Codage Scalable et Controle de Congestion pour Transmission Video sur Reseaux Heterogenes

Jérôme Viéron

N^o Ordre : 2819

THÈSE

présentée

DEVANT L'UNIVERSITÉ DE RENNES 1

pour obtenir

le grade de : *DOCTEUR DE L'UNIVERSITÉ DE RENNES 1*

Mention : *Informatique*

PAR

Jérôme VIÉRON

Équipe d'accueil : TEMICS/IRISA/INRIA

École Doctorale : Mathématiques, Informatique, Signal, Électronique, Télécommunications (MATISSE)

Composante universitaire : Institut de Formation Supérieur en Informatique et Communication (IFSIC)

TITRE DE LA THÈSE :

Codage Scalable et Contrôle de Congestion pour Transmission Vidéo sur Réseaux Hétérogènes

Soutenue le 14 avril 2003 devant la commission d'Examen

COMPOSITION DU JURY

M.	RUBINO	Gerardo	Président
M.	MACQ	Benoît	Rapporteurs
M.	BASKURT	Atilla	
Mme	PESQUET-POPESCU	Béatrice	Examinateurs
M.	FRANÇOIS	Edouard	
Mme	GUILLEMOT	Christine	

A Marion, Aurane et Thiméo,

Remerciements

Ce travail a été réalisé à l'IRISA (Institut de Recherche en Informatique et Systèmes Aléatoires) / INRIA Rennes, au sein du projet TEMICS (TraitemEnt, Modélisation d'Images et CommunicationS).

J'aimerais remercier tout spécialement Christine GUILLEMOT, Directrice de recherche à l'INRIA et responsable du projet TEMICS, pour m'avoir permis de réaliser ces travaux de thèse et les avoir encadrés, mais également pour sa disponibilité et sa confiance.

Je remercie M. Gerardo RUBINO, Directeur de recherche à l'INRIA, responsable du projet ARMOR (Architecture et MOdèles de Réseaux), qui m'a fait l'honneur de présider le Jury de cette thèse.

Je remercie également M. Benoît MACQ, Professeur à l'Université Catholique de Louvain, Laboratoire de Télécommunications et Télédétection, et M. Atilla BASKURT, Professeur à l'Université Claude Bernard Lyon 1, Laboratoire d'InfoRmatique en Images et Systèmes d'information, d'avoir bien voulu accepter la charge de rapporteur.

Je remercie Mme Béatrice PESQUET-POPESCU, Maître de conférences à l'Ecole Nationale Supérieure des Télécommunications, Département Traitement du Signal et des Images, et M. Edouard FRANÇOIS, Technical Advisor, Thomson R&D, Laboratoire Multimedia Streaming and Storage, d'avoir accepter de juger ce travail.

Ce séjour à l'IRISA fut également l'occasion de travailler en équipe. A ce titre, je tiens à remercier Fabrice Le Léannec, Xavier Hénocq, Thierry Turletti (INRIA Sophia-Antipolis), Stéphane Pateux et Franck Galpin pour les nombreuses discussions constructives (...je vous assure que c'est arrivé !!) que nous avons eues ensemble, leurs critiques et leur aide.

Je tiens enfin à envoyer un salut amical à tous les membres du projet TEMICS passés et actuels que j'ai croisés durant ces 4 dernières années. Je remercie plus particulièrement Thomas Guionnet (Curves man), Jean-Marie Pinel (Kart man), Guillaume Larignon (Squash man) et Gaëtan Le Guelvouit (Fridge man) pour m'avoir *supporté* au cours de ces années.

Sans l'aide et le soutien de tous, ces travaux de thèse n'auraient jamais pu aboutir.

Table des matières

Table des figures

Glossaire

ACK	*Acknowledgement* Rapport réseau d'acquittement d'un paquet.
AIMD	*Additive Increase Multiplicative Decrease*
CABAC	*Context-based Adaptive Binary Arithmetic Coding* Codeur arithmétique basé contextes utilisé dans H.264 pour coder le mouvement.
CIF	*Common Intermediate Format* Format d'images pour la vidéo.
CQF	*Conjugate Quadrature Filter* Famille de filtres.
DCT	*Discrete Cosine Transform*
DFD	*Displaced Frame difference*
DPCM	*Differential Pulse Coding Modulation* Mode de codage différentiel.
DWT	*Discrete Wavelet Transform*
DSG	*Destination Set Grouping* Algorithme de régulation en simulcast.
EBCOT	*Embedded Block Coding with Optimized Truncation* Algorithme de compression d'images fixes (utilisé dans JPEG-2000).
EQM	*Erreur Quadratique Moyenne*
EZBC	*Embedded ZeroBlocks coding based on Context modeling* Algorithme de compression d'images fixes.
EZW	*Embedded Zerotree of Wavelet* Algorithme de compression d'images fixes.
FEC	*Forward Error Control* Mécanismes de contrôle et de correction d'erreur.
FGS	*Fine Granular Scalability*
GOF	*Group Of Frame*
H.26X	H.261, H.262, H.263X, H.26L sont des normes de compression vidéo numérique bas débit issues de l'ITU. Notons que H.264 est issue du groupe de travail commun ISO et ITU et prolonge la norme H.26L.

IETF	*Internet Engeneering Task Force* Organisme de standardisation des technologies Internet.
IGMP	*Internet Group Management Protocol* Protocole de gestion d'abonnements aux groupes multicast.
ISO	*Organisation Internationale de normalisation*
ITU	*International Telecommunications Union* Institut de normalisation des Télécommunications.
JPEG	*Joint Picture Expert Group* JPEG et JPEG 2000 sont des normes de compression d'images fixes.
LDA	*Loss Delay Adjustment* Protocole de contrôle de congestion point-à-point.
MDS	*Maximum Distance Separable* Type de codes correcteurs.
MLDA	*Multicast enhanced Loss-Delay Adaptation* Protocole de contrôle de congestion multipoint.
MPEG	*Motion Picture Expert Group* MPEG-1, MPEG-2 et MPEG-4 sont des normes de compression numérique. MPEG-7 est une norme de description d'outils audio-visuels.
MSS	*Maximum Segment Size*
MTU	*Maximum Transfert Unit*
NACK	*Negative Acknowledgement* Rapport réseau indiquant la non réception d'un paquet.
NS 2	*Network Simulator version 2* Outil de simulation réseau.
NTSC	*National Television System Committee*
OBMC	*Overlapped Block Motion Compensation*
OBME	*Overlapped Block Motion Estimation*
OTT	*One Trip Time* Délai de transmission entre deux entités.
PGMCC	*Pragmatic General Multicast Congestion Control* Protocole de contrôle de congestion multipoint.
PSNR	*Peak Signal to Noise Ratio*
QoS	*Quality of Service*
QMF	*Quadrature Mirror Filter* Famille de filtres.
RAP	*Rate Adaptation Protocol* Protocole de contrôle de congestion point-à-point.
RFC	*Request For Comments* Notes techniques et organisationnelles au sujet de l'Internet.
RLM	*Receiver-driven Layered Multicast* Protocole de contrôle de congestion multipoint.
RLC	*Receiver-driven Layered Congestion* Protocole de contrôle de congestion multipoint.

RR	*Receiver Report* Rapport de réception RTCP.
RTP/RTCP	*Real-time Transport Protocol/Real-Time Control Protocol* Protocoles de transport applicatifs dédiés aux flux temps-réels.
RTT	*Round Trip Time* Délai aller-retour entre deux entités sur le réseau.
SAD	*Sum of Absolute Differences*
SAMM	*Source Adaptive Multi-layered Multicast* Protocole de contrôle de congestion multipoint.
SDP	*Session Description Protocol*
SNR	*Signal to Noise Ratio*
SPECK	*Set Partition Embeddeded bloCK* Algorithme de compression d'images fixes.
SPIHT	*Set Partitioning In Hierarchical Tree* Algorithme de compression d'images fixes.
SR	*Sender Report* Rapport d'émission RTCP.
TEAR	*TCP Emulation At the Receiver* Protocole de contrôle de congestion point-à-point.
TCP	*Transmission Control Protocol* Protocole de transport fiable.
TFRC	*TCP-Friendly Rate Control* Protocole de contrôle de congestion point-à-point.
TFMCC	*TCP-Friendly Multicast Congestion Control* Protocole de contrôle de congestion multipoint.
UDP	*User Datagram Protocol* Protocole de transport non fiable.
VLC	*Variable Length Code*
VO	*Video Object* Objet vidéo de formes arbitraires dans MPEG-4.
VOL	*Video Object Layer* Couches associées à chaque VO, contenant chacune un VOP.
VOP	*Video Object Plan* Plan relatif à un objet vidéo de formes arbitraires dans MPEG-4.
WEBRC	*Wave and Equation Based rate Control* Protocole de contrôle de congestion multipoint.

Introduction

Contexte de l'étude et problématique Avec l'apparition de nouvelles infrastructures de télécommunications, on observe, à l'heure actuelle, une explosion des applications et services multimédia. On parle alors de visioconférence, de vidéo à la demande (*Streaming*) et de multiples autres applications permises et transmises sur des réseaux filaires (ou non). Ce type d'application, engendrant la transmission de flux continus de données temps-réel, présente des besoins qui diffèrent notablement des communications de données classiques. Ces applications doivent, en effet, pouvoir fournir à leurs utilisateurs un minimum de qualité de service (QoS) comme le délai et la qualité du rendu.

Parce que l'Internet fournit aujourd'hui un service de type "Best-effort" (ou au mieux), à caractéristiques (délais, bande passante, pertes) hétérogènes et variant dans le temps, les services multimédia temps-réel souffrent de dégradations. Ces dégradations sont des conséquences directes des phénomènes néfastes liés à la congestion. En effet, lorsque la demande des utilisateurs excède la capacité du réseau, on observe un phénomène de saturation qui se traduit par une dégradation des performances globales du réseau. Les délais de "bout en bout", ainsi que le taux de perte des paquets, augmentent alors rapidement, dégradant le débit effectif du réseau. Pour éviter ces problèmes, des mécanismes de contrôle de transmission sont nécessaires pour limiter le taux d'utilisation des ressources du réseau en contrôlant le débit des paquets émis par les utilisateurs. L'accroissement de la capacité effective du canal et la minimisation de l'impact négatif de ce service "au mieux" sur la qualité des données multimédia recues par les clients, passent par la réalisation d'un couplage entre la source et les mécanismes de transmission.

Les flux multimédia se trouvent en concurrence, en terme d'utilisation de bande passante, avec les autres trafics de l'Internet. Ainsi, afin de maintenir une certaine équité entre les échanges de données traditionnelles et les transmissions multimédia, il est nécessaire de concevoir des nouvelles stratégies de contrôle de congestion, dédiées à ces flux continus, au moins aussi réactives que le protocole TCP (*Transmission Control Protocol*) [Pos81] (cf. Annexe A), principal protocole de transport utilisé sur l'Internet (i.e. pour les données téléinformatiques classiques). Il faut noter que le protocole TCP du fait de ces mécanismes d'acquittement et de demande de retransmission ne permet pas de respecter les contraintes de délai inhérentes aux applications multimédia temps-réel de type streaming, visioconférence ou télémédecine.

De nouveaux modèles représentation et de codage dits "scalables" (rapport signal

à bruit, spatial, temporelle) des signaux vidéo, sont apparus dans les normes standards de codage vidéo (i.e. H.263X, MPEG-4). Ces modèles sont particulièrement bien adaptés pour répondre aux problèmes d'adaptation de débit de la source vidéo aux caractéristiques du réseau dans des scénarios de transmission multipoint hétérogène ou lorsque le codage des signaux n'est pas réalisé en temps réel (applications de consultation multimédia par exemple). Le principe d'une réprésentation scalable (ou multiniveaux), consiste à générer, pour une source vidéo donnée, plusieurs flux compressés correspondant chacun à un incrément de débit et de qualité. Cependant, la granularité d'adaptation de débit permise par les réprésentations classiques reste assez grossière.

Les travaux réalisés dans le cadre de cette thèse se placent dans un contexte d'optimisation de la qualité de service de bout-en-bout d'une chaîne de communication vidéo. Leur objectif concerne l'étude de nouveaux modèles de représentation scalable à grain fin de signaux vidéo et de techniques de régulation de débit (contrôle de congestion) associées pour la transmission sur des réseaux de paquets hétérogènes, aux caractéristiques variant dans le temps, tels que l'Internet.

Contributions de le thèse. Les contributions de cette thèse se situent dans la problématique du couplage source-réseau. Dans ce cadre, nos contributions sont :

- Dans un premier temps, nous proposons un **nouvel algorithme de régulation de débit point-à-point** couplant un **protocole de congestion TCP-compatible basé sur le protocole RTP/RTCP** (*Real-time Transport Protocol/Real-time Transport Control Protocol*) [SCFJ96] (cf. Annexe B) avec un **modèle de régulation global intégrant les modèles de délais et de *buffers*** de la source, dans le but de minimiser la distorsion du signal décodé au récepteur. Le modèle global proposé permet de réduire de manière significative les pertes dûes aux retards et donc de minimiser la distorsion tout en maximisant l'utilisation de la bande passante TCP-compatible. A l'inverse des approches proposées dans la littérature le protocole développé est dédié à la transmission multimédia et prend en considération les différentes contraintes inhérentes à ce type de flux.
- L'extension de l'approche précédente proposée pour le mode point-à-point au cas multipoint n'est pas envisageable par nature. Nous avons donc développé un **nouvel algorithme de contrôle de débit TCP-compatible hybride orienté émetteur-récepteur** prenant en compte les **caractéristiques débit-distorsion de la source** pour la **transmission multicast de vidéo en couches**. La stratégie que nous proposons s'appuie sur un **mécanisme d'agrégation des rapports des récepteurs dans les noeuds du réseau afin d'éviter l'implosion de la voie de retour**. Une fois l'information collectée, un algorithme de décision déterminant le nombre de couches à émettre, leurs débits et leur éventuel niveau de protection. La décision cherche à maximiser la qualité globale perçue par l'ensemble des clients de la session multicast, est mis en oeuvre. Un protocole complet, gérant les décisions d'abonnements/désabonnements ainsi que la signalisation associée, est proposé. Ce nouvel algorithme est, de plus, couplé

avec un système de transmission vidéo scalable à grain fin (FGS: *Fine Granular Scalability*) multicouche.

- Après nous être intéressés à l'aspect canal, nous proposons l'**architecture complète d'un nouvel algorithme de compression vidéo bas débit finement scalable** permettant une régulation fine du débit de la source. Ce type d'algorithme permet de pallier les limitations des codeurs scalables standards en terme de granularité d'adaptation. Le schéma de codage vidéo nouvelle génération proposé se base sur l'utilisation d'une **décomposition en ondelettes dans les dimensions spatiale et temporelle** (2D+t). Dans ce cadre, nous proposons également une **étude sur la problématique de l'analyse spatio-temporelle compensée en mouvement**. Différents schémas de filtrage basés sur des filtres courts ou longs et utilisant divers modèles de mouvement sont utilisés dans cette étude. L'étude menée a pour but de souligner les limites des approches classiques en ouvrant d'éventuelles voies de réflexion et de recherche. Les schémas de codage hautement scalables que nous proposons se placent, à l'inverse de très nombreuses approches de la littérature, dans une optique bas débit.

Organisation du document. Ce document est organisé en deux parties. La première partie traite des mécanismes de régulation réseau et du couplage avec des applications de transmission vidéo. Dans ce cadre, le **chapitre 1** décrit différentes solutions de contrôle de flux et de congestion proposées dans la littérature dans le cadre de transmission en mode point-à-point ou multipoint. Ce chapitre s'intéresse également à la gestion des *buffers* qui jouent un rôle très important notamment afin d'assurer la qualité de rendu des flux multimédia transmis. Le **chapitre 2** concerne les transmissions unicast et décrit, en détails, le nouveau protocole de contrôle de contrôle de congestion TCP-compatible adaptée à la transmission multimédia que nous proposons. Le **chapitre 3** décrit, quant à lui, le nouveau protocole TCP-compatible de transmission vidéo multipoint en couches que nous proposons.

La seconde partie concerne les algorithmes de codage vidéo scalable nouvelle génération. Dans ce cadre, le **chapitre 4** donne un état de l'art des différents algorithmes de codage scalables portant sur les images fixes ou la vidéo. Ce chapitre est spécialement orienté vers l'obtention d'une scalabilité à grain fin par l'utilisation de décompositions en ondelettes dans les dimensions spatiale et temporelle. Le **chapitre 5** propose une étude portant sur la problématique de l'analyse temporelle et décrit les algorithmes de compression vidéo, bas débit, finement scalables que nous avons développés.

I Contrôle de flux et de congestion pour la transmission multimédia

Chapitre 1

Contrôle de flux et de congestion : état de l'art

1.1 Introduction

Pour toute communication sur l'Internet se posera le problème de l'adaptation du débit des données issues de la source aux contraintes de charges du réseau et à aux capacités de traitement des récepteurs. Le **contrôle de flux** vise à mettre en correspondance le débit des informations transmises à un récepteur et les capacités de réception de ce dernier. Lorsque l'agrégation des trafics, transitant dans un point du réseau, excède les capacités de traitement des routeurs, des phénomènes de congestion se produisent. Ceci aboutit à la saturation des files d'attente de ces équipements puis à la destruction, faute de place, de paquets de données. Le **contrôle de congestion** a pour but de réguler la source afin d'éviter ces phénomènes de congestion. Le principe sur lequel repose ces mécanismes de contrôle est un processus adaptatif de rétroaction entre une source de données et un récepteur, qui permet de réguler le débit d'émission en fonction de l'évolution de certains paramètres de la communication (délais, pertes, etc).

Le protocole TCP (*Transmission Control Protocol*)[Ste94, WS95] (cf annexe A), principal protocole de transport utilisé sur l'Internet, intègre des mécanismes de contrôle de flux et de congestion. Ces mécanismes performants sont à la base même du bon fonctionnement de l'Internet. Cependant, la transmission de flux continus de données multimédia dans l'Internet présente des contraintes qui diffèrent des transmissions de données classiques utilisant TCP. En effet, bien que les médias sonores et visuels, du fait des capacités de leur destinataire (humain), tolérent généralement une certaine dégradation, induite par des pertes de paquets dans le réseau, ceux-ci s'accomodent difficilement des délais de transmission introduits par les mécanismes de contrôle d'un protocole fiable, tel que TCP. De plus, ces approches conduisent à des variations de débits brutales, ce qui s'avère être en contradiction avec les besoins d'un flux multimédia qui demande une Qualité de Service (QoS) assez stable. C'est pourquoi, le protocole TCP a été délaissé au profit de protocoles non fiables comme UDP (*User Da-*

tagram Protocol) et par extension RTP/RTCP (*Real-time Transport Protocol/ Real-time Transport Control Protocol* [SCFJ96], démunis de contrôle de congestion. Ces derniers peuvent donc ne pas répondre aux signes de congestion, alors que TCP y répondra en réduisant le débit de ses applications. En laissant TCP supporter seul l'évitement de congestion, on obtient inévitablement un partage inéquitable des ressources réseaux. Pour éviter cela, il est impératif que chaque application mette en oeuvre son propre mécanisme de contrôle de congestion permettant un partage équitable et optimal des ressources utilisables du réseau afin de garantir une coexistence harmonieuse entre les sessions multimédia et les échanges de données traditionnelles. Un protocole se comportant équitablement avec TCP sera dit **TCP-compatible**.

Dans ce chapitre, nous décrivons différentes solutions de contrôle de flux et de congestion proposées dans la littérature afin de réguler des flux continus tels que les flux multimédia. Nous nous intéresserons également dans ce cadre à la gestion des *buffers* jouant un rôle très important notamment afin d'assurer la continuité de rendu des flux multimédia au récepteur. Cette étude est menée à la fois pour le cas point-à-point et le cas multipoint.

1.2 Transmission point-à-point versus multipoint

Nous montrons ici dans quelle mesure les problématiques liées au contrôle de flux et de congestion diffèrent dans le cadre de communications point-à-point et multipoint.

1.2.1 Transmission point à point

Dans le cas de transmission point-à-point ou **unicast**, les mécanismes de contrôle de congestion mis en oeuvre sont dits d'adaptation à la source. On considère dans ce cas, l'émetteur comme étant, au moins en partie, responsable de la congestion observée. Basé sur un processus de rétroaction entre les deux acteurs de la session, la source est régulée en fonction de l'évolution de certains paramètres de la communication (délais,pertes, etc).

1.2.2 Transmission multipoint

Les communications multipoints ou **multicast** obéissent à un principe d'économie: lorsque plusieurs entités destinataires souhaitent recevoir une même information, il est possible d'établir simultanément autant de communications point à point qu'il y a de récepteurs, et de transmettre à chacun la même information. Mais il est évident que, dès lors que plusieurs connexions empruntent une même portion du réseau, les ressources de communication existant sur cette portion vont devoir manipuler inutilement plusieurs fois la même information. La communication multipoint vise alors à ne faire transiter l'information qu'une seule fois sur la portion de réseau commune à plusieurs connexions (voir figure 1.1). La transmission d'informations provenant d'une source unique, ou transmission 1 vers N, aboutit à la définition d'un arbre de diffusion multipoint, dont la source est la racine et les récepteurs sont les feuilles.

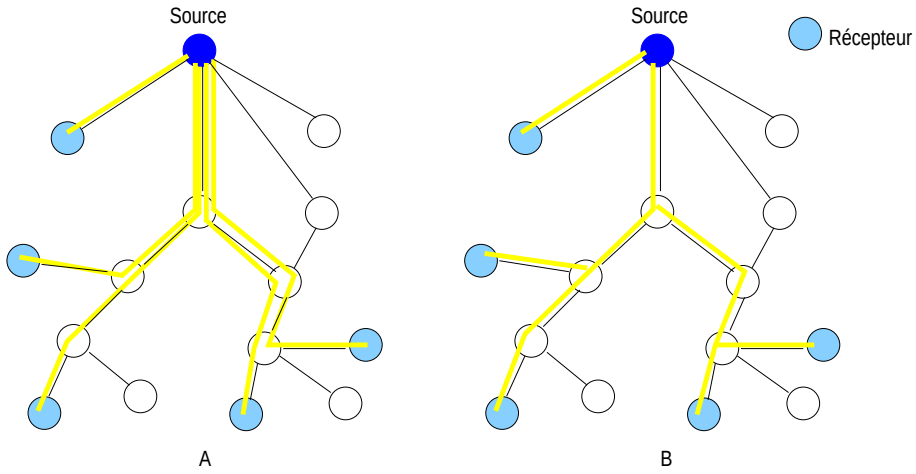

FIG. 1.1 – *Communications point-à-point (A) versus multipoint (B)*

Les communications multipoints posent, donc, des problèmes supplémentaires liés à l'hétérogénéité des conditions de transmission dans le réseau, mais aussi aux capacités de traitements des récepteurs. L'adaptation à la source peut être satisfaisante dans le cas d'une transmission point-à-point, mais elle perd de son intérêt lors d'une transmission multipoint car toute décision prise à la source dans le but de réduire la congestion va affecter la totalité des destinations. Cela signifie que, pour une congestion apparaissant sur un noeud isolé de la session multipoint, la qualité de réception de tous les récepteurs sera affectée par une éventuelle baisse de débit. De plus, il n'est plus possible de permettre à chaque récepteur de transmettre directement des informations à l'émetteur sous peine d'écrouler la voie de retour. Il s'agit alors de concevoir un ensemble de mécanismes permettant à une source de transmettre un même flux (e.g. vidéo) à plusieurs récepteurs, tout en ayant la possibilité d'adapter de manière dynamique et avec une granularité suffisamment fine le débit reçu par chacun aux conditions de transmission qu'il perçoit. De nouveaux schémas de contrôle de flux et de congestion ont été proposés dans le cadre du multipoint. Ils se caractérisent par un transfert, total ou partiel, de la responsabilité du contrôle vers les récepteurs qui prennent des décisions locales afin d'adapter le débit des données véhiculées jusqu'à eux aux contraintes du réseau. Ces méthodes s'appuient sur les fonctions de routage multipoint et sur l'utilisation de sources capables de générer des flux multiniveaux ou scalables (cf section 4). A chaque niveau correspond un incrément de qualité. Chaque sous-flux se voit assigner une adresse multipoint. Ce principe permet une reconfiguration dynamique de l'arbre multipoint par élagages et greffes pour l'acheminement des sous-flux via le protocole IGMP (*Internet Group Multicast Protocol*). Il suffit alors à chaque récepteur de s'abonner aux adresses multipoints qui correspondent aux sous-flux qu'il souhaite ou peut recevoir. En cas de congestion, chaque récepteur peut choisir d'abandonner un sous-flux particulier en signifiant qu'il ne souhaite plus recevoir l'adresse multipoint correspondante. Les techniques d'*élagage* (pruning) vont alors entraîner l'arrêt de l'acheminement de ce

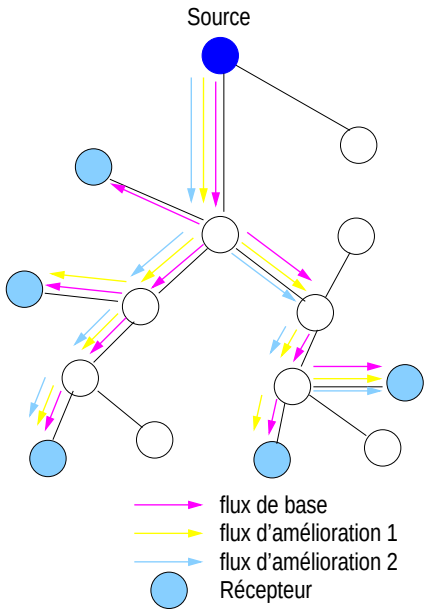

FIG. 1.2 – *Arborescence d'une transmission multicast multicouche d'un flux scalable.*

sous-flux dans la portion du réseau concernée.

Il faut toutefois noter certaines limitations inhérentes à ce type d'approche. La première concerne la granularité d'adaptation du débit qui est fonction du nombre de niveaux de scalabilité fournis par la source. Augmenter le nombre de niveaux de scalabilité pour améliorer la granularité d'adaptation va augmenter la charge induite par le mécanisme de contrôle de débit sans nécessairement résulter en un accroissement significatif de la qualité. De plus, un nombre trop important de couches va accroître la complexité de gestions des adresses multipoints. Ceci va également entraîner une surcharge de trafic IGMP liés à la gestion dynamique de l'arborescence et à un gaspillage de bande passante dû aux entêtes de paquets.

1.3 Contrôle de flux et de congestion : point-à-point

Dans cette section, nous décrivons les principaux protocoles TCP-compatibles proposées dans la littérature dans le cadre de transmission point à point. Parmi les protocoles présentés ici, on peut distinguer trois types d'algorithmes de contrôle de congestion. Les plus proches de TCP sont les approches **basées-fenêtre** qui emploient une fenêtre de congestion à la TCP. Le second type d'algorithme sont les approches qui ajustent leur débit d'émission à la manière de TCP, c'est-à-dire par une augmentation additive du débit et une diminution multiplicative, mais sans utilisation de fenêtre de congestion. On appellera ces approches **AIMD** (*Additive Increase Multiplicative Decrease*). Enfin, la troisième catégorie, qui est sûrement celle la plus éloignée de TCP en terme de comportement instantanné, regroupe ce que l'on appellera les protocoles

basés-modèle ou **basés-équation**. Ces mécanismes utilisent une équation, dérivée d'une modélisation analytique du protocole TCP, afin de déterminer le débit d'émission appropriée.

1.3.1 Protocoles basés-fenêtre

Les mécanismes de contrôle de congestion unicast les plus proches de TCP maintiennent une fenêtre de congestion qui est utilisée directement ou indirectement afin de contrôler le mécanisme d'émission de paquets. Dans [JE96], les auteurs utilisent directement l'algorithme de contrôle de congestion de TCP afin de réguler le débit d'émission de leur flux vidéo. Ainsi l'émetteur contrôle la taille de sa fenêtre à la manière de TCP, mais au lieu de retransmettre les paquets perdus, il permet à la source de continuer à transmettre de nouvelles données. La taille de sa fenêtre de congestion est traduite en une contrainte de débit pour le codec vidéo. Du fait de l'utilisation directe des mécanismes de TCP le comportement de ce protocole n'est pas véritablement utilisable pour la transmission de médias continus.

Dans [Gol98], les auteurs présentent une étude générale portant à la fois sur les mécanismes de contrôles de flux basés-fenêtre et basés-débit (i.e. ne spécifie pas un nombre de paquets à envoyer mais un débit d'émission). Sur les bases de cette étude, un schéma de contrôle de congestion unicast pour la transmission sur réseau IP a été développé. Ce schéma est appelé **MCFC (Minimum Cost Flow Control)**. Pour que l'algorithme exact puisse pleinement être mis en oeuvre il nécessite l'ajout de fonctionnalités supplémentaires dans les routeurs non supportées à l'heure actuelle. Toutefois, une version plus grossière a été également conçue afin de fonctionner sur l'Internet actuel. Le protocole MCFC apporte certaines améliorations par rapport à TCP mais a un comportement général identique.

L'approche utilisée dans **TEAR (TCP Emulation At the Receivers)** [RO00], utilisable à la fois pour les sessions unicast et multicast, maintient également une fenêtre de congestion, et la taille de cette fenêtre est utilisée afin d'en déduire un débit d'émission. La fenêtre de congestion est gérée au récepteur afin de pouvoir fonctionner en unicast et multicast. Les retards et les triples acquittements sont émulés par le récepteur qui les utilise à la manière de TCP pour ajuster sa fenêtre. Le récepteur divise ensuite la taille de la fenêtre de congestion par la valeur estimée du RTT (*Round-Trip Time* : délai d'aller-retour entre l'émetteur et le récepteur) afin d'obtenir une estimation de débit TCP-compatible.

1.3.2 Protocoles AIMD

L'algorithme **LDA (Loss Delay Adjustment**) présenté dans [SS98] émule les mécanismes AIMD de TCP. Ce protocole s'appuie sur l'utilisation du protocole RTP / RTCP et notamment sur l'utilisation des rapports RTCP afin d'estimer les taux de pertes et le RTT. Il est proposé, de plus, la modification de RTP afin de per-

mettre d'estimer le débit du goulot d'étranglement (*bottleneck*) du lien sous-jacent par l'utilisation de la technique de paire de paquets proposée dans [BVG96]. Pendant les périodes où aucune perte de paquet n'intervient, le débit d'émission est incrémenté par une valeur dépendante du ratio du débit d'émission courant par rapport au débit du goulot d'étranglement. Lorsque RTCP indique des pertes pendant le RTT courant, le débit d'émission est réduit exponentiellement en proportion du nombre de paquets perdus. Une amélioration appelée LDA+ est proposée dans [SW00a]. Dans cette dernière, lorsqu'interviennent des pertes le débit est donné par le modèle analytique de TCP [PFTK98] (cf section sur les protocoles basés-modèle).

Dans le protocole **RAP (Rate Adaptation Protocol)** [RHE99] chaque paquet est acquitté par le récepteur. Les paquets d'acquittement (ACK) sont utilisés afin d'estimer le RTT et de détecter les pertes de paquets. Des informations de redondances rajoutées dans les ACK permettent une meilleure résistance aux pertes de ces paquets. Quand le protocole décèle la présence de congestion par l'observation d'une perte de paquet, il divise son débit d'émission par deux. En l'absence de pertes, il augmente son débit d'un paquet par RTT. Ces décisions d'augmentation ou de diminution sont prises une fois par RTT. Afin de fournir une gestion plus fine d'évitement de congestion, le ratio entre une estimation de RTT à court terme et moyenne à long-terme est utilisé afin de modifier les délais entre deux envois de paquets de données.

Un algorithme purement AIMD est caractérisé par deux paramètres a et b, correspondant aux paramètres d'augmentation et de diminution [FHP00] [YL00]. Les auteurs de [BB01] considèrent un nouveau type d'algorithmes de contrôle de congestion dits *binômiaux*, et qui sont une généralisation non linéaire des schémas AIMD couplés avec l'utilisation d'une fenêtre. Ces algorithmes sont caractérisés par quatre paramètres k, l, a et b. En présence de congestion, un algorithme binômial réduit sa fenêtre de W à $W - bW^l$, alors que chaque RTT sans perte conduit une augmentation de la fenêtre de W à $W + a/W^k$. Un algorithme binômial est dit TCP-compatible si et seulement si $k + l = 1$ et $l \leq 1$, pour des valeurs de a et b adéquates. Deux algorithmes TCP-compatibles dits lents/réactifs sont étudiés et évalués.

1.3.3 Protocoles basés-modèle

Le concept de contrôle de congestion utilisant une équation modélisant le comportement de TCP a été proposé pour la première fois dans [MF97]. De nombreux travaux de recherches ont été menés afin de dériver des modèles analytiques des connexions TCP [Flo91, OkM96, MSMO97, OKM, MF97, PFTK98, BH00]. Les premiers travaux ont montré que le débit stationnaire d'une connexion TCP varie dans l'ordre inverse de la racine carrée du taux de pertes observé par la connexion [MSMO97, OKM].

Ainsi un premier modèle, qu'on appelera **modèle simple**, a été proposé dans

[MF97]. La fonction de réponse TCP est donnée par :

$$T = \sqrt{1.5}\frac{MTU}{RTT \times \sqrt{p}} \tag{1.1}$$

Ce dernier donne une borne supérieure T pour le débit TCP en fonction du RTT, du taux de pertes en état stationnaire et du MTU (*Maximum Transfer Unit*). Le MTU d'un lien représente la taille maximale que peut prendre un paquet sur ce lien sans être fragmenté. T représente le débit moyen qu'utiliserait une connexion TCP dans les mêmes conditions (pertes, délai,...). Cette version de l'équation est dérivée à partir d'un simple modèle, étudié dans [Flo91] puis [OkM96], dans lequel une connexion TCP considère des pertes de paquets déterministes.

Une utilisation directe de ce modèle appliquée à la transmission vidéo unicast a été proposée dans [TZ99]. Le récepteur mesure le RTT et le taux de pertes sur un nombre fixé de RTT. L'émetteur réinjecte ensuite ces informations dans l'équation (1.1) afin de déterminer le débit d'émission et d'encodage de leur codeur MPEG associé. Il faut noter que les auteurs, afin de lisser le débit, ont décidé de considérer plusieurs pertes intervenant pendant un RTT comme un unique événement de congestion, influant ainsi sur la manière de calculer le paramètre p. Le but de ces travaux n'était pas le mécanisme de contrôle de congestion en lui même, mais le couplage entre contrôle de congestion et compression vidéo scalable et robuste.

Observant qu'une utilisation directe de ces modèles menait à des variations de débit en "dents de scie", inacceptables pour les flux multimédia un nouveau protocole a été proposé dans [TZ98]. Ce dernier ne permettant pas une réaction rapide par rapport à TCP, [LTG99] propose un nouveau protocole. Ce dernier se fonde sur un mécanisme de contrôle de débit combinant une prédiction de débit TCP-compatible utilisant l'équation (1.1) et une boucle de contrôle basée sur le RTT permettant une réaction rapide. Il est également utilisable lorsque le taux de pertes est nul, ce qui n'est pas le cas du modèle direct. Afin de lisser encore plus le comportement, le taux de pertes est estimé sur une large fenêtre de temps. Une moyenne exponentielle pondérée est également utilisée pour lisser les valeurs de RTT. Cet algorithme de régulation hybride permet une régulation de débit relativement "lissée"tout en étant très réactif. Un couplage avec un codeur vidéo MPEG-4 robuste est d'ailleurs décrit dans [LTG99] .

Dans [MSMO97] et [PFTK98] les auteurs ont montré que, dans la plupart des connexions TCP réelles, un pourcentage important d'événements entrainant une réduction de fenêtre est dû aux mécanismes de gestion des retards. Ainsi, le modèle simple n'est plus valide pour des taux de pertes supérieurs à 5%, puisque pour des taux de pertes supérieurs TCP, est affecté par le mécanisme de retransmission sur retards. C'est pourquoi un nouveau modèle prenant en compte l'impact des retards est proposé dans [PFTK98]. La formulation de l'équation de ce modèle TCP , que l'on appellera **modèle**

complexe, est donnée par :

$$B = \frac{s}{RTT\sqrt{\frac{2bp}{3}} + T_o min(1, 3\sqrt{\frac{3bp}{8}})p(1+32p^2)}, \tag{1.2}$$

avec RTT le temps d'aller-retour entre l'émetteur et le récepteur, et p le taux de pertes, b le nombre de paquets acquittés par un ACK TCP et T_o le délai avant retransmission de l'algorithme TCP.

Basés sur ce nouveau modèle, les auteurs de [PKTK99b], proposent un nouveau protocole TCP-compatible pour les transmissions de flux continus en unicast dans lequel le récepteur acquitte chaque paquet. A intervalles de temps régulier, l'émetteur calcule le taux de pertes observés durant l'intervalle précédent. Lorsqu'il n'observe aucune perte, le débit d'émission est multiplié par deux durant le prochain intervalle. Lorsque des pertes sont observées le débit d'émission est calculé avec l'équation (1.2). Comme montré dans [FHPW00], le fait que ce protocole calcule ses paramètres et ajuste son débit à intervalles fixes, déterminés a priori, le rend vulnérable aux changements de RTT et de débit d'émission.

Malgré les efforts de recherches déployés dans ce domaine, la majorité des approches proposées dans la littérature et présentées ici, exhibent des propriétés de variations de débit encore très instables. C'est pourquoi, adoptant le modèle analytique complexe de TCP proposé dans [PFTK98], un nouveau protocole de contrôle de congestion appelé **TFRC (TCP-Friendly Rate Control protocol)** est introduit dans [FHPW00, FHP01]. Plusieurs points novateurs ont été introduits dans ce protocole afin de le rendre plus stable tout en restant réactif. Le principal point est l'utilisation du taux d'événements de congestion au lieu du taux de pertes pour le paramètre p. L'idée est que TCP ne diminue pas sa fenêtre par deux à chaque paquet perdu. Ainsi, si plusieurs pertes interviennent pendant un RTT un seul *événement de congestion* sera comptabilisé. Le nombre de paquets reçus entre deux événements de congestion est appelé un *intervalle de pertes*. Afin d'estimer le taux d'événements de pertes, on calcule ensuite la taille moyenne d'un intervalle de pertes par une moyenne pondérée des n derniers intervalles de pertes par

$$s_{avg} = \frac{\sum_{i=1}^{n} w_i s_i}{\sum_{i=1}^{n} w_i} \tag{1.3}$$

avec s_i les derniers intervalles de pertes et w_i les pondérations. L'idée des pondérations est d'accorder plus de poids aux derniers intervalles de pertes. On ne calcule plus ici les pertes sur une fenêtre de temps fixe, mais sur un nombre d'intervalle de pertes. le taux d'événements de congestion est ensuite donné par

$$p = \frac{1}{s_{avg}} \tag{1.4}$$

L'utilisation du taux d'événements de congestion permet d'avoir un comportement nettement plus stable et de réagir plus lentement aux congestions tout en restant réactif.

Nous avons décrit ici, l'idée générale du mécanisme de calcul de taux d'événements de congestion; la description complète est donnée dans [FHPW00, FHP01]. Le taux d'événements de congestion est calculé tous les RTT et envoyé à l'émetteur. Le RTT est quand à lui lissé à l'aide d'une moyenne exponentielle pondérée. Le protocole proposé comprend également une phase d'initialisation tendant à émuler le *slow-start* de TCP. Les performances exhibées par ce protocole en terme de variation de débits et de compatibilité avec TCP sont très intéressantes.

Remarques:

- TCP Reno est l'implémentation de TCP la plus couramment utilisée sur l'Internet. C'est pourquoi, la plupart des travaux se sont attachés à modéliser TCP Reno bien qu'il existe plusieurs autres versions (Tahoe, New Reno, Sack, Vegas). Bacelli et al. dans [BH00] propose une étude et un modèle général de TCP, à partir duquel les diverses versions sont données par différentes valeurs de paramètres.
- Deux études comparatives sur les principaux types d'algorithmes de contrôle de congestion et sur l'intérêt de conception de protocoles lissés et réactifs sont proposées dans [FHP00] et [BBFS01].

1.4 Contrôle de flux et de congestion : multipoint

Comme on a pu le voir dans la section 1.2, les transmissions multicast se caractérisent, en règle générale, par une forte hétérogénéité des récepteurs tant au niveau de la bande passante qu'au niveau de leur capacité de traitement. C'est pourquoi, les solutions utilisées en point à point ne sont plus envisageables en multipoint. L'objet de ce paragraphe est de présenter les principales méthodes de régulation de débit et de contrôle de congestion relatives à la transmision multimédia multicast, envisagées dans la littérature. Plusieurs méthodes orientées récepteurs ou hybrides (proposant une collaboration entre l'émetteur et les récepteurs) ont été proposées. Nous distinguerons ici également les approches se souciant de l'équité vis-à-vis de TCP (i.e. TCP-compatibles) des autres.

1.4.1 Approches non TCP-compatibles

Les premières approches de contrôle de débit proposées pour des environnements multipoints ne se sont généralement pas préoccupées de l'équité vis-à-vis de TCP.

L'une des premières approches est **RLM (Receiver-Driven Layered Multicast)** proposée dans [MVJ97]. Elle sous-entend une source finement scalable capable de générer de nombreux niveaux de scalabilité. Ces niveaux de scalabilité sont supposés interdépendants et sont envoyés sur des sessions multipoint différentes. Mis à part son rôle de génératrice de flux, la source n'a aucun rôle actif dans ce protocole. Chaque récepteur souscrit au nombre de sessions que sa capacité lui permet. Une phase de recherche du nombre de niveaux de scalabilité optimal est mise en oeuvre par chaque récepteur. Elle consiste à accroître le nombre de sessions souscrites par des tentatives d'abonnement

(join experiments). Chaque tentative d'abonnement est suivie d'une période d'observation de la congestion. Si une congestion est observée, le récepteur abandonne la session considérée. Pour éviter les phases de congestions transitoires répétées, provoquées par des tentatives d'abonnement avortées, RLM utilise un algorithme d'apprentissage qui permet d'identifier le nombre d'abonnements critiques. L'idée est d'espacer de plus en plus les tentatives d'abonnement pour les sessions ayant déjà provoquées de la congestion. RLM présente l'avantage de ne pas nécessiter une voie de retour. Cependant, l'inconvénient d'une telle méthode provient du nombre élevé de flux vidéo générés par le codeur scalable granulaire. En effet, ceci conduit à un faible débit attribué à chaque couche de scalabilité, et donc à une faible proportion de données utiles au regard des entêtes. L'incrément de qualité visuelle associé à la réception d'une couche donnée est donc très limité, voir négligeable. De plus, comme montré dans [LB00a], cette approche souffre de différents problèmes de comportements : périodes de congestion transitoires, instabilité et pertes périodiques en rafales [Gho97]. D'autre part, ces abonnements/désabonnements répétés génèrent un trafic IGMP lié à la gestion dynamique des arbres multipoint non négligeable. Bolot et al. [BT98] soulignent enfin le temps de convergence non négligeable de l'algorithme.

Palliant les problèmes comportementaux de RLM liés à son mécanisme d'inférence de la bande passante [LB00a], le protocole **PLM** a été proposé récemment. PLM [LB00b] est un protocole de contrôle de congestion pour la transmission multicast en couches basé sur la génération de paires de paquets dans le but de déduire la bande passante disponible. Deux paquets prioritaires sont envoyés l'un derrière l'autre en rafale et, le délai à la réception entre les deux paquets fournit une information quant à la capacité maximale du lien. PLM ne souffre plus alors des problèmes de comportements cités plus haut, mais requiert un réseau intégrant de la différenciation de service.

On trouve également des approches hybrides où la source et les récepteurs jouent chacun un rôle actif dans la régulation de débit. L'approche **DSG (Destination Set Grouping)**, proposée dans [CAL96], n'utilise pas de codeur scalable, mais diffuse en *simulcast* (en parallèle) trois flux contenant la même scène vidéo, compressée à différents niveaux de qualité. L'ensemble des récepteurs est partitionné en groupe de destinations recevant le même flux vidéo. Le débit de chaque flux est régulé via l'utilisation d'un protocole basé sur des rapports spécifiques. En fonction des fluctuations des ressources du réseau, les récepteurs ont la possibilité de quitter un groupe de destination pour en rejoindre un autre. On peut considérer que DSG est une méthode relativement scalable pouvant répondre à un nombre de groupe de récepteurs élevé tout en offrant une bonne finesse de régulation de débit. Toutefois, son inconvénient majeur réside dans le gaspillage très important de bande passante inhérent au simulcast le rendant non réalisable dans le cas de topologies hétérogènes.

Plus récemment est apparue **SAMM (Source Adaptive Multi-layered Multicast)** [VAS00a], méthode elle aussi hybride émetteur-récepteur mais s'appuyant cette

fois sur une représentation scalable des flux vidéo. Les auteurs comptent sur la mise en place de mécanismes de "différenciation de service" permettant de privilégier la transmission des niveaux de scalabilité les plus importants. Dans cette approche, la source s'appuie sur les observations des récepteurs pour ajuster à la fois le nombre et le débit de chaque niveau de scalabilité. Chaque récepteur estime son "*goodput*", représentant sa capacité de réception, et le renvoie vers la source. Le *goodput* est défini comme le débit cumulé de tous les niveaux reçus sans perte. Si aucune perte n'est observée (tous les niveaux sont reçus intégralement), SAMM permet aux récepteurs de transmettre un *goodput* supérieur à celui mesuré, permettant ainsi, en théorie, d'approcher la capacité du lien. Cet algorithme considère la présence d'agent d'agrégation déployés dans les noeuds du réseau chargés de collecter l'information provenant des rapports des récepteurs, de la fusionner et de l'acheminer vers la source. Les opérations d'agrégation sont les suivantes :

- Si le nombre d'informations de *goodput* différentes reçues N_G est inférieur ou égal au nombre de niveaux de scalabilité que peut générer la source L_{max}, l'agrégateur forme deux vecteurs $r = \{r_i, i = 1, \ldots, L_{max}\}$ (transportant les *goodput* demandés) et $c = \{c_i, i = 1, \ldots, L_{max}\}$ (avec c_i le nombre de récepteurs demandant le débit r_i).
- Si $N_G > L_{max}$ l'agrégateur devra faire un choix parmi les débits demandés. Il choisira la combinaison de $l \in [1, \ldots, L_{max}]$ *goodput* pris parmi les N_G reçus qui maximisera la métrique de "*goodput cumulé*" (*Combined goodput*):

$$G = \sum_{i=1}^{l} r_i c_i \tag{1.5}$$

  Ce n'est qu'après avoir déterminé la sélection optimale qu'il formera ses deux vecteurs.

Ces informations agrégées seront envoyées vers la source. La méthode SAMM présente donc l'avantage d'optimiser le nombre de couches et le débit de chacune des couches au sens du *goodput* cumulé. Elle semble donc conduire à un bon compromis entre finesse du contrôle de débit, économie de bande passante affectée aux en-têtes et bonne utilisation de la bande passante disponible sur le réseau. De plus, grâce à son processus d'agrégation, elle évite tous risques d'implosion de la voie de retour. Toutefois l'approche SAMM présuppose l'existence dans les noeuds du réseau d'entités (agrégateurs) possédant une capacité de calcul non négligeable. D'autre part, comme nous le verrons dans le chapitre suivant le mécanisme d'estimation de la bande passante n'est pas réaliste sur l'Internet actuel. On peut enfin se demander si une telle finesse de régulation de débit est justifiée. En effet, en présence de récepteurs ayant des débits très proches, mais différents, l'algorithme peut conduire à l'envoi de niveaux de scalabilité ayant un débit très faible, n'apportant que peu, voire pas, d'incrément de qualité.

1.4.2 Approches TCP-compatibles

Dans les approches de régulation de débit multipoint décrites ci-dessus, les auteurs ne se sont pas préoccupés de l'équité de leur schéma vis-à-vis de TCP. Il paraît pourtant

tout aussi important d'obtenir un partage équitable de la bande passante avec les applications classiques en multipoint qu'en point à point. Certains auteurs se sont donc attachés à obtenir des mécanismes de contrôle de congestion TCP-compatible en multipoint. On distinguera ici, non seulement les approches orientées récepteurs des approches hybrides émetteur-récepteur, mais également les approches monocouche des approches multicouche.

Parmi les approches monocouche (monosession multipoint), on trouve dans [WH01] une adaptation de TFRC [FHPW00] (décrit dans la section point à point) au multipoint appelée **TFMCC (TCP-Friendly Multicast Congestion Control)**. L'approche TFMCC s'appuie, elle aussi sur l'équation (1.2) modélisant le comportement de TCP. A la différence de TFRC toutefois, ce sont les récepteurs qui calculent leur RTT et estiment leur débit. Cette information est périodiquement envoyée vers l'émetteur. Si un récepteur renvoie une information de retour indiquant un débit plus faible que le débit actuel de la source, celle-ci réduira immédiatement son débit au débit indiqué. De manière à éliminer un grand nombre de rapports, seuls les récepteurs estimant un débit plus faible que le débit de la source transmettent une information. Les auteurs proposent le concept de CLR (*Current Limiting Receiver*). Le CLR est le récepteur qui, vu de l'émetteur, a la plus faible capacité de réception. L'émetteur pourra accroître son débit en s'appuyant sur les estimations du CLR. Le CLR pourra changer en cours de session, suivant les évolutions dans l'arbre multipoint. L'un des points clef de TFMCC est de permettre à tous les récepteurs d'estimer leur RTT sans pour autant augmenter le trafic. Supposant la synchronisation des horloges les membres de la session, chaque récepteur déduit la valeur de son RTT par l'observation du temps d'aller OTT (*One-way Trip Time*). Dès que le CLR est désigné, lui seul continue à estimer réellement et régulièrement son RTT. Tous les autres membres se contenteront de remettre à jour une estimation du RTT grâce à l'observation de l'OTT.

Un protocole assez similaire nommé **PGMCC (Pragmatic General Multicast Congestion Control)** a été proposé dans [Riz00]. Ce schéma s'appuie sur PGM [SFC+00], un protocole de robustification de session multipoint. Ce protocole est basé sur l'utilisation d'accusés de réception négatifs(NACK). Une procédure de "suppression" des rapports, similaire à celle de TFMCC, permet de limiter la quantité de NACK. PGM ne spécifie toutefois pas de mécanisme de contrôle de congestion, et considère des sources à débit fixe. PGMCC peut donc être vu comme une évolution de PGM permettant le contrôle de congestion. Comme dans TFMCC, ce protocole sélectionne le récepteur le plus faible (nommé *"acker"*) parmi l'ensemble des récepteurs. Le processus de sélection du *acker* est un des points clef de la méthode. De lui dépendra l'équité du protocole vis-à-vis de TCP. Ce processus est basé sur une version simplifiée de l'équation (1.2) utilisée par chaque récepteur. De plus, chaque paquet de données transportant des informations sur le *acker* courant, à chaque instant tous les récepteurs peuvent comparer leur débit à celui du *acker* et devenir *acker* le cas échéant. Un mécanisme de contrôle de congestion similaire à celui de TCP est ensuite mis en place entre l'émetteur et le *acker*, qui transmettra un ACK pour chaque paquet reçu. Le *acker* contrôle donc

le débit d'émission ainsi que les éventuelles retransmissions. On remarque que le calcul du débit TCP-compatible n'a pas besoin d'être précis puisqu'il ne sert qu'à déterminer le *acker* et non à contrôler le débit de la source. Un calcul grossier du RTT est donc suffisant. Bien que PGMCC se comporte équitablement avec TCP, celui-ci a plus tendance à générer des débits en dents de scie, assez proche de ce que donne TCP.

On peut remarquer que ces deux protocoles (TFMCC et PGMCC) ne pourront convenir qu'à des topologies d'arbres multipoints très particulières n'ayant qu'un seul goulot d'étranglement. Leur utilisation pour des topologies réalistes, avec plusieurs groupes de récepteurs, ne satisfera qu'une partie des récepteurs et défavorisera les groupes de récepteurs ayant des capacités plus élevées. Seul un protocole capable de réguler les débits d'une source multicouche peut espérer satisfaire un ensemble de récepteurs hétérogènes.

Certaines approches se sont attachées à améliorer et adapter le protocole RLM pour parvenir à une régulation de débit multicast TCP-compatible pour la transmission en couches. L'algorithme proposé dans [TFPB98] consiste à remplacer le mécanisme de *join experiments* par une estimation explicite de la bande passante disponible réalisée par chaque récepteur. En fonction de cette estimation les récepteurs s'abonnent ou se désabonnemnt en conséquence. Afin de réduire le coût d'utilisation de la voie de retour, notamment pour l'estimation du RTT, des paquets de requêtes minimalistes contenant seulement le SSRC du récepteur et l'instant d'émission du paquet ont été proposés. En réponse à ces paquets, la source ne renvoie qu'un seul rapport d'émission contenant les informations de délai relatives à chaque SSRC. Toutefois, le coût de la voie de retour induit par une telle méthode reste encore prohibitif et non scalable. De plus, le trafic IGMP régulant les abonnements/désabonnements reste très élevé.

Un autre protocole de contrôle de congestion multicast multicouche orienté récepteur connu est le protocole **RLC (Receiver-driven Layered Congestion)** [VRC98]. Toutefois, bien que la source n'adapte pas le débit de ses différentes couches, celle-ci est active et notamment dans l'estimation de la bande passante. Ce protocole est basé sur deux principes majeurs : la génération de rafales de paquets et l'utilisation de points de synchronisation. Les rafales de paquets ont pour but de permettre aux récepteurs d'estimer leur bande passante ou, en tous cas, de déterminer ou non si leur abonnement à une couche supérieure est possible. En effet, le débit des rafales sur une couche donnée est égale au débit de la couche courante augmenté du débit de la couche directement supérieure. Le comportement TCP-compatible du protocole est principalement dû à la distribution exponentielle des couches qui résultent en une diminution exponentielle de la bande passante en cas de pertes (comme TCP). Le deuxième mécanisme consiste à spécifier, à l'aide d'une information particulière (*flag*) contenue dans un paquet de données aux récepteurs, quand tenter de s'abonner ou non. L'idée ici est de réduire la latence induite par des tentatives d'abonnements et de désabonnements non synchronisées, notamment pour des récepteurs appartenant à des régions locales observant les mêmes caractéristiques de réception. Comme montré dans [LB00a], et similairement à RLM, cette approche souffre de différents problèmes de comportements : périodes de

congestion transitoires, instabilité et pertes périodiques en rafale. Ces problèmes étant principalement dûs au mécanisme d'inférence de la bande passante. De plus, l'efficacité de ce mécanisme d'inférence semble être fortement couplée avec la taille des files d'attente des routeurs traversés.

MLDA (Multicast Enhanced Loss-Delay Adaptation Algorithm) proposé dans [SW00b] permet la gestion de sources multicouche. Dans cette approche, l'émetteur envoie régulièrement à tous les récepteurs des rapports (SR : *sender report*) contenant des informations sur les débits transmis. Dès réception d'un SR, chaque récepteur estime son taux de pertes et son RTT. Il peut alors estimer son débit TCP-compatible en utilisant l'équation (1.2). Suivant la valeur estimée, le récepteur décide de s'abonner ou de se désabonner aux différentes sessions disponibles. Au bout d'une période aléatoire T_{wait}, il pourra envoyer en multipoint cette information de débit dans un rapport (RR : *receiver report*). Toutefois, si pendant la période T_{wait} il reçoit de la part d'un autre récepteur, une demande pour un débit similaire, il annule l'envoi de son propre rapport. Le nombre et le débit de chaque couche sont déterminés dynamiquement en fonction des informations de retour générées par les récepteurs. Comme dans SAMM [VAS00a], la source devra faire un choix parmi les demandes des récepteurs si le nombre de demandes est supérieur au nombre de sessions multipoint pouvant être ouvertes par la source. Toutefois MLDA ne spécifie pas la méthode qui permettrait de faire ce choix. Tous les débits estimés et générés se situent dans un intervalle de débit $[R_{min}, R_{max}]$ qui est ensuite découpé en sous intervalles. Chaque récepteur n'indique donc pas dans ses RR une valeur de débit mais plutôt un index sur un sous intervalle. Cet algorithme appelé "*Partial Suppression*" est un élément essentiel permettant d'éviter l'explosion de la voie de retour. Comme pour TFMCC, l'un des points délicats de la méthode réside dans le calcul du RTT. Une estimation du RTT basée sur l'observation de l'OTT est réalisée dès réception du SR. Le protocole autorise aussi chaque récepteur à calculer réellement son RTT par envoie d'un RR vers la source. Cette procédure permettant d'éviter les accumulations d'erreurs sur l'estimation du RTT basée OTT, est toutefois réalisée plus rarement (les tests présentés dans [SW00b] montrent que les SR sont envoyés toutes les 5s, alors que le calcul effectif du RTT n'est réalisé que toutes les 60s). On peut regretter que les auteurs ne proposent pas d'algorithme permettant à la source de sélectionner le cas échéant et de manière optimale des débits parmi les demandes des récepteurs. De plus, les délais assez élevés entre deux adaptations de débit ne permettent pas une réactivité forte du mécanisme global engendrant par conséquent une équité vis-à-vis de TCP sur le long terme uniquement, et peuvent expliquer l'instabilité relative des débits générés par la source. Enfin, un tel mécanisme de suppression partielle des rapports peut engendrer une homogénéité non réelle des débits requis.

FLID-DL Fair Layered Increase/Decrease with Dynamic Layering [BFH+00] est un algorithme de contrôle de congestion multicast multicouche. Ce dernier permet de réduire l'impact négatif des délais de latence IGMP avant désabonnement et élimine la nécessité d'intervalle de sondage utilisée dans RLC. Toutefois, la quantité de trafic IGMP et PIM-SM générée par chaque récepteur est très conséquent. De plus, comme

montré dans [LGSH02] un déploiement réel de ce protocole n'est pas réalisable. Le protocole **WEBRC Wave and Equation Based Rate Control** [LG02, LGSH02] est un nouveau protocole basé-modèle qui a été récemment proposé. Il permet de résoudre les principales limites de FLID-DL en utilisant une méthode innovante de transmission en vagues. La couche de base est transmise à un débit quasi-constant alors que les différentes autres couches sont envoyées par vagues en décalage de phase. Le débit de transmission est périodiquement constitué d'une décroissance exponentielle durant les périodes actives suivie d'une période de silence. En fonction de son estimation de bande passante faite en utilisant une version simplifiée de l'équation (1.2), les récepteurs changent de vagues. Il est d'ailleurs proposé la notion de RTT multicast définie comme étant le délai entre l'instant où une demande d'abonnement à une vague est réalisée et l'instant où le premier paquet de cette vague est reçu. Ce protocole permet de réduire de manière très importante le nombre d'abonnements/désabonnements et a un comportement équitable avec TCP. Cependant, WEBRC ainsi que FLID-DL sont avant tout conçus pour des applications de téléchargements fiables voire pour des applications de *streaming*. Il faut noter toutefois que la granularité de la source doit être assez fine et que, par conséquent, la transmission de flux hiérarchiques classiques de type H.263+ ou MPEG-4 n'est pas réalisable.

1.5 Transmission multimédia : délais et buffers

Une des principales différences entre les transmissions multimédia temps-réel ou non (*streaming*) et les applications télé-informatiques classiques, réside dans les contraintes de délais inhérentes à ces médias. Il est ainsi indispensable qu'il y ait une continuité dans la présentation de la vidéo ou de l'audio au récepteur. Idéalement, si les images vidéo sont capturées à une fréquence de 25 Hz, elles doivent être rendues au récepteurs avec une fréquence de 25 Hz.

Sur des réseaux de type Internet, les délais de transmission de bout-en-bout varient beaucoup. On appelle ce phénomène la gigue du réseau (*jitter*). Afin de réduire l'effet de cette gigue sur la qualité du rendu, on utilise classiquement des mécanismes de mémoire tampon (*buffer*) du côté récepteur. Lorsqu'un paquet de données arrive, il n'est pas joué tout de suite, mais il patiente dans le *buffer* jusqu'à son instant de présentation. Naturellement, lorsque le *buffer* est trop petit celui-ci peut se vider plus vite qu'il ne se remplit, du fait des paquets arrivant en retard. Ces sous-remplissages causent un problème dans la visualisation de la vidéo (i.e effet d'image gelée). Les paquets arrivant en retard sont généralement considérés comme perdus. A l'inverse, l'utilisation d'un *buffer* très grand induit des délais de bout-en-bout important non désirables pour des applications interactives comme la visiophonie ou la téléphonie (sur IP).

Nous décrivons brièvement ici les rares travaux (publiés) portant sur la gestion de *buffer* dans le cadre de transmissions multimédia interactives ou non sur l'Internet.

Plusieurs méthodes basées sur des mécanismes d'adaptation de taille de *buffer* dynamique ont été proposées dans la littérature [RKTS94, ACS98, LS99, SFG01]. L'idée

est de minimiser le nombre de retard en limitant le plus possible le nombre de fois où le *buffer* de réception est vide au moment de la consommation. Pour cela on permettra au récepteur une accélération ou un ralentissement de la vitesse de visionnage ou d'écoute en fonction de l'état de remplissage du *buffer* et de l'état du réseau. Ainsi, lorsque le *buffer* est trop rempli, l'audio ou la vidéo sont joués plus rapidement et à l'inverse, lorsque il est sous-rempli la vitesse de rendu est ralentie. Afin d'obtenir un ajustement dynamique du délai de mise en *buffer* du côté récepteur, l'approche décrite dans [RKTS94] se base sur une prédiction du délai de transmission en utilisant un modèle autorégressif. Dans [ACS98], une représentation statistique du délai, établie sur des quantités importantes d'observations réseaux en mode de fonctionnement normal, sert de base à la prédiction de délai court-terme. Dans [LS99], l'ajustement de délai se base sur un prédicteur au moindre carré médian normalisé couplé avec une mesure de variation de délai. Dans [SFG01] l'approche est conçue pour le *streaming* vidéo induisant une faible latence. Les auteurs modélisent ici le délai de sous-remplissage de *buffer* à partir d'une chaine de Markov à deux états (Elliot-Gilbert) modélisant le processus de pertes du réseau. Toutefois, ce dernier fait plusieurs hypothèses non réalistes en vidéo comme le fait que chacune des images est mise dans un paquet. De plus, les temps de mise en *buffer* proposés sont encore très importants dans le cas d'applications interactives. Enfin, toutes les approches décrites ici font l'hypothèse d'une source à débit constant non régulée à des fins d'évitement de congestion.

L'approche proposée dans [REH99] fait l'hypothèse d'une source contrôlée en débit par un protocole de contrôle de congestion basé AIMD. En pratique l'algorithme RAP présenté plus haut est utilisé. L'idée est de se servir de la forme caractéristique en dents de scie des variations de débit du protocole afin de déduire une politique de mise en *buffer* optimale permettant d'assurer la meilleure qualité au récepteur. Ainsi lorsque la bande passante augmente, on en profite pour transmettre des données en avance afin de pallier la prochaine diminution de bande passante. Cette approche fait l'hypothèse d'une source pré-encodée, elle même composée de plusieurs (i.e. au moins 5) couches à débit constant. Il faut noter que l'algorithme mis en oeuvre ne fonctionne que pour une régulation AIMD.

Une autre approche fonctionnant à la fois sur une source à débit variable, et compatible avec une régulation de débit réseau TCP-compatible quelconque (i.e. AIMD et autres) a été proposée dans [SR00]. Dans ce cas, la source est à débit variable pré-encodée et composée, de manière plus réaliste, de 2 couches uniquement. L'idée est assez proche de la technique précédente et consiste en une politique de pré-chargement (*pre-fetching*). Ainsi lorsque la bande passante disponible devient supérieure au débit nécessaire à la vidéo courante, on décide de prendre de l'avance et de commencer à transmettre les images suivantes. Ce processus est évidemment couplé avec l'utilisation d'un délai de pré-chargement initial des données dans le *buffer* de réception de quelques secondes (i.e. au moins 4 secondes). De plus, afin d'estimer l'état de remplissage du *buffer* de réception, l'émetteur se base sur des rapports émis par le récepteur. Toutefois, plusieurs hypothèses fortes et peu réalistes sont faites ici. On considère ainsi que le

buffer de réception a une capacité infinie. On fait l'hypothèse également que le délai de transmission est nul. On considère, de plus, que le serveur n'envoie pas les images qui peuvent arriver en retard, sans donner d'algorithme précis notamment concernant la gestion des prédictions inter-images inhérentes au codage hybride classique. On estime, enfin, que toute image émise sera reçue et décodée, sans prendre en compte les retards dûs à la gigue et les pertes réseaux.

Remarques :

- Les deux dernières approches proposent des solutions hybrides orientées émetteur-récepteur alors que les premières approches n'interviennent qu'au niveau du récepteur.
- La plupart des approches décrites ici font l'hypothèse de temps de pré-chargement très élevés non compatibles avec des applications interactives.
- Même si le ralentissement et l'accélération sont des solutions élégantes au problème d'évitement des retards, celles-ci ont des limites et, encore une fois, ne sont pas toujours compatibles avec des applications très interactives.

1.6 Conclusion

Dans ce chapitre, nous avons décrit les principales approches de contrôle de flux et de congestion proposées dans la littérature pour les transmissions vidéo en mode point à point et multipoint. Parmi les approches point à point, le protocole de contrôle de congestion TFRC est très intéressant. Toutefois, celui-ci, comme tous les autres protocoles de la littérature, considère l'émission de flux continus mais ne prend pas en compte les caractéristiques propres aux flux multimédia émis. De plus, aucune approche pertinente de régulation d'une source vidéo couplée avec un mécanisme de contrôle de congestion TCP-compatible n'a été proposée dans la littérature. Plusieurs approches multicast sont également intéressantes. Cependant, aucune d'elles ne considère les performances débit-distorsion de la source et ne présente un schéma de transmission vidéo multicouche multipoint TCP-compatible en tenant compte. Dans le chapitre 2, nous proposerons donc un nouveau protocole de contrôle de débit et de congestion, inspiré de TFRC, mais prenant en compte les caractéristiques des flux multimédia transportés (délais, tailles de paquets, granularité d'adaptation, modèle débit-distorsion de la source,...). Un modèle global de régulation de la source vidéo est également proposé conduisant à une limitation du nombre de retards de paquets. Puis dans le chapitre 3, nous décrirons un nouveau protocole de contrôle de congestion pour la transmission de vidéo multicouche en multicast. Ce mécanisme de régulation orienté émetteur et récepteur, cherche à maximiser la qualité de la vidéo perçue par l'ensemble des récepteurs.

Chapitre 2

Un nouveau protocole TCP-compatible pour la transmission vidéo point à point

2.1 Introduction

Les besoins en terme de Qualité de Service (QoS) des flux multimédia diffèrent totalement de ceux des communications informatiques classiques. Afin de maintenir une certaine équité entre les échanges de données traditionnelles et les transmissions multimédia, il est nécessaire de concevoir des nouvelles stratégies de contrôle de congestion dédiées à ces flux. Ces dernières doivent pouvoir répondre aux besoins des flux multimédia concernant la stabilité des contraintes de débit, tout en étant presque aussi réactives que TCP (principal protocole utilisé sur l'Internet). Nous avons décrit dans le chapitre précédent différents protocoles et systèmes de transmission proposés dans la littérature.

Cependant, tous ces protocoles ne prennent pas en compte les caractéristiques des flux multimédia qu'ils prétendent contrôler. De ce fait, dans le cas de flux vidéo ou audio, ces mauvaises considérations peuvent mener à un partage inéquitable de la bande passante vis-à-vis des flux TCP. Elles considèrent, de plus, que l'émetteur peut ajuster son débit d'émission exactement au débit TCP-compatible estimé, que la source vidéo s'adapte avec une granularité arbitraire et n'est en aucun cas contraint par des valeurs de bande passante minimum ou maximum par exemple. Enfin, dissocier totalement la partie réseau de la partie source, par l'injection directe dans l'encodeur de la bande passante prédite comme une contrainte de débit, mène à des comportements non souhaitables pour l'application. Nous montrerons ici que ce type d'approche peut souffrir de nombreux effets dûs aux retards de paquets induits par les contraintes temps-réel de la source.

Ce chapitre traite de ces restrictions en proposant un nouvel algorithme de régulation de débit couplant un protocole de congestion TCP-compatible avec un modèle de régulation global intégrant les modèles de délais et de *buffers* de la source, dans le but

de minimiser la distorsion du signal décodé au récepteur. Le modèle global proposé permet de réduire de manière significative les pertes dûes aux retards et donc de minimiser la distorsion tout en maximisant l'utilisation de la bande passante TCP-compatible.

2.2 Un modèle générique pour les communications multimédia

La conception d'un protocole de communication adapté aux communications multimédia requiert une phase de modèlisation fournissant ainsi une liste de contraintes et besoins inhérents à ce type d'application. Nous donnons ici un modèle générique décrivant diverses contraintes imposées par les utilisateurs, les algorithmes de compressions et les données transmises. Le modèle liste six points clés :

Gammes de débit : En règle générale, la qualité des flux multimédia s'améliore avec l'augmentation de débit. Cependant, au dessus d'une certaine limite de débit R_{max}, l'augmentation de qualité est négligeable. De la même façon, réduire la bande passante en dessous d'une certaine limite R_{min} rend les données inutilisables par les récepteurs. Ces valeurs R_{min} et R_{max} sont avant tout dépendantes du modèle débit-distorsion de la source utilisée.

Contraintes de transmission : La transmission de flux multimédia est régie par différentes contraintes de délais et de robustesse. Ainsi pour des raisons de délais les flux audio seront transmis dans des paquets de petites tailles et de tailles variables. De la même façon, il est nécessaire de mettre en oeuvre des politiques intelligentes de paquetisation lors de communication vidéo afin de rendre celle-ci plus robuste aux pertes de paquets. Ces règles de paquetisation conduisent à l'utilisation de paquets de tailles variables dépendantes à la fois du débit de la source mais également de la nature de la vidéo transmise.

Granularité d'adaptation : Suivant le schéma de compression et le type de données utilisés, l'adaptation du débit de la source aux contraintes de réseau ne peut se faire qu'avec une certaine granularité. Ainsi, pour une source vidéo classique la granularité est fonction du pas de quantification ou de la stratégie de réduction de fréquence temporelle (i.e. nombre d'images par seconde). On parle également de granularité d'adaptation temporelle. En effet, l'adaptation possible des paramètres d'encodage a lieu de manière générale en frontière d'image. En effet, elle ne peut intervenir un n'importe quel instant du processus de compression (e.g. en plein milieu d'une image).

Contraintes de délais : Les flux multimédia ont en règle générale des contraintes de délais que l'on peut qualifier de temps-réel. Pour des applications interactives telles que la visiophonie, il est impératif que le délai entre la capture de l'image et l'instant où le récepteur visionne l'image soit le plus court possible. Il est impératif également, et ceci est aussi vrai pour les applications de streaming, d'assurer une continuité dans la présentation des médias au récepteur. Idéalement, si les

images vidéo sont capturées à une fréquence de 25 Hz, elles doivent être rendues au récepteurs avec une fréquence de 25 Hz. Les données doivent donc arriver à temps du côté du récepteurs afin d'éviter des problèmes de qualité dans le rendu. Les données arrivant en retards sont considérées comme perdues.

Qualité de service stable : L'adaptation fréquente du débit de la source à des fins de réactivité face aux phénomènes de congestion peut mener à de fortes variations de débit pour la source. Ces fortes variations ne sont évidemment pas compatibles avec les besoins de QoS des applications multimédia. Lors d'une transmission vidéo, par exemple, les clients sont très sensibles aux brusques différences de qualité. Afin, de permettre aux récepteurs d'avoir une qualité plus stable il est nécessaire de fixer des seuils de variations de qualité acceptables (en débit ou en PSNR) et de s'y conformer. De la même manière, il convient de fixer des bornes concernant les changements de fréquence temporelle acceptables.

Tolérance aux pertes : De manière générale, on cherchera à limiter les pertes de paquets au maximum et on pourra utiliser des mécanismes de protection et de redondance afin de pallier les pertes résiduelles. Toutefois la notion de tolérance aux pertes est très dépendante de l'application et des utilisateurs de l'application. Ainsi, le streaming d'un morceau de musique sera très peu tolérant alors qu'une tolérance supérieure est acceptée pour des applications de visiophonie.

Ce modèle de communication multimédia ne recense pas de manière exhaustive toutes les contraintes possibles inhérentes à ce type d'application. Il donne les principaux points et principales voies à considérer dans la conception d'un protocole de régulation de débit dédié aux transmissions multimédia.

2.3 Un nouveau protocole de débit TCP-compatible adapté aux flux multimédia

Nous avons vu dans le chapitre précédent que différentes techniques de contrôle de congestion dites TCP-compatibles ont été proposées dans la littérature. Les techniques *basées-fenêtre* et AIMD induisant généralement un comportement du trafic très instable non souhaitable pour la transmission de flux multimédia [FHP00]; nous nous intéresserons donc ici a l'utilisation d'un modèle.

Dans cette section, nous soulignons tout d'abord la non adéquation des protocoles existants au transfert de flux multimédia. Puis nous proposons un nouveau protocole de contrôle de congestion dédié **basé-modèle** utilisant le protocole RTP/RTCP (*Real-time Transport Protocol/Real-time Transport Control Protocol*).

2.3.1 Limites des protocoles existants

Les différents protocoles proposés dans la littérature font l'hypothèse de flux continus mais ne prennent pas en compte les caractéristiques des flux multimédia temps-réel à transmettre. Ainsi le modèle le plus communément utilisé [PFTK98] (notamment

dans le protocole TFRC [FHPW00] - cf chapitre 1.3), estime le comportement d'une session TCP en état stationnaire transmettant des paquets de taille s. La plupart des sessions TCP sur l'Internet liées au Web ou FTP transfère des paquets de taille MSS (*Maximum Segment Size*) ou MTU (*Maximum Transfer Unit*). Le MSS représente la taille maximum que peut prendre un segment TCP sur le lien. Ainsi, pour la plupart des sessions TCP le paramètre s est fixé à MSS. Pour des applications multimédia, la situation est différente.

2.3.1.1 Paquets de tailles variables

Pour certaines applications, il est en effet nécessaire de pouvoir utiliser des paquets de tailles différentes. Ainsi, prenons par exemple la transmission de voix sur IP avec RTP. Pour des raisons de délai il est conseillé d'utiliser des paquets de taille faible (e.g. 140 octets) et, du fait notamment des silences, ces paquets ne sont pas de tailles constantes. La transmission dynamique et robuste de flux vidéo requiert également la définition de politiques de mise en paquet intelligentes. Les règles de paquetisation communément choisies mènent naturellement à la constitution de paquets de tailles variables [Ba98].

2.3.1.2 Equité?

Dans les modèles classiques tels que celui donné par l'équation (1.2) [PFTK98], le débit estimé est linéairement dépendant de la valeur du paramètre de taille de paquet s. Ainsi, si une application utilisant des petits paquets par exemple de taille 150 octets est en compétition avec une application utilisant des paquets de taille 1500 octets alors celle-ci aura le droit à dix fois moins de débit. Cette situation ne semble pas très équitable.

L'utilisation d'une taille moyenne de paquets comme valeur pour le paramètre s pourrait sembler une solution envisageable. Toutefois, en conséquence directe de la dépendance linéaire entre la taille des paquets et le débit calculé, les applications faisant ce choix sont également pénalisées par rapport aux applications TCP. De plus, lorsque l'on essaye d'adapter le débit des flux aux variations de bande passante du réseau, dans le cas d'application temps-réel par exemple, les variations de taille de paquets sont difficilement prédictibles et sont *surtout* très étroitement liées au débit d'émission.

2.3.1.3 TCP-compatibilité revisitée

Le concept *TCP-friendly* ou *TCP-compatible* a été introduit dans [MF97, FF99] et a évolué au cours du temps pour aboutir dans [BCC+98] de la facon suivante : un flux est dit *TCP-compatible* si il se comporte comme un flux TCP en présence de congestion et si en état stationnaire il n'utilise pas plus de bande passante qu'une session TCP dans des conditions comparables (pertes, tailles de paquets, délais,...).

Toutefois, cette définition (ainsi que les précédentes), considèr comme équitable qu'un flux utilisant des paquets plus petits ait le droit à moins de bande passante qu'une session TCP ou qu'une session utilisant des paquets de tailles supérieures. C'est

pourquoi nous avons décidé de redéfinir la notion de TCP-compatibilité en l'adaptant de la manière suivante.

Définition 2.3.1 (TCP-compatible) *Un flux est dit* TCP-compatible *si il se comporte comme un flux TCP en présence de congestion et si en état stationnaire il n'utilise pas plus de bande passante qu'une session TCP transmettant des paquets de tailles MSS et fonctionnant dans des conditions comparables (pertes, délais, ...).*

Nous considérons ici qu'une application ne doit pas pâtir de la taille ses paquets. Nous montrerons dans la suite que cette nouvelle définition n'entraine pas de comportement inéquitable ni envers TCP ni entre les différents flux TCP-compatibles.

2.3.2 Modèle de débit TCP utilisé

Considérant cette nouvelle définition, nous avons choisi afin de partager équitablement la bande passante avec TCP, de nous baser sur le modèle utilisé dans TFRC et de fixer le paramètre `s` dans l'équation (1.2) au MSS du lien. Nous avons, de plus, fixé le paramètre b à 1, considérant que chaque rapport ACK n'acquitte qu'un seul paquet. L'équation modélisant TCP devient donc :

$$B = \frac{MSS}{RTT\sqrt{\frac{2p}{3}} + T_o 3\sqrt{\frac{3p}{8}}p(1+32p^2)}, \tag{2.1}$$

avec RTT le temps d'aller-retour entre l'émetteur et le récepteur, et p le taux d'événements de pertes. Le terme T_o représente le délai avant retransmission de l'algorithme TCP.

Nous considérons ici que même si la taille des paquets varie, son impact sur le RTT et sur T_o est quasi-nul. Nous verrons, par contre, que l'utilisation de MSS au numérateur à des incidences sur le taux d'événements de pertes p.

2.3.3 Estimation des paramètres du modèle

Notre but étant de réguler le débit de flux multimédia, il semble assez naturel de tenir compte des propriétés intrinsèques de ces flux dans l'estimation des paramètres. Nous montrerons dans la suite que cette approche n'est pas en inadéquation avec les besoins de TCP-compatibilité et peut mener à une amélioration de la qualité de service tout en maintenant une utilisation comparable de la bande passante.

La prédiction de bande passante peut-être réalisée soit par l'émetteur soit par le récepteur. Cependant, tandis que le RTT et par conséquent le paramètre de délai T_o peuvent être estimés aux deux extrémités, le taux d'événements de congestion p doit être déterminé par le récepteur. Afin de simplifier les choses, nous avons choisi d'estimer tous les paramètres (RTT, T_o et p) et de prédire la bande passante du côté récepteur. La valeur prédite de la bande passante sur une fenêtre est ensuite acheminée vers la source dans un rapport de réception RTCP (Receiver Report: RR) en utilisant le mécanisme d'extension décrit dans [SCFJ96]

2.3.3.1 Fréquence des rapports de réception

L'adaptation de débit d'une source vidéo, via des changements de fréquence temporelle ou l'adaptation du pas de quantification, se fait en règle générale à certains instants clés du processus d'encodage. C'est pourquoi nous avons choisi ici de synchroniser la réception des rapports de réception avec la frontière d'une image (instant de début d'encodage) tout en respectant les recommandations d'utilisation de la bande passante réservée à RTCP. D'après [SCFJ96], la bande passante RTCP ne doit pas excéder 5% de la bande passante totale allouée à la session SB, avec 75% réservé aux rapports de réception et 25% aux rapports d'émission.

Ainsi, nous avons défini la fréquence δ_{feed} entre deux rapports en fonction du RTT et fréquence temporelle FR (*Frame Rate*) représentant le nombre d'images codées par seconde :

$$\delta_{feed} = \frac{\lceil max(S_{RTT}, \frac{Avgsize \times NR}{0.75 * 0.05 * SB}) \times FR \rceil}{FR}, \tag{2.2}$$

avec NR le nombre de récepteurs dans la session. Le terme $Avgsize$ représente la taille moyenne des rapports RTCP et S_{RTT} représente une estimation lissée du RTT (son calcul est décrit dans la section 2.3.3.2).

Ceci permet une periodicité de rapports en phase avec un nombre entier de trame pour une source audio ou vidéo tout en respectant les contraintes d'utilisation de bande passante de RTCP. En effet, il est montré dans [PYM99] que pour les transmissions point à point, une utilisation de 5% de la bande passante permet d'avoir pratiquement un rapport de réception par paquet reçu.

2.3.3.2 Estimations du RTT et du délai de retransmission (T_o)

L'estimation du RTT se fait en se basant sur une moyenne des mesures récentes. On mesure le RTT courant en utilisant les rapports RTCP comme décrit dans la partie B.2. A chaque fois qu'une mesure est réalisée m_{RTT}, une valeur lissée du RTT, notée S_{RTT}, est mise à jour en utilisant une moyenne exponentielle pondérée (EWMA) :

$$S_{RTT} = \alpha S_{RTT} + (1 - \alpha) m_{RTT} \tag{2.3}$$

avec α facteur de lissage (choisi à 0.9).

Le paramètre de délai de retransmission T_o est quant à lui estimé comme dans [Jac88, Jac90] par :

$$T_o = S_{RTT} + 4 \times D_{RTT} \tag{2.4}$$

avec D_{RTT} une valeur lissée (par une moyenne exponentielle pondérée) de la variance du RTT.

2.3.3.3 Estimation du taux d'évènement de congestion (p)

La méthode d'estimation des pertes est un élément clé du protocole afin de garantir des variations de débit douces ainsi qu'un partage équitable de la bande passante

avec les autres sessions ou connexions [RR99]. Estimer le taux de pertes, en faisant le rapport entre le nombre de paquets perdus et le nombre total de paquets transmis, ne modélise pas exactement le comportement de TCP face aux pertes. En effet, la plupart des implémentations TCP (Sack, New Reno, Tahoe) ne divisent pas leur fenêtre de congestion par deux à chaque fois qu'une perte intervient pendant un RTT.

Ainsi, comme dans [HF98, PKTK99a, TZ99, FHPW00, HPFW01], nous considérons ici la notion de taux d'**évènement de congestion** ou **évènement de pertes**. Toutefois, afin de prendre pleinement en compte les paquets de tailles variables dans la transmission de flux audiovisuels, nous proposons ici une méthode d'estimation du taux d'évènement de congestion révisée. L'idée ici est d'utiliser le modèle de manière correcte et de faire en sorte que tous les paramètres utilisés pour la prédiction de bande passante TCP-compatible soient cohérents entre eux.

2.3.4 Notion d'événement de congestion revisitée

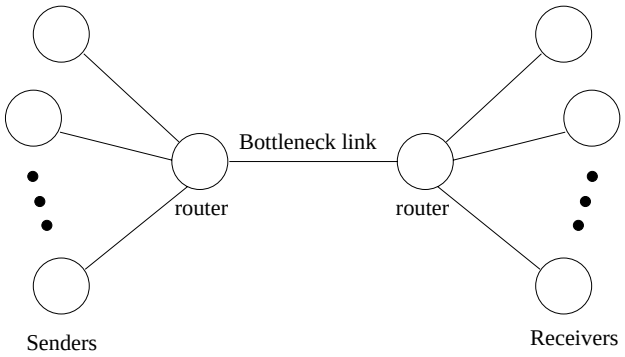

FIG. 2.1 – *Topologie de simulation de "goulot d'étranglement" (*bottleneck*).*

Dans cette partie, nous montrons que l'algorithme *basé-paquet* utilisé dans TFRC n'est plus adapté pour la prise en compte de paquets de tailles variables. Afin de montrer l'inadéquation du protocole TFRC à la prise en compte de paquets de tailles variables, nous avons expérimenté l'algorithme TFRC (avec le simulateur de réseau NS2) en considérant la topologie illustrée sur la figure 2.1. La topologie choisie consiste à connecter n sources avec n récepteurs à travers un lien commun appelé goulot d'étranglement (ou lien *bottleneck*). La capacité de ce lien est fixée à 1.5 Mb/s et le délai de transmission sur ce lien est fixé à 100 ms. Afin de simuler des paquets de tailles variables, les tailles des paquets sont choisies de manière aléatoire et prennent des valeurs comprises entre 150 et 1000 octets. Le MSS choisi dans l'équation est fixée à 1000 octets. On peut observer sur la figure 2.2 que l'estimation du taux d'évènement de congestion basé paquet utilisée [FHPW00] n'est pas adaptée à ce type de flux. Considérer comme un unique évènement de congestion toutes les pertes qui interviennent dans une fenêtre temporelle d'un RTT mène à une sous-estimation de l'état de congestion du réseau du

fait du nombre élevé de paquets considérés comme non perdus. Le flux devient alors plus agressif qu'une connexion TCP classique. Il apparait alors nécessaire de définir une nouvelle technique d'estimation de taux d'événements de pertes prenant en compte la taille effective des paquets transmis.

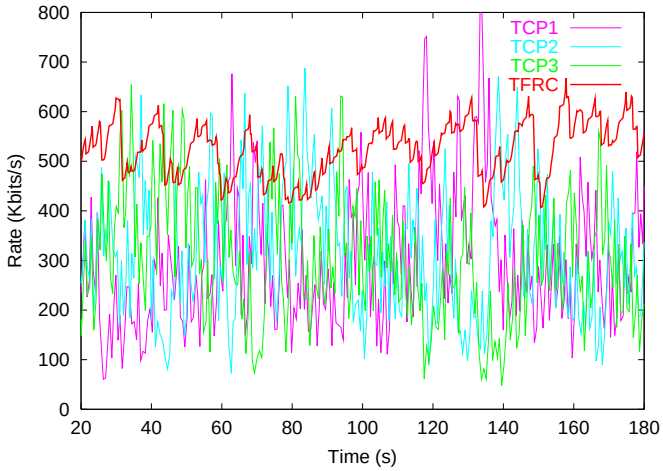

FIG. 2.2 – *Débits respectifs d'un flux TFRC et de trois flux TCP sur un même lien à 1.5 Mbit/s.*

2.3.4.1 Evènement de congestion et Intervalle de pertes

L'estimation des paramètres et notamment du taux de pertes se fait sur une fenêtre temporelle que l'on appellera "fenêtre d'historique".

Dans les précédents travaux [FHPW00], toutes les pertes intervenant pendant un RTT étaient considérées comme un unique **évènement de congestion**. Ainsi, lorsqu'une perte de paquet est observée, un nouvel évènement de congestion est ajouté à la fenêtre d'historique. Un **intervalle de pertes** est alors défini comme le nombre de paquets entre deux évènements de congestion. La fenêtre d'historique est alors composée de n intervalles de temps ou intervalles de pertes notés τ_i.

Afin de gérer les paquets de tailles variables par la prise en compte des quantités de données transmises et perdues, nous avons donc révisé les définitions des notions d'évènement de congestion et d'intervalle de pertes. Le mécanisme de calcul que nous proposons ici se base sur l'utilisation de deux compteurs (en octets) : SizeLost représentant la quantité cumulée de données perdues sur le RTT courant et SizeInterval représentant la quantité cumulée de données transmises depuis le dernier événement de congestion.

Ainsi, lorsqu'un paquet est reçu sa taille est rajoutée à SizeInterval. A l'inverse, lorsqu'une perte de paquet intervient, elle n'est pas rajoutée directement à la fenêtre d'historique comme un évènement de congestion mais la taille du paquet perdu est ajoutée à SizeLost (sauf si un événement de congestion a déjà été ajouté pour le RTT

courant). Un nouvel événement de congestion sera ajouté à la fenêtre d'historique dans les deux cas suivants :

- Si SizeLost > MSS : il y a ajout d'un événement de congestion. Le complément (i.e. SizeLost − MSS) sera alors rajouté au compteur du prochain intervalle de pertes.
- A la fin du RTT courant, si SizeLost > 0 il y a également ajout d'un événement de congestion.

Dans ces deux cas, lorsqu'un ajout est fait, l'intervalle de pertes courant τ_i est considéré comme clos. La notion d'*intervalle de pertes*, redéfinie donc comme la quantité totale de données en unité MSS transmises entre deux évènements de congestion, est calculée comme suit :

$$\tau_i = \frac{\mathsf{SizeInterval}}{MSS} \tag{2.5}$$

Les deux compteurs SizeLost et SizeInterval sont ensuite remis à 0. Le compteur SizeInterval est alors incrémenté si nécessaire.

La taille des paquets perdus peut-être calculée en utilisant le champ d'extension du paquet RTP OS décrit dans la section 2.3.7. Comme dans [FHPW00][HPFW01], le "surplus" de données perdues pendant le RTT courant n'est pas considéré comme perdu dans le calcul du taux d'événement de congestion.

Remarque:
Il est intéressant de noter que notre approche est compatible avec TFRC puisque dans le cas particulier où les paquets sont de tailles constantes le taux d'évènements de congestion est le même avec les deux méthodes. Dans la suite nous appellerons notre protocole s'appuyant sur la notion révisée d'évènement de congestion, **PSA-TFRC** (*Packet Size Adapted - TFRC*).

2.3.4.2 Moyenne pondérée des intervalles de pertes

Un point clé du calcul du taux d'évènement de congestion est la taille de la fenêtre d'observation. Nous calculons ici le taux d'évènement de congestion, sur un nombre d'intervalle de pertes plutôt que sur une fenêtre temporelle fixe définie a priori afin d'obtenir un mécanisme plus réactif. De plus, afin d'éviter des changements brusques dans le calcul de taux d'évènement de congestion dûs à des anciens intervalles de pertes non représentatifs, la méthode proposée dans [FHPW00] consiste à faire une moyenne pondérée de ces n derniers intervalles de pertes. Les intervalles les plus récents ont ainsi des poids égaux tandis que les plus anciens ont des poids plus faibles et décroissants avec le temps. Le taux d'évènement de congestion se calcule alors comme suit :

$$\hat{p} = \frac{\sum_{i=1}^{n} w_i}{\sum_{i=0}^{n-1} w_i \tau_i}. \tag{2.6}$$

avec w_i les poids que nous utiliserons ici définis par

$$w_i = \begin{cases} 1\,, & 1 \leq i \leq n/2, \\ 1 - \frac{i-n/2}{n/2+1}, & n/2 < i \leq n. \end{cases} \tag{2.7}$$

Il faut noter que le calcul proposé ici ne prend pas en compte les éventuels derniers paquets recus dans l'intervalle le plus récent T_0 qui n'est pas borné à l'instant du calcul du taux d'évènement de congestion. En effet, aucune perte clôturant cet intervalle n'est intervenue. L'idée est donc de considérer ce dernier intervalle dans le calcul lorsque celui-ci contribue à faire décroître le taux d'évènement de congestion. Pour cela, nous utiliserons ici un facteur de correction.

Ainsi, lorsque le taux d'évènement de congestion $\hat{p}_{old}^{nd}$ calculé sans prendre en compte le dernier intervalle devient non significatif en comparaison du taux d'évènement de congestion $\hat{p}_0$ calculé sur le dernier intervalle (i.e. quand $\hat{p}_0$ est inférieur à deux fois $\hat{p}_{old}^{nd}$), un facteur de correction est appliqué. Ce mécanisme a pour but d'atténuer les poids donnés aux intervalles de pertes et donc d'oublier progressivement l'historique. Ce facteur de correction prend la forme d'un facteur de dépondération d_i défini par

$$d_i = \begin{cases} 1\,, & i = 0, \\ max(\ 0.5\ , min(\frac{2\,\hat{p}_0}{\hat{p}_{old}^{nd}}\ ,\ 1\ )\ )\,, & i > 0. \end{cases} \tag{2.8}$$

Le calcul du taux d'évènement de congestion est finalement donné par

$$p = min(\hat{p}_{new}, \hat{p}_{old}), \tag{2.9}$$

avec

$$\hat{p}_{old} = \frac{\sum_{i=1}^{n} d_i w_i}{\sum_{i=1}^{n} d_i w_i \tau_i} \quad et \quad \hat{p}_{new} = \frac{\sum_{i=1}^{n} d_{i-1} w_i}{\sum_{i=0}^{n-1} d_i w_{i+1} \tau_i}. \tag{2.10}$$

Dans les expérimentations décrites ici, le nombre d'intervalle de pertes considérés n est fixé à 8.

2.3.4.3 Phase d'initialisation : Slow-start

La phase d'initialisation adoptée ici diffère du mécanisme dit de *slow-start* décrit dans [FHPW00]. Dans [FHPW00], la phase d'initialisation mise en oeuvre est proche de TCP en ce sens qu'à chaque RTT le débit utilisé est multiplié par deux et ceci jusqu'à obtenir une perte. Après la première perte, la fenêtre d'historique est initialisée en prenant comme bande passante disponible la bande passante actuelle divisée par deux. Puis, l'équation (2.1) est utilisée. Toutefois, le débit initial est fixé à un paquet ce qui induit un temps de convergence très important de l'ordre d'au moins 20 secondes pour des gammes de débits classiques.

Pour des applications multimédia, requièrant un minimum d'interactivité il est impératif que le temps de convergence soit le plus court possible. En effet, il est conseillé

de n'envoyer les premières données réelles qu'après la phase de Slow-start. Aisni dans notre approche le débit initialement requis est fixé à R_{min} (i.e. débit minimal requis par l'application). Puis le débit dans la phase de slow-start B_t^{slow} est donné par :

$$B_t^{slow} = min(2 \times B_{t-1}^{slow}, min(2 \times g_t, R_{max})) \tag{2.11}$$

avec g_t la quantité de données effectivement reçues dans le RTT précédent, B_{t-1}^{slow} la précédente requête et R_{max} le débit maximal auquel peut émettre la source. Après la première perte, la fenêtre d'historique est ré-initialisée en prenant g_t comme bande passante disponible et l'on utilise alors l'équation (2.1) pour prédire le débit.

2.3.5 Analyse de l'équité Inter/Intra protocoles

Nous avons implémenté l'algorithme proposé dans le simulateur de réseau NS2 afin de tester son comportement dans des conditions contrôlées. Toutes les simulations décrites ici l'ont été en considérant des routeurs de type "*drop-tail*" qui sont les routeurs classiquement présents sur l'Internet. L'équité de partage de la bande passante a été analysée en utilisant la topologie de réseau représentée sur la figure 2.1. Plusieurs scénarios de simulations ont été testés afin de valider le bon comportement de notre protocole. Comme pour les expériences realisées avec TFRC décrites dans la section 2.3.4 la taille de chaque paquet est choisie aléatoirement et prend ses valeurs entre 150 et 1000 octets. Le MSS a également été fixé à 1000 octets.

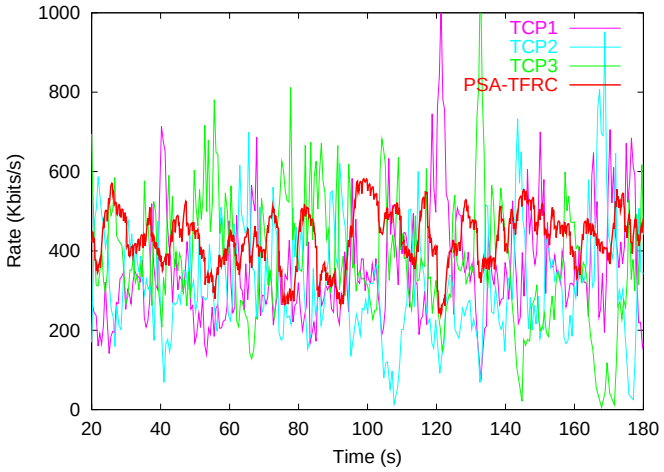

FIG. 2.3 – *Débits respectifs d'un flux PSA-TFRC et de trois flux TCP sur un même lien à 1.5 Mbit/s avec un délai de transmission de 50 ms.*

Les figure 2.3 et 2.4 illustrent le débit d'un flux PSA-TFRC et de trois flux TCP partageant le même lien *bottleneck* à 1.5 Mb/s avec respectivement deux délais de transmission du lien sous-jacent de 50 ms et 100 ms. Le débit moyen du flux régulé avec notre algorithