained by AVI, MCI, and HMCI. The key frames involved in the interpolation process are also considered as possible solutions. Therefore, the initial number of candidates, for

each block of the current WZ frame, is five (the co-located block in each of the five candidate frames). The initialization of the GA consists of duplicating each candidate a number of times proportional to its fitness, until the desired population size S_p is reached. This population then undergoes subsequent genetic operations.

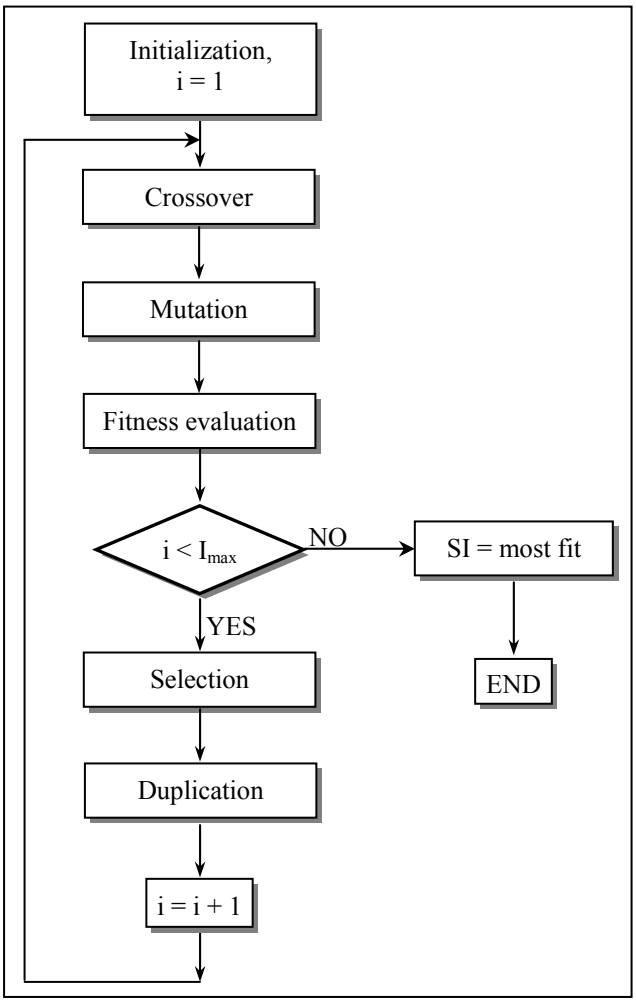


Figure 6.1 Flowchart diagram of the proposed genetic frame-fusion algorithm.

Step 2: Crossover

The chromosomes of the current population are randomly shuffled and arranged into pairs. Each pair then produces a new pair of child chromosomes (called off-springs). Offsprings are obtained by combining and exchanging genes between parent chromosomes. This process is illustrated in Fig.2.

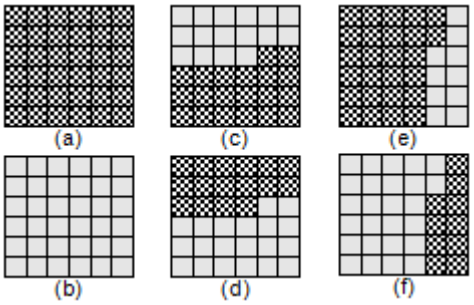


Figure 6.2 Crossover of parents (a) and (b) to produce offsprings (c) and (d) in vertical crossover or (e) and (f) in horizontal crossover.

Given a pair of parent chromosomes (Figure 6.2-a and Figure 6.2-b), a crossover position is selected at random and each of the parents is divided into two regions. One offspring is then obtained by combining the first region of the second parent with the second region of the first parent (Figure 6.2-c, Figure 6.2-f). Similarly, another offspring is obtained by combining the first region of the first parent with the second region of the second parent (Figure 6.2-d, Figure 6.2-e). The crossover can be performed vertically using a horizontal scan of lines within the block as in Figure 6.2-c and Figure 6.2-d, or horizontally using a vertical scan of columns as in Figure 6.2-e and Figure 6.2-f.

In case of a one-directional crossover (horizontal or vertical), the probability of a crossover to occur is controlled by a tunable parameter P_c. Therefore, for each pair of parent chromosomes, either a new pair is obtained (with a probability P_c), or parents remain unchanged (with a probability 1- P_c). If the crossover is performed in both (horizontal and vertical) directions, a crossover probability is specified for each direction. In this case, the GA will be referred to as the two-directional genetic algorithm (2D-GA). In the 2D-GA, step 2 either produces one or two new pairs of chromosomes or keeps the two parent chromosomes unchanged. Note that other directional crossovers (e.g. diagonal crossover) can be envisaged, but we limit our study to only two directions, for simplicity.

Step 3: Mutation

A random change is performed on a given chromosome. In our GA, a gene is first selected at random, and a bit in the selected pixel (gene) is then inverted. Mutation allows the GA to extend its solution space and reduce the possibility of falling into local optima, and it usually occurs with a very low probability P_m [CHA01].

Step 4: Fitness evaluation

In our GA, the SSE is first used as a distance measure between the received WZ hash word and each of the hash words computed for all the chromosomes in the current population. Then, the fitness function is evaluated for every chromosome as explained earlier.

Step 5: Selection

A number $S_f \leq S_p$ of chromosomes is selected, while $S_p - S_f$ chromosomes are deleted to make room for new ones. Only the most fit chromosomes survive. S_f must be chosen not too close to S_p in order not to give a chance for the least fit chromosomes to duplicate, and not too small in order to allow for a variety of chromosomes to exchange genes and duplicate.

Step 6: Duplication

Each of the S_f remaining chromosomes is duplicated a number of times proportional to its fitness, until the population size S_p is reached.

Steps 2 to 6 are then repeated until the maximum number of iterations is reached. Finally, the fittest chromosome is chosen as the best candidate to be used as side information for decoding the co-located block in the WZ frame.

We noticed that when a very high motion occurs in some parts of the video, the interpolation techniques used to initialize the GA often fail in these parts, which also degrades the performance of the GA. In this case, simply computing the inverse DCT (IDCT) given the received hash word (by first substituting the missing coefficients with zeros) yields a better result, provided that a sufficient number of coefficients was transmitted. On the other hand, this IDCT block cannot be included as a candidate in the GA since in this case, with the proposed metric for fitness evaluation, the algorithm would converge towards this block all the time. Therefore, to determine whether the GA yielded a good SI candidate, the fitness F_{GA} of the fittest chromosome is compared to a threshold F_T: if $F_{GA} > F_T$, the system takes the GA output as side information; otherwise, the system opts for the co-located block in the IDCT frame. This substitution by the IDCT block using the threshold F_T can also be applied in the simple fusion technique presented in section 6.2.1.

The performance of the GA is greatly affected by its different parameters. It can be clearly seen that, by moving from one iteration to the next, possible solutions that have a greater fitness value have a greater chance to survive, whereas candidate solutions having the least

fitness values are eliminated. Therefore, the fitness function is a major factor that influences the quality of the final SI. The number of iterations I_{max} required for the GA to converge depends on the population size S_p and the number of surviving candidates S_f in each iteration. I_{max} defines the number of crossover points, or in other words, the number of regions to be combined from the co-located blocks in the different candidate SI frames initially generated at the decoder.

When the GA operates at the frame level, entire frames are considered in the genetic operations instead of individual blocks. However, since crossover and mutation influence a small number of blocks in this case (in every iteration), the DCT coefficients (necessary for fitness evaluation) need to be updated only for the blocks where a change has occurred. This leads to a simpler algorithm compared to the block-level GA. A comparison of the computational load of the two approaches will be presented in section 6.4 .

6.3 Simulation Results

In our simulations, we consider 300 frames from the Carphone, Foreman, and News QCIF video sequences, and 150 frames from the Trevor sequence, sampled at a rate of 30 fps, and a WZ GOP size of 2 (i.e. a total of 149 WZ frames). The hash word transmitted to the decoder consists of $(1/8)^{th}$ of the WZ frame's DCT coefficients, computed for blocks of dimensions 16x16 pixels. In the first experiments, the crossover is performed only vertically and the parameters for the fitness function are set as $K_1 = 1$, $K_2 = 0$, and $K_3 = 1$ (i.e., the fitness function is $F = 1 / D$). As for the GA parameters, the following set was determined experimentally after intensive simulations: $\{S_p = 60, S_f = 40, I_{max} = 10, P_c = 0.8, P_m = 0.01\}$.

Since the frame fusion technique presented in Section 6.2.1 operates at the block level, it will be evaluated later and compared with the GA operating at the block level. Therefore, we first begin by evaluating the frame-level GA.

In Figure 6.3 to Figure 6.6, we show the cumulative density function (CDF) of the difference in PSNR (Δ_{PSNR}) between the GA (without using the IDCT in high-motion areas) on one side, and each of the different interpolation techniques on the other, for Carphone, Foreman, News and Trevor sequences, respectively. We notice that for all these sequences, a significant gain in PSNR is obtained most of the time. For example, with the Trevor sequence, we notice a performance gain for more than 85% of frames. A gain that exceeds 1dB is observed for

approximately 70% of the frames compared to AVI, 15% compared to MCI, and 5% compared to HMCI. The percentage of frames with a gain exceeding 1 dB becomes 50%, 25% and 12%, in the Carphone sequence, 78%, 40% and 2% in the Foreman sequence, and 75%, 75% and 40% in the News sequence, compared to AVI, MCI and HMCI, respectively.

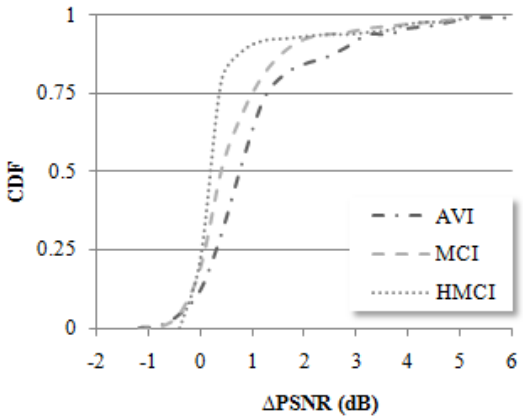


Figure 6.3 CDF of Δ_{PSNR} for the Carphone sequence, with a frame-level GA and $F_T = 0$.

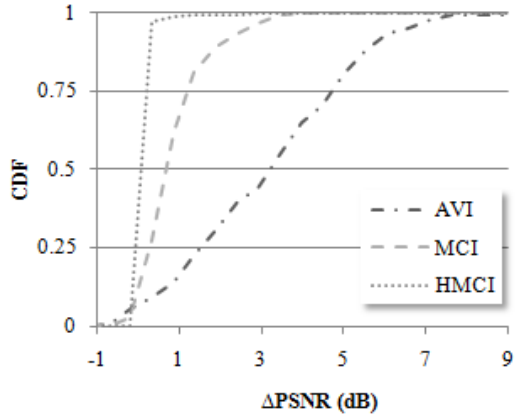


Figure 6.4 CDF of Δ_{PSNR} for the Foreman sequence, with a frame-level GA and $F_T = 0$.

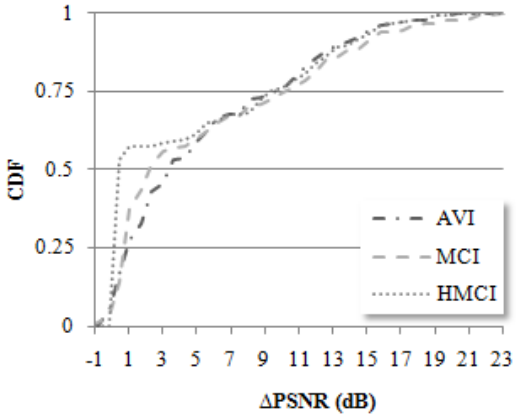


Figure 6.5 CDF of Δ_{PSNR} for the News sequence, with a frame-level GA and $F_T = 0$.

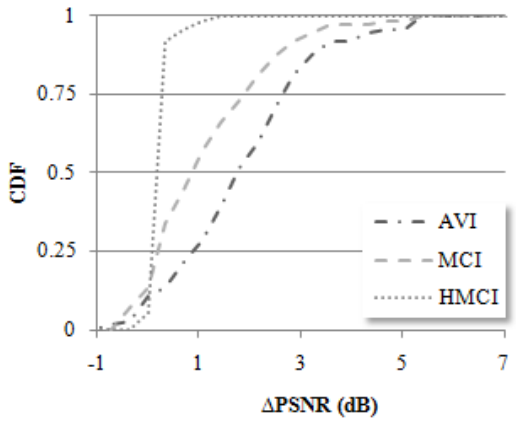


Figure 6.6 CDF of Δ_{PSNR} for the Trevor sequence, with a frame-level GA and $F_T = 0$.

Similar curves are shown in Figure 6.7 to Figure 6.10 for the case where the IDCT frame is used instead of the interpolated frames in high motion areas, with $F_T = 0.03$. It can be noticed that the number of frames with a gain greater than 1 dB increases by 15% to 40% for Carphone, and by 8% to 20% for Foreman. No significant improvement was observed for News and Trevor, since they generally present lower levels of motion compared to Foreman and Carphone.

On the other hand, a slight performance degradation can be noticed with the GA, where a performance loss of less than 1 dB is observed for less than 5% of the frames, for all four sequences.

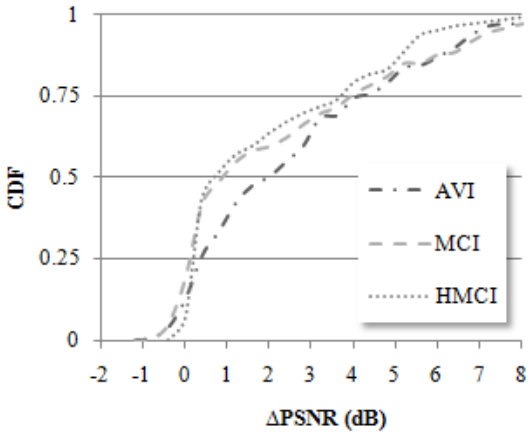


Figure 6.7 CDF of Δ_{PSNR} for the Carphone sequence, with a frame-level GA and $F_T = 0.03$.

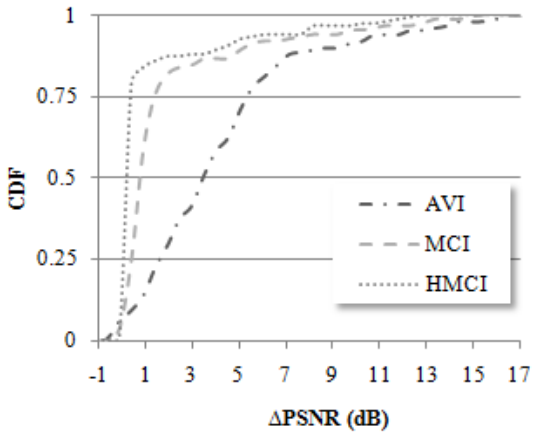


Figure 6.8 CDF of Δ_{PSNR} for the Foreman sequence, with a frame-level GA and $F_T = 0.03$.

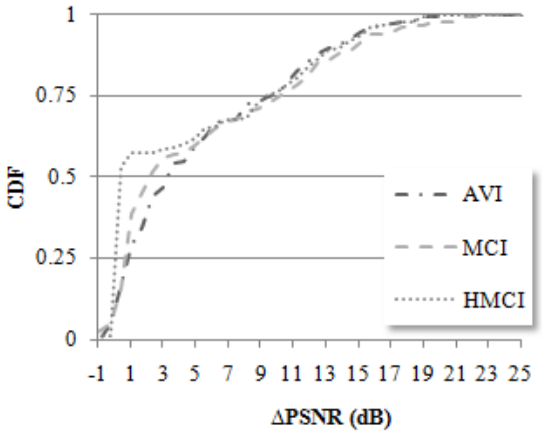


Figure 6.9 CDF of Δ_{PSNR} for the News sequence, with a frame-level GA and $F_T = 0.03$.

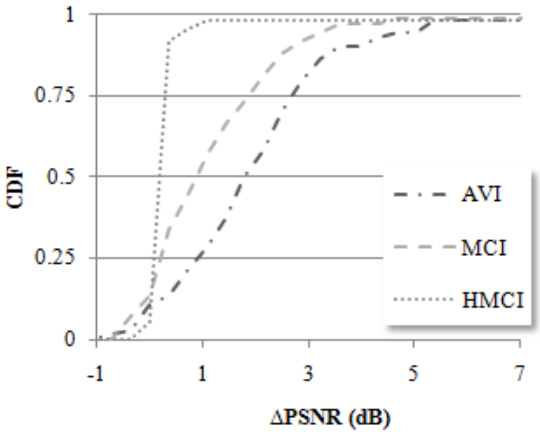


Figure 6.10 CDF of Δ_{PSNR} for the Trevor sequence, with a frame-level GA and $F_T = 0.03$.

Now that we have analyzed the behavior of the proposed frame-level GA, we will evaluate next the block-level GA along with the simple fusion technique (SFT) presented in Section 6.2.1.

In Figure 6.11, we show the PSNR variations along the Carphone sequence obtained with the block-level GA and the different interpolation techniques, for WZ frames 60 to 110 (we limited the plot to this interval for clarity of visualization). The same parameters of the frame-level GA are used. We also show the PSNR for the 2D-GA and SFT. However, with the 2D-GA, since every couple of chromosomes can yield up to 4 offsprings (instead of 2), we doubled S_p and S_f (S_p = 120 and S_f = 80) while maintaining the same ratio S_p/S_f = 2/3. Additionally, we noticed that an improved performance can be obtained by modifying the values of K_1, K_2 and K_3. In fact, when $K_1 = 1$, $K_2 = 0$, and $K_3 = 1$ (i.e. $F = 1 / D$), a small value of D, for a given chromosome, results in a very large value of F, and the chromosome will dominate the next population after the selection and duplication phases. As a result, the GA will quickly converge, which is not desired in our case since the distance measure in use is not an exact indicator of similarity. Even though this phenomenon does not occur very often, it motivated us to investigate different values of the parameters K_1, K_2 and K_3. After a certain number of tests, we found that $K_1 = 10$, $K_2 = 100$ and $K_3 = 1$ yield a significant improvement for a certain number of the video frames. For example, in the case of the News sequence, an improvement greater than 0.5 dB is noticed for almost 10% of the frames.

The superior performance of the GA (and 2D-GA) can be observed all the time, particularly in high motion areas (where the PSNR drops after frame 90), except in few regions where its PSNR converges toward the best among the other curves.

To further improve our system's performance, we use the IDCT instead of interpolated frames in high motion areas, as was explained earlier, with $F_T = 0.02$. A "+" sign is shown next to the frame fusion technique in this case (2D-GA+ and SFT+). Figure 6.12 and Figure 6.13 show the percentage of blocks fused from the AVI, MCI, and HMCI interpolated frames, from the previous (PRV) and next (NXT) key frames, and from the IDCT (in case of SFT+), for each of the video sequences, with SFT and SFT+ respectively. It can be noticed that with SFT+, 17%, 11%, 4% and 5% of the blocks are obtained from IDCT for Carphone, Foreman, News and Trevor sequences, respectively, which yields significant performance improvement compared to SFT as shown in Figure 6.11. Note that both techniques rely largely on AVI since it yields better results in near-static areas (e.g. non moving objects,

static background, texture, etc…) that usually occupy a significant amount of a frame's surface. The News sequence relies the least on IDCT, since, in this sequence, the high motion is confined in very small areas, as will be shown later.

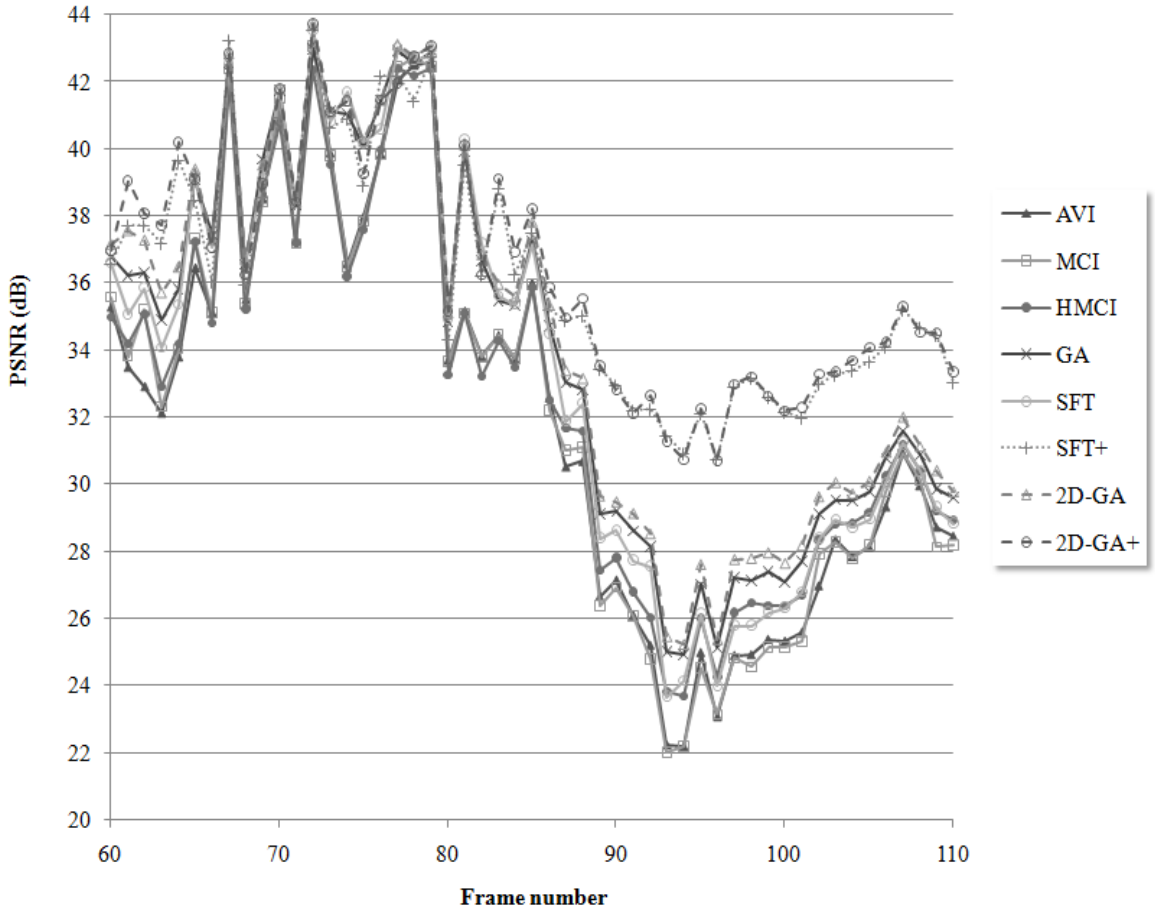


Figure 6.11 PSNR variations along the Carphone sequence.

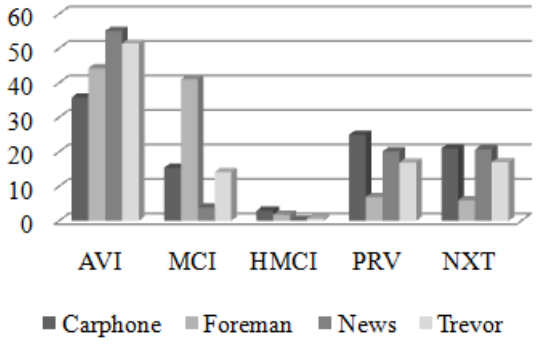


Figure 6.12 Percentage of blocks fused from different candidate SI frames using SFT.

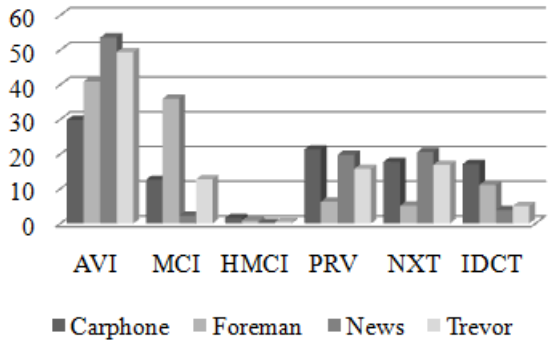


Figure 6.13 Percentage of blocks fused from different candidate SI frames using SFT+.

On the other hand, it can be observed (in Figure 6.11) that very close performance is obtained with SFT+ and 2D-GA+, except in few regions where the 2D-GA+ presents a superior performance that could exceed 1 dB for some frames. In fact, the 2D-GA significantly outperforms the SFT in high motion areas, while in the case of 2D-GA+ and SFT+, they both rely on the IDCT in this case and thus, they yield very close performance most of the time.

Figure 6.14 through Figure 6.17 show the cumulative density function of Δ_{PSNR} between the block-level one-directional GA on one side, and each of the different interpolation techniques (AVI, MCI, and HMCI) on the other, for Trevor, Foreman, Carphone and News sequences, respectively. We notice that for all these sequences, a significant gain in PSNR is obtained most of the time. For example, with the Trevor sequence, we notice a performance gain for more than 97% of frames. A gain that exceeds 1dB is observed for approximately 85% of the frames compared to AVI, 62% compared to MCI, and 14% compared to HMCI. The percentage of frames with a gain exceeding 1 dB becomes 86%, 78% and 62%, in the Carphone sequence, 77%, 42% and 10% in the Foreman sequence, and 80%, 80% and 55% in the News sequence, compared to AVI, MCI and HMCI, respectively. These statistics show an improved performance of the block-level GA compared to the frame-level GA. In fact, the block-level GA better adapts to local image statistics, at the expense of increased complexity.

As for the occasional performance degradation experienced by the GA, a loss smaller than 1 dB is observed for less than 5% of the frames in Trevor, Foreman and News, while no significant loss was observed with Carphone.

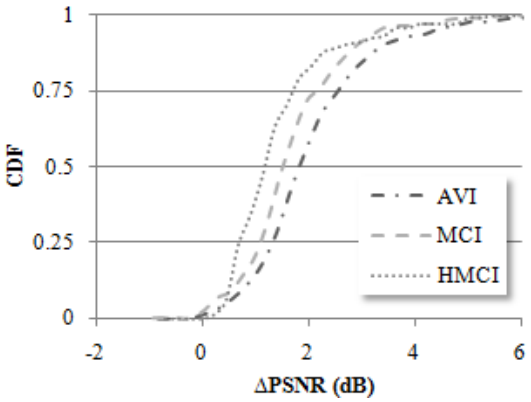


Figure 6.14 CDF of Δ_{PSNR} for the Carphone sequence, with a block-level GA and $F_T = 0$.

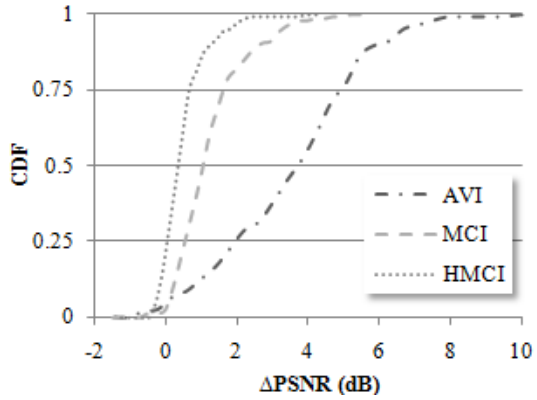


Figure 6.15 CDF of Δ_{PSNR} for the Foreman sequence, with a block-level GA and $F_T = 0$

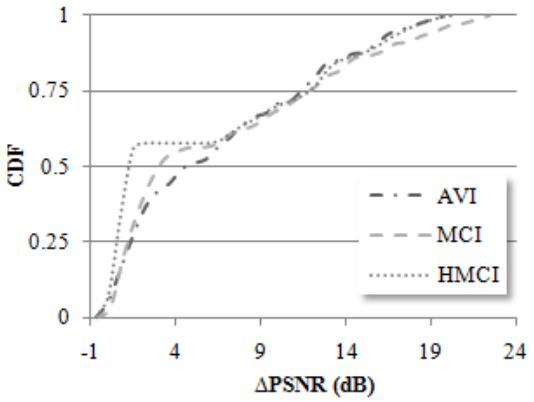


Figure 6.16 CDF of Δ_{PSNR} for the News sequence, with a block-level GA and $F_T = 0$

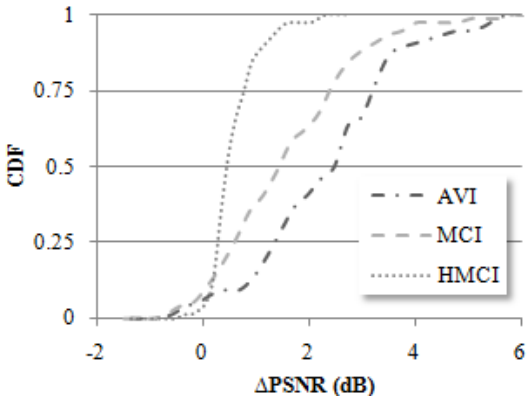


Figure 6.17 CDF of Δ_{PSNR} for the Trevor sequence, with a block-level GA and $F_T = 0$.

In Table 6.1, we show the average and maximum gains obtained with the one-directional GA for the different video sequences. Minimal values can be observed with HMCI, since it offers the best average performance compared to AVI and MCI. It is important to note that these PSNR gains for the side information may not necessarily be the same in terms of the RD performance of a distributed video codec. However, they reflect, to a large extent, a significant improvement in the SI quality, which is expected to yield an improved RD performance too, as will be shown later.

	AVI	MCI	HMCI
Carphone	(2.0, 6.8)	(1.6, 5.5)	(1.4, 5.4)
Foreman	(3.5, 9.8)	(1.2, 5.2)	(0.4, 3.8)
News	(6.5, 19.7)	(6.6, 21.9)	(5.7, 19.7)
Trevor	(2.3, 5.5)	(1.6, 5.6)	(0.6, 2.2)

Table 6.1 Average and maximum PSNR gains (in dB) obtained with the GA for the different video sequences.

Figure 6.18 shows the CDF of the difference in PSNR between the GA+ output on one hand, and the best among the IDCT frame and the different interpolation techniques (AVI, MCI and HMCI) used, on the other. We notice that for all the sequences, a significant gain in PSNR is obtained most of the time. For example, with the Carphone and Trevor sequences, we notice a performance loss that does not exceed 1 dB for less than 10% of the frames, whereas a gain in PSNR is observed for the remaining 90% frames. Approximately 65% of the frames have a gain that exceeds 1 dB, and for some frames, the gain reaches 5.5 dB with Carphone, whereas with Trevor, near 50% of the frames have a gain that exceeds 1 dB, and the gain could reach 4 dB. As for the Foreman sequence, 18% of the frames have a performance degradation that

does not exceed 1 dB, and only 2% suffer from a performance loss between 1 dB and 2 dB. On the other hand, a performance gain (up to 5 dB) is observed for 80% of the frames. Similarly, in the News sequence, a small subset (approximately 10%) of the frames suffers from a slight performance loss, while the gain exceeds 3 dB for more than half of the frames. With the News sequence, the gain exceeds 10 dB for 30% of the frames, and can even reach 20 dB (the curve was cropped at 10 dB for clarity).

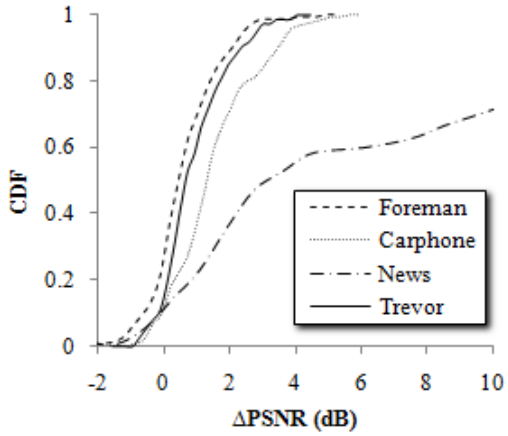


Figure 6.18 CDF of Δ_{PSNR} for different sequences, with a block-level GA+ and $F_T = 0.02$.

In the above discussions, we notice that the genetic algorithm (whether a frame level or block-level GA or GA+), behaves surprisingly well with the News sequence, where the maximum gain exceeds 19 dB. This is due to the very particular nature of the motion in this sequence: the scene contains two almost static (slowly moving) speakers in the foreground, and a couple of dancers in the background, alternating slow and fast movements. Figure 6.19 shows a sequence of consecutive WZ frames from the news sequence. The top row shows the original WZ frames, and the second row their respective side information obtained using the GA. The third row shows the local MSE (for each block) of the SI frame represented as a grayscale image. The last two rows show the SI frame PSNR obtained with the GA and the HMCI technique. Note that block MSE values (in the figures of the third row) are quantized to 256 gray levels (8 bits), where a lighter intensity indicates a greater error. It can be seen that the dominant error (light regions) is mainly concentrated in the background scene (the dancing couple). The error increases every other frame due to the alternating nature of the movement (slow/fast), which can also be seen in the fluctuations of the PSNR (and the gain towards HMCI) that significantly drops every other frame.

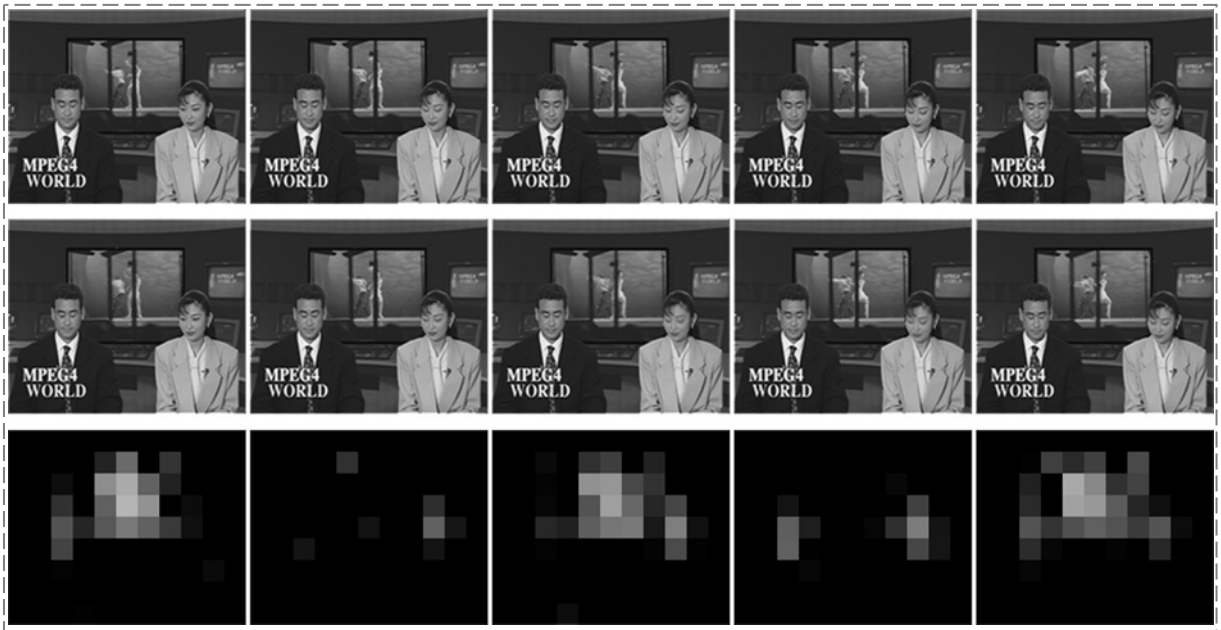


Figure 6.19 Snapshot from the News sequence, showing WZ frames 111 to 115. Top: original frame. Middle: side information obtained with GA. Bottom: error mask (MSE) and SI frame PSNR.

In the aim of evaluating our proposed frame fusion algorithms in terms of RD performance, we encoded the above mentioned sequences using our WZ video codec, presented in the previous chapters, but without feedback channel suppression. In Figure 6.20 through Figure 6.23, we show the average PSNR of the decoded WZ frames as a function of the average WZ bit rate. Since hash-based DVC [AAR04-2, ASC07] outperforms traditional WZ codecs, our results are compared with the case where HMCI is used for generating the side information at the decoder. Given that the same hash is transmitted in all cases (HMCI, SFT+ and block based 2D-GA+), the rate overhead for the transmission of the hash was not considered. Otherwise, all curves would be shifted by the same amount, which is almost 12% of the WZ DCT coefficients, in case the hash bits are transmitted without compression.

As expected, the proposed frame fusion techniques yield significant improvements compared to HMCI. In the case of SFT+, the average gain reaches approximately 3 dB with Carphone, 1 dB with Foreman and Trevor, and 6 dB with News. More performance improvements are obtained with the genetic algorithms, where 2D-GA+ yields additional 0.3 dB for Carphone

and Foreman, 0.6 dB for Trevor, and 1 dB for News. Note that these gains are averaged over the complete sequence and may not reflect the real potential of using a GA in frame fusion. In fact, GAs present exceptional performance in particular situations, as was shown in the case of the News sequence.

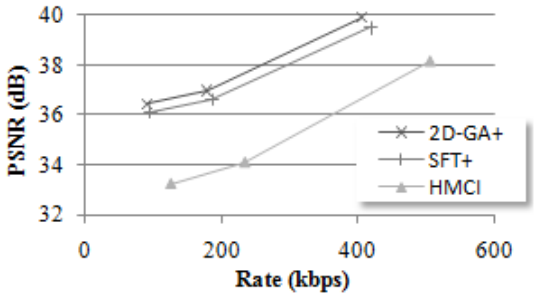


Figure 6.20 RD curves for the Carphone sequence with HMCI, SFT+, and 2D-GA+.

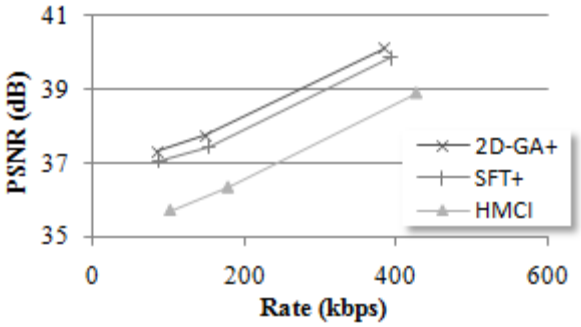


Figure 6.21 RD curves for the Foreman sequence with HMCI, SFT+, and 2D-GA+.

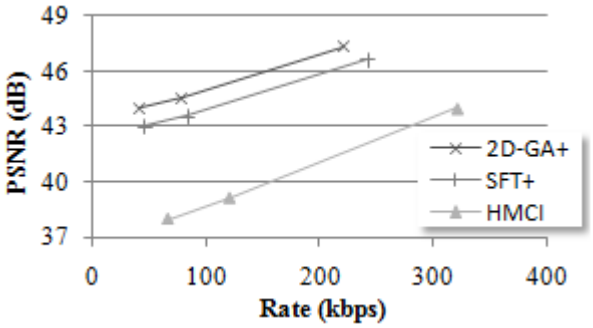


Figure 6.22 RD curves for the News sequence with HMCI, SFT+, and 2D-GA+.

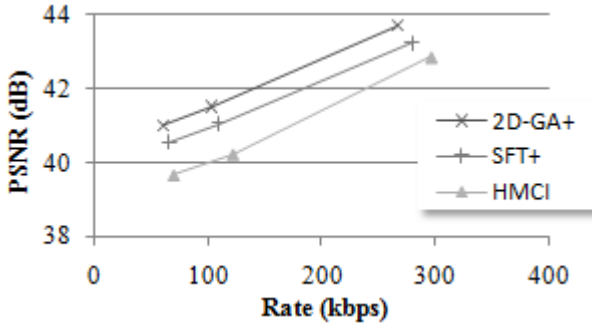


Figure 6.23 RD curves for the Trevor sequence with HMCI, SFT+, and 2D-GA+.

By analyzing the Trevor sequence, we noticed that it consists of two different scenes, subsequent in time, where the first ends at the 29th WZ frame and the second starts at the 31st. The 30th WZ frame is a transitional frame between the two scenes and is thus omitted from the below analysis. Figure 6.24 shows a snapshot from each of these scenes.

In the first part of the sequence, we can observe six different sub-windows each showing a different play, whereas in the second part, one play occupies the whole screen. Figure 6.25 and Figure 6.26 show the RD curves independently for each of the two parts of the video sequence. In the former (WZ frames 1 to 29), 2D-GA+ outperforms SFT+ where the average gain reaches 1 dB, whereas in the latter (i.e. WZ frames 31 to 74) the average gain with respect to SFT+ is only 0.3 dB.

Figure 6.24 Snapshot from the Trevor sequence.

As a result, it can be concluded that the application of the GA is particularly advantageous in the case of complex video scenes containing parts with different levels of motion. By adding an intelligent step to the proposed technique for the analysis of the varying video content, the decoder would be able to decide, for each block, whether to apply the GA (that would significantly improve the performance towards to the simple fusion approach), or to simply rely on the SFT (to avoid the additional computation burden incurred by the GA).

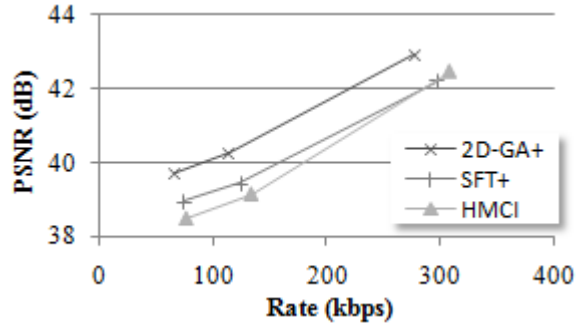


Figure 6.25 Average RD curves for the first 29 WZ frames in the Trevor sequence.

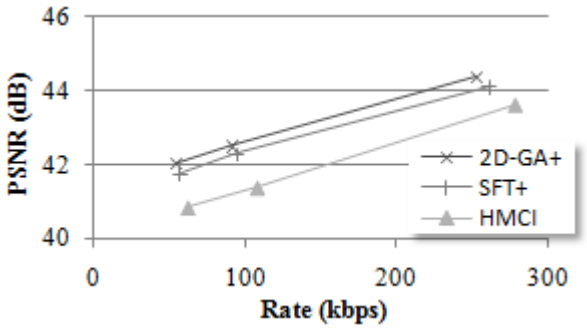


Figure 6.26 Average RD curves for WZ frames 31 to 74 in the Trevor sequence.

6.4 Complexity issues

From a complexity point of view, the proposed fusion techniques incur no additional complexity at the encoder side, in case of hash-based DVC, since the same transmitted hash word can be used at the decoder side for improving the side information using SFT or GA-based frame fusion. However, some additional computations are required at the decoder side.

Let N represent the number of candidate SI frames initially generated at the decoder using previously developed interpolation techniques, PxQ (pixels) represent the dimensions of a frame, and B^2 (pixels) the dimensions of a block. Therefore, a frame contains PQ/B^2 blocks.

In case of SFT, 1 DCT and 1 SSE are evaluated for each block in each of the N candidate frames, which results in NPQ/B^2 DCT and NPQ/B^2 SSE evaluations for one SI frame output.

In case of a block-level GA (assuming only vertical crossover, for simplicity), in addition to the operations performed for the case of SFT, the DCT, SSE and fitness function are evaluated for each of the S_p blocks in each of the I_{max} iterations, in case a crossover or a mutation has occurred. Therefore, $(P_c+P_m)S_pI_{max}PQ/B^2$ DCT, $(P_c+P_m)S_pI_{max}PQ/B^2$ SSE and

$(P_c+P_m)S_pI_{max}PQ/B^2$ fitness evaluations are performed, with $P_cS_pI_{max}PQ/B^2$ crossovers and $P_mS_pI_{max}PQ/B^2$ mutations for each single SI frame output.

In case of a frame level GA (also assuming only vertical crossover), only Q/B blocks (per chromosome and per iteration) are modified due to crossover or mutation, and the fitness is evaluated once. As a result, $(P_c+P_m)S_pI_{max}Q/B$ DCT, $(P_c+P_m)S_pI_{max}Q/B$ SSE and $(P_c+P_m)S_pI_{max}$ fitness function are computed, in addition to $P_cS_pI_{max}$ crossovers and $P_mS_pI_{max}$ mutations for a single SI frame output. Figure 6.27 shows an example for $Q = 12$ and $B = 4$. A vertical crossover occurred at the center block and one of the resulting offsprings is shown in Figure 6.27-a. The top and bottom rows of blocks are exact copies obtained from parent chromosomes, and their DCT coefficients remain unchanged, whereas the middle blocks (patterned blocks in Figure 6.27-b) are the result of a combination of parent genes. Therefore, DCT coefficients need to be updated only for the marked region ($Q/B = 3$ blocks) in Figure 6.27-b.

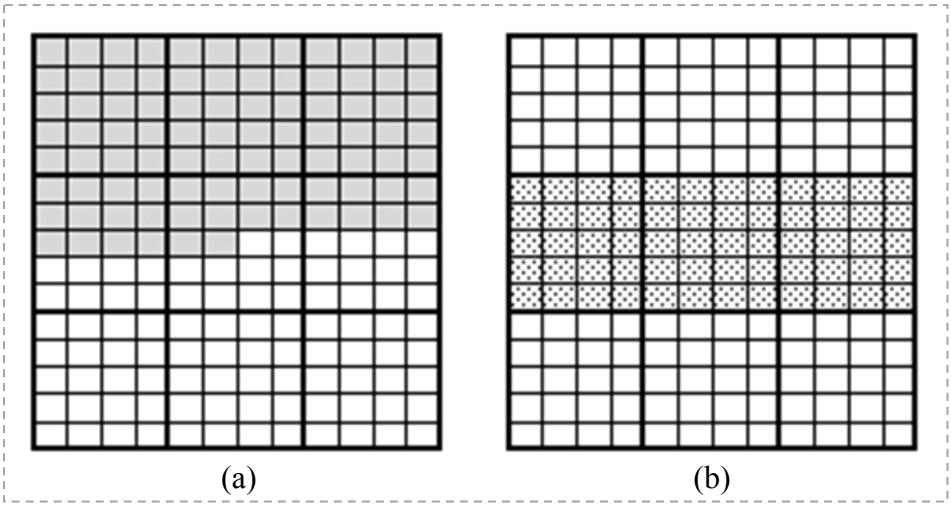


Figure 6.27 (a) Offspring obtained after a frame-level vertical crossover that occurred at the center block. (b) Patterned region marked for DCT update.

In our simulations, PxQ = 144x176 (since we consider QCIF video sequences), $N = 5$, $B = 16$, $S_p = 60$, $I_{max} = 10$, $P_c = 0.8$ and $P_m = 0.01$. Table 6.2 summarizes the number of operations required for each case.

As it can be seen in Table 6.2, frame-level (resp. block-level) GA necessitates more than 10 times (resp. 100 times) the number of operations performed with SFT. Therefore, it may seem computationally inefficient to run the proposed fusion techniques at the decoder, due to the additional complexity they incur. However, with the recent advances in hardware

technology and parallel computing techniques, the additional decoding complexity becomes feasible. Furthermore, in order to limit the decoding complexity, the GA parameters S_p, I_{max}, P_m and P_c can be tuned accordingly. Additionally, a stopping criterion (e.g. the fitness value of the fittest chromosome remaining constant for several consecutive iterations) can be used to stop the GA before reaching the maximum number of iterations, thus reducing the total number of operations performed to obtain the result.

Operation	DCT	SSE	Fitness	Crossover	Mutation
SFT	495	495	0	0	0
Block-level GA	48609	48609	48114	47520	594
Frame-level GA	5401	5401	486	480	6

Table 6.2 Computational load for each of the proposed frame fusion techniques.

6.5 Conclusions and future perspectives

In this chapter, we developed novel techniques for improving the side information in distributed video coding. The proposed solutions rely on frame-fusion techniques, and especially on genetic algorithms, that make use of different interpolation techniques available in the literature, in order to generate an enhanced version of SI. The algorithm runs at the decoder and incurs no additional complexity at the encoder side, in case of hash-based DVC codecs. The superior performance of the GA is especially observed in complex video scenes containing parts with different levels of motion.

This study presents the beginning of a new area for investigation and research. While our preliminary results show a great potential of using GA in the context of frame-fusion for enhancing the SI, much effort is still to be done in order to develop a more mature solution. In the following, we discuss some of our visions for the future of this study.

Since a hash word is constructed at the encoder using DCT coefficients, it would be more practical to use a transform domain WZ codec with the proposed frame fusion algorithms, instead of the pixel domain codec that we developed. For this purpose, we are currently developing a transform domain codec that encloses all the proposed frame fusion techniques.

In our simulations, we only considered a GOP size of 2. Additionally, the set of the GA parameters that yields good results was determined experimentally. Therefore, we intend to

evaluate the proposed frame fusion techniques with larger GOP sizes, and also to optimize the parameter set so as to further enhance the GA performance.

On the other hand, we noticed that the performance of the GA is highly influenced by the fitness function used for evaluating the quality of the candidate SI. Therefore, the search for a more reliable metric would be an interesting topic to be investigated. The desired metric should require a minimal amount of hash information and, at the same time, provide an accurate measure of similarity between the candidate SI and the initial WZ frames (or blocks). The hash information needs not necessarily to be constructed from DCT coefficients. An ideal hash word would present a precise description of the frame it has been extracted from, with a minimal amount of information to be transmitted. This constitutes another topic for investigation.

In our current implementation, the amount of transmitted hash has been fixed for all the WZ frames, regardless of their content. However, the length of a hash word could be dynamically varied in order to take into account the motion level within a video scene. This dynamic variation of the hash information can also be optimized to take the transmission conditions into consideration, thus resulting in a joint source-channel approach for frame fusion. On the other hand, since the GA does not always improve the quality of the side information, as it was shown earlier in this chapter, some intelligence could be added to the decoder in order to enable or disable the GA as necessary. If additional complexity can be tolerated at the encoder side, such intelligence could be implemented at the encoder in order to determine whether to transmit hash information or not, thus resulting in a reduced amount of transmitted hash bits.

Finally, since several fusion-based techniques [ART06, OUA06, OUA07] have been developed for multiview DVC, our GA approaches can be applied in this context too to enhance SI generation.

Chapter 7

Conclusion

7.1 Research Overview

In this study, we aimed at optimizing the distributed coding of video sources. Therefore, we mainly targeted two major problems in distributed video coding: the suppression of the return channel and the generation of the side information.

After introducing our work in Chapter 1, we started, in Chapter 2, by analyzing the performance of binary and non-binary turbo-codes in channel coding as well as in distributed source compression. We showed that duo-binary turbo-codes can offer the best tradeoff between decoding complexity and error correction capability. However, for applications requiring a very low BER, quadri-binary codes offer the best performance if sufficient resources are available at the decoder.

In Chapter 3, we investigated the use of turbo-codes as a tool for joint compression and error protection. We extended the study of the previous chapter for the case where turbo-codes were used for the compression of distributed sources transmitted over noisy channels. Besides, we considered joint source-channel coding from a networking point of view, where it can be seen as a cross-layer optimization approach. A video streaming system was developed as a sample application, where turbo-codes were used for channel coding, taking into account the channel conditions at the link layer and the distortion at the application layer.

Then, we presented, in Chapter 4, implementation details for a pixel-domain distributed video coding system based on quadri-binary turbo-codes. We also derived the lower compression bound for Wyner-Ziv frames for the case of error-free transmission, as well as for the case of error-prone channels. We showed that the analytical estimations can be used in a broadcasting system to predict the compression level for each frame, with a minor loss in the decoding PSNR, compared to the classical feedback-based coding system.

Based on our study in Chapter 4, we presented in Chapter 5 two different applications of feedback channel suppression in DVC. First, we developed an algorithm for dynamic rate allocation and adaptive quantization for multi-user WZ coding. The proposed a novel technique that takes into account the channel conditions for each user's transmission, and the motion level in the captured video scenes, in order to unequally distribute channel resources among users, without the excessive need for a feedback channel. Once each user is assigned a transmission rate, adaptive quantization aims at selecting the optimal quantization parameter for the target bit rate. Simulation results showed significant improvement over a traditional coding system with equal allocation of channel resources and fixed quantization parameter. Then, we developed novel algorithms for adaptive GOP size control in Wyner-Ziv video coding, without the need of a return channel. In high motion areas where WZ coding would usually fail, our algorithms automatically switch to H.264 intra-coding mode. On the other hand, when WZ coding outperforms intra-coding, the proposed algorithms determine the ideal GOP size for optimal decoding performance.

Finally, in Chapter 6, we aimed at improving the generation of the side information in DVC. Since there exists several interpolation techniques in the literature, we did not develop a new interpolation method, but rather tried to make benefit from these available techniques, in a fusion-based approach, by selecting the best among the different versions of the side information, for every region within a frame. Therefore, a simple fusion technique was first presented, and a novel technique based on genetic algorithms then developed, without additional encoding complexity in the case of hash-based WZ coding. Genetic algorithms showed exceptional performance in complex video scenes containing parts with different levels of motion. The work presented in this chapter leaves an open topic for discussion and future research, in order to develop more mature solutions for enhancing the side information in WZ video coding.

7.2 Perspectives

At the end of this manuscript, we would like to present a brief discussion on several research points that seem interesting to investigate in the future.

- Since transform-domain coding yields improved performance compared to pixel-domain DVC, with a marginal complexity increase at the encoder, the scope of this study could be extended to the case of transform-domain Wyner-Ziv video coding.

- In Chapter 3, a video streaming system was designed as a sample application of cross-layer optimization based on turbo-codes. However, a simplified network model was used during the design phase and in the simulations. It would be interesting to design a complete solution for a realistic network model, where the number of users would be greater than three, and the complete protocol stack would be taken into account, depending on the application where the system would be incorporated.

- In our dynamic rate allocation technique presented in Chapter 4, we considered joint source-channel coding from a signal processing point of view, where the presence of a protocol stack in practical applications was not considered. It could be argued that our system could be implemented at the application level in a realistic network, and the lower layers (in the protocol stack) would remain unchanged. However, in this case, redundant channel coding would be performed (at different layers). On the other hand, if only source coding would be performed at the application layer, and channel coding at one of the lower layers, the implementation of our solution would require a cross-layer communication along with new network communication protocols that can handle the exchange of messages between the users and the base station accurately, taking into account limited bandwidth restrictions and real time transmission constraints. This makes an important research topic that could pave the way towards a practical implementation of Wyner-Ziv video coding in current and future generations of communication networks.

- In the proposed algorithms for adaptive GOP size control, we tackled the problem of optimizing the GOP size selection and the Intra/WZ mode decision without considering the effect of error-prone transmissions. These algorithms could be extended to take into account channel impairments, as was done with the multiuser application that we developed in Chapter 4.

- In Chapter 6, we listed some headlines for the continuity of the research regarding genetic algorithms and their application in DVC. Parameter selection could be optimized and dynamically varied along the sequence. Hash-word extraction could be enabled or disabled in the encoder as necessary (depending on whether the current block contains significant movement or not), and the hash-word length could be varied depending on the motion level within the video sequence. It would also be interesting to implement and evaluate the performance of genetic algorithms with

several directional crossovers. Moreover, it would be important to investigate the use of genetic-based frame fusion algorithms for improving the side information in multiview DVC, where the fusion of multiple side information is even a more common approach than in monoview DVC. Even though additional complexity can be tolerated at the decoder side, the increased complexity incurred by the genetic algorithms could constitute a problem for real-time applications. Therefore, reducing the complexity of genetic algorithms in the context of DVC would be an interesting research topic for investigation.

Publications

Published Journal Papers

[1] C. Yaacoub, J. Farah, B. Pesquet-Popescu, "New Adaptive Algorithms for GOP Size Control with Return Channel Suppression in Wyner-Ziv Video Coding". *International Journal of Digital Multimedia Broadcasting (IJDMB), special issue on Advances in Video Coding for Broadcast Applications*. 2009. Vol.2009, open access paper.

[2] C. Yaacoub, J. Farah, 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. Vol.2008, open access paper.

Published Conference Papers

[1] C. Yaacoub, J. Farah, B. Pesquet-Popescu, "Optimal Rate Allocation in Multi-User Wyner-Ziv Video Coding Systems with Coded Key Frames", *19th Annual IEEE International Symposium on Personal, Indoor & Mobile Radio Communications,* PIMRC'08, Cannes, France, September 2008.

[2] C. Yaacoub, J. Farah, B. Pesquet-Popescu, "A Novel Technique for Practical Implementation of Pixel-Domain Wyner-Ziv Video Coding in Multi-User Systems", *16th European Signal Processing Conference*, EUSIPCO'08, Lausanne, Switzerland, August 2008.

[3] C. Yaacoub, J. Farah, B. Pesquet-Popescu, "A Cross-Layer Approach with Adaptive Rate Allocation and Quantization for Return Channel Suppression in Wyner-Ziv Video Coding Systems", *3rd IEEE International Conference on Information and Communication Technologies, form Theory to Applications*, ICTTA'08, Damascus, Syria, April 2008.

[4] C. Yaacoub, J. Farah, B. Pesquet-Popescu, "Dynamic Rate Allocation with Variable Quantization in Multi-Sensor Wyner-Ziv Video Coding Systems", *3rd IEEE International Symposium on Communications, Control and Signal Processing*, ISCCSP'08, St. Julians, Malta, March 2008.

[5] C. Yaacoub, J. Farah, B. Pesquet-Popescu, "Joint Source-Channel Wyner-Ziv Coding in Wireless Video Sensor Networks", *7th IEEE International Symposium on Signal Processing and Information Technology*, ISSPIT'07, Cairo, Egypt, December 2007.

[6] C. Yaacoub, J. Farah, N. Rachkidy, B. Pesquet-Popescu, "A Cross-Layer Approach for Dynamic Rate Allocation in H.264 Multi-User Video Streaming", *14th IEEE International Conference on Electronics, Circuits and Systems*, ICECS'07 , Marrakech, Morocco, December 2007.

[7] J. Farah, C. Yaacoub, F. Marx, B. Pesquet-Popescu, "Analyse des performances d'un système de compression distribuée de séquences vidéo transmises sur un lien non fiable", *21e Colloque GRETSI*, GRETSI'07, Troyes, France, September 2007.

[8] J. Farah, C. Yaacoub, F. Marx, B. Pesquet-Popescu, "Performance Analysis of a Distributed Video Coding System - Application to Broadcasting over an Error-Prone Channel", *15th European Signal Processing Conference*, EUSIPCO'07, Poznan, Poland, September 2007.

[9] C. Yaacoub, J. Farah, N. Rachkidy, B. Pesquet-Popescu, "Cross-Layer Optimization for Dynamic Rate Allocation in a Multi-User Video Streaming System", *1st International Conference on New Technologies, Mobility and Security*, NTMS'07, Paris, France, May 2007. (Poster)

[10] J. Farah, C. Yaacoub, F. Marx, B. Pesquet-Popescu, "Distributed Coding of Video Sequences Transmitted through Error-Prone Channels", *IEEE 4th International Conference on Sciences of Electronics, Technologies of Information and Telecommunications*, SETIT'07, Hammamet, Tunisia, March 2007.

[11] C. Yaacoub, J. Farah, N. Rachkidy, F. Marx, B. Pesquet-Popescu, "Dynamic RCPT-Based Cross-Layer Optimization for Multi-User H.264 Video Streaming", *2nd International Computer Engineering Conference*, ICENCO'06 , Cairo, Egypt, December 2006.
BEST PAPER in "Network Analysis and Management".

[12] J. Farah, C. Yaacoub, N. Rachkidy, F. Marx, "Binary and non-Binary Turbo Codes for the Compression of Correlated Sources Transmitted through Error-Prone Channels", *4th International Symposium on Turbo Codes*, ISTC'06, and *6th ITG Conference on Source and Channel Coding*, Munich, Germany, April 2006.

Accepted Conference Papers

[1] C. Yaacoub, J. Farah, B. Pesquet-Popescu, "A Genetic Algorithm for Side Information Enhancement in Distributed Video Coding", *IEEE International Conference on Image Processing*, ICIP'09, Cairo, Egypt, November 2009.

[2] C. Yaacoub, J. Farah, B. Pesquet-Popescu, "Content Adaptive GOP Size Control with Feedback Channel Suppression in Distributed Video Coding", *IEEE International Conference on Image Processing*, ICIP'09, Cairo, Egypt, November 2009.

[3] C. Yaacoub, J. Farah, B. Pesquet-Popescu, "Improving Hash-Based Wyner-Ziv Video Coding Using Genetic Algorithms", *5th International Mobile Multimedia Communications Conference*, MOBIMEDIA'09, London, UK, September 2009. **INVITED PAPER**.

[4] J. Farah, C. Yaacoub, B. Pesquet-Popescu, "Nouvelle technique de génération de l'information adjacente en codage vidéo distribué basée sur les algorithmes génétiques", *22e Colloque GRETSI*, GRETSI'09, Dijon, France, September 2009.

[5] J. Farah, C. Yaacoub, B. Pesquet-Popescu, "Nouvelle technique d'adaptation dynamique de la taille du GOP dans le codage Wyner-Ziv des séquences vidéo", *22e Colloque GRETSI*, GRETSI'09, Dijon, France, September 2009.

[6] C. Yaacoub, J. Farah, B. Pesquet-Popescu, "A Genetic Frame Fusion Algorithm for Side Information Enhancement in Wyner-Ziv Video Coding", *17th European Signal Processing Conference*, EUSIPCO'09, Glasgow, Scotland, August 2009.

Bibliography

[3GP00] 3GPP, "Technical Specification Group: Multiplexing and channel coding (TDD)", TS25.222 V3.2.0, Mar. 2000.

[AAR02-1] A. Aaron and B. Girod, "Compression with Side Information Using Turbo Codes", Data Compression Conference, Snowbird, UT, USA, April 2002, pp. 252- 261.

[AAR02-2] A. Aaron, R. Zhang and B.Girod, "Wyner-Ziv Coding of Motion Video", 36th Asilomar Conference on Signals, Systems and Computers, November 2002, pp. 240-244.

[AAR03] A. Aaron, E. Setton and B. Girod, "Towards practical Wyner-Ziv coding of video", IEEE International Conference on Image Processing, Barcelona, Spain, Sept. 2003.

[AAR04-1] A. Aaron, S. Rane, E. Setton and B. Girod, "Transform-domain Wyner-Ziv codec for video", Visual Communications and Image Processing, San Jose, CA, January 2004.

[AAR04-2] A. Aaron, S. Rane and B. Girod, "Wyner-Ziv video coding with hash-based motion compensation at the receiver", IEEE International Conference on Image Processing, Singapore, Oct. 2004.

[ART05] X. Artigas and L. Torres, "Improved signal reconstruction and return channel suppression in Distributed Video Coding systems", 47th International Symposium ELMAR-2005, Croatia, June 2005.

[ART06] X. Artigas, E. Angeli, L. Torres, "Side Information Generation for Multiview Distributed Video Coding Using a Fusion Approach", 7th Nordic Signal Processing Symposium, NORSIG'06, Reykjavik, Iceland, June 2006.

[ASC05-1] J. Ascenso, C. Brites and F. Pereira, "Motion Compensated Refinement for Low Complexity Pixel Based Distributed Video Coding", IEEE International Conference on Advanced Video and Signal-Based Surveillance, Italy, September 2005.

[ASC05-2] J. Ascenso, C. Brites, and F. Pereira, "Improving frame interpolation with spatial motion smoothing for pixel domain distributed video coding", 5th EURASIP Conference on Speech and Image Processing, Multimedia Communications and Services, Smolenice, Slovak Republic, Jun. 2005

[ASC06] J. Ascenso, C. Brites, F. Pereira, "Content Adaptive Wyner-Ziv Video Coding Driven by Motion Activity", IEEE International Conference on Image

Processing, Atlanta, USA, October 2006.

[ASC07] J. Ascenso, F. Pereira, “Adaptive hash-based side information exploitation for efficient Wyner-Ziv video coding”, International Conference on Image Processing, USA, Sept. 2007.

[BAU07] G. Baudoin, J-F. Bercher, C. Berland, D. Courivaud, N. Gresset, G. Lissorgues, P. Jardin, C. Ripoll, O. Venard, J-M. Brossier, M. Villegas, "Radiocommunications numériques" Tome 1, 2Ed, Dunod 2007, pp. 371-388.

[BER03] C. Berrou, “The Ten-Year-Old Turbo Codes are Entering into Service”, IEEE communications magazine, Aug. 2003, pp. 110-116.

[BER93] C. Berrou, A. Glavieux and P. Thitimajshima, “Near Shannon Limit Error-Correcting Coding and Decoding: Turbo-Codes”, International Conference on Communications, Geneva, May 1993, pp. 1064-70.

[BRI06-1] C. Brites, J. Ascenso and F. Pereira, “Feedback Channel in Pixel Domain Wyner-Ziv Video Coding: Myths and Realities”, 14th European Signal Processing Conference, Italy, September 2006.

[BRI06-2] C. Brites, J. Ascenso and F. Pereira, “Modeling Correlation Noise Statistics at Decoder for Pixel Based Wyner-Ziv Video Coding”, Picture Coding Symposium, Beijing, China, April 24-26, 2006

[BRI07] C. Brites and F. Pereira, “Encoder rate control for transform domain Wyner-Ziv video coding”, International Conference on Image Processing, USA, September 2007.

[BUC04] P. Bucciol, G. Davini, E. Masala, E. Filippi, and J.C. De Martin, "Cross-layer perceptual ARQ for H.264 video streaming over 802.11 wireless networks", IEEE Global Telecommunications Conference, November-December 2004, Vol. 5, pp. 3027-3031.

[CHA01] P. H. Chang, J. J. Leou, and H. C. Hsieh “A genetic algorithm approach to image sequence interpolation”, EURASIP Journal on Signal Processing: Image Communication, Vol. 16, No. 6, pp. 507-520, 2001.

[CHO93] K. Hung-Kei Chow, M.L. Liou, "Genetic motion search algorithm for video compression", IEEE Trans. Circuits and Systems for Video Technology, Vol. 3, pp. 440-445, Dec. 1993.

[CHU00] S.-Y. Chung, "On the Construction of Some Capacity-Approaching Coding Schemes", Ph.D. thesis, Massachusetts Institute of Technology, 2000.

[DIV95] D. Divsalar and F. Pollara, "Multiple turbo Codes", Military Communication Conference, 1995, pp. 279-285.

[DOU00] C. Douillard, M. Jezequel, C. Berrou, N. brengarth, J. Tousch, and N. Pahm, “The Turbo Code Standard for DVB-RCS”, 2nd International Symposium On Turbo Codes and Related Topics, Brest, France, Sep. 2000, pp. 535-538.

[GAL68] R. Gallagher, "Information Theory and Reliable Communication", Wiley 1968.

[GAO02] Y. Gao and M.R. Soleymani, "Triple-binary Circular Recursive Systematic Convolutional Turbo Codes", 5th International Symposium on Wireless Personal Multimedia Communications, Oct. 2002, pp. 951-955.

[GOL89] D.E. Goldberg, "Genetic Algorithms: Search, Optimization, and Machine Learning", Addison-Wesley, Reading, MA, 1989.

[HAG96] J. Hagenauer, E. Offer, L. Papke, “Iterative decoding of binary block and convolutional codes”, IEEE Transactions on Information Theory, vol.45, n°2, Mar. 1996, pp. 429-445.

[HAY01-1] S. Haykin, "Communication Systems", 4Ed, Wiley 2001, pp. 611-616.

[HAY01-2] S. Haykin, "Communication Systems", 4Ed, Wiley 2001, pp. 657-660.

[ITU03] ITU-T and ISO/IEC JTC1, "Advanced Video Coding for Generic Audiovisual Services," ITU-T Recommendation H.264 – ISO/IEC 14496-10 AVC, 2003.

[KHA05] S. Khan, M. Sgroi, E. Steinbach, and W. Kellerer, "Cross-Layer Optimization for Wireless Video Streaming - Performance and Cost", IEEE International Conference on Multimedia and Expo, July 2005, pp. 924-927.

[KUB06] D. Kubasov, C. Guillemot, “Mesh-based motion-compensated interpolation for side information extraction in distributed video coding”, International Conference on Image Processing, Atlanta, USA, Oct. 2006

[KUB07] D. Kubasov, J. Nayak and C. Guillemot, “Optimal Reconstruction in Wyner-Ziv Video Coding with Multiple Side Information”, 2007 IEEE International Workshop on Multimedia Signal Processing, Chania, Crete, Greece, October 2007

[KUB07-2] D. Kubasov, K. Lajnef and C. Guillemot, “A hybrid encoder/decoder rate control for a Wyner-Ziv video codec with a feedback channel” , IEEE MultiMedia Signal Processing Workshop, Grece, October 2007.

[LIN96] Chun-Hung Lin and Ja-Ling Wu, "Genetic block matching algorithm for video coding", IEEE International Conference on Multimedia Computing and Systems, Japan, Jun. 1996.

[LIU05] H. Liu, W. Zhang, S. Yu, and J. Cai, "A Client-Driven Scalable Cross-Layer Retransmission Scheme for 3G Video Streaming", IEEE International Conference on Multimedia and Expo, July 2005, pp. 538-541.

[LIV02] A.D. Liveris, Z. Xiong, and C. N. Georghiades, "Joint Source-Channel Coding of Binary Sources with Side Information at the Decoder Using IRA codes", IEEE Workshop on Multimedia Signal Processing, Dec. 2002, pp. 53-56.

[MIS05] K. Misra, S. Karande, H. Radha, "Multi-hypothesis distributed video coding using LDPC codes", 43rd Annual Allerton Conference on Communication,

Control, And Computing, Monticello, USA, September 2005.

[MOR07] M. Morbee, J. Prades-Nebot, A. Pizurica and W. Philips, “Rate Allocation Algorithm For Pixel-Domain Distributed Video Coding Without Feedback Channel”, 32nd International Conference on Acoustics, Speech and Signal Processing, Honolulu, Hawaii, USA, April 2007.

[ORT98] A. Ortega, K. Ramchandran, "Rate-distortion methods for image and video compression", IEEE Signal Processing Magazine, August 2008, pp. 23-50.

[OUA06] M. Ouaret, F. Dufaux, T. Ebrahimi, “Fusion-based multiview distributed video coding”, ACM International Workshop on Video Surveillance and Sensor Networks, Santa Barbara, CA, USA October 27, 2006.

[OUA07] M. Ouaret, F. Dufaux, T. Ebrahimi, “Multiview distributed video coding with encoder driven fusion”, European Conference on Signal Processing (EUSIPCO), Poznan, Poland, September 2007.

[PEN05] Y. Peng, S. Khan, E. Steinbach, M. Sgroi, and W. Kellerer, "Adaptive resource allocation and frame scheduling for wireless multi-user video streaming", IEEE International Conference on Image Processing, Italy, September 2005,Vol. 3, pp. 708-711.

[PUR02] R. Puri and K. Ramchandran, “PRISM: A New Robust Video Coding Architecture Based on Distributed Compression Principles”, 40th Allerton Conference on Communication, Control and Computing, Allerton IL, October 2002.

[QQU04] Q. Qu, Y. Pei, J.W. Modestino, and X. Tian, "Source-adaptive FEC/UEP coding for video transport over bursty packet loss 3G UMTS networks: a cross-layer approach", IEEE 60th Vehicular Technology Conference, September 2004, Vol. 5, pp. 3150-3154.

[QQU05] Q. Qu, Y. Pei, J.W. Modestino, X. Tian, B. Wang, and X. Yu, "Cross-layer rate-adaptation and joint source-channel coding for delay-constrained video streaming over wireless ad-hoc networks", 1st International Conference on Multimedia Services Access Networks, June 2005, pp. 1-5.

[ROB97] P. Robertson, P. Hoeher and E. Villebrun, “Optimal and suboptimal maximum a posteriori algorithms suitable for turbo decoding”, ETT, Vol. 8, Mar.-Apr. 1997, pp. 119-125.

[ROW00] D.N. Rowitch and L.B. Milstein, "On the performance of hybrid FEC/ARQ systems using rate compatible punctured turbo (RCPT) codes", IEEE Transactions on Communications, Vol. 48, June 2000, pp. 948-959.

[SAL06] D. Salomon, "Data Compression: The Complete Reference", 4Ed, Springer 2006.

[SHA48] C. Shannon, “A mathematical theory of communication”, Bell System Technical Journal, vol. 27, July and October 1948, pp. 379-423 and 623-656.

[SKL97] B. Sklar, “A primer on turbo code concepts”, IEEE communications magazine, Dcc. 97, pp. 94-102.

[SLE73] D. Slepian and J.K. Wolf, “Noiseless coding of correlated information sources”, IEEE Transactions on Information Theory, vol. IT-19, July 1973, pp. 471–480.

[TAN03] A.S. Tanenbaum, “Computer Networks”, 4th Ed. Prentice Hall 2003, pp. 37-41.

[TOA07] Toan Nguyen Dinh, GueeSang Lee, June-Young Chang, and Han-Jin Cho, "A Novel Motion Compensated Frame Interpolation Method for Improving Side Information in Distributed Video Coding", International Symposium on Information Technology Convergence, Korea, Nov. 2007.

[TON06] Y. Tonomura, D. Shirai, T. Nakachi, and Tetsuro Fujii, "Optimal Bit Allocation for Wavelet-Based Distributed Video Coding", 8th IEEE International Symposium on Multimedia, USA, December 2006.

[WTU04] W. Tu, W. Kellerer, and E. Steinbach, "Rate-Distortion Optimized Video Frame Dropping on Active Network Nodes," Packet Video Workshop, California, December 2004.

[WYN76] A.D. Wyner and J. Ziv, “The Rate-Distortion Function for Source Coding with Side Information at the Decoder”, IEEE Transactions on Information Theory, Vol. IT-22, January 1976, pp. 1-10.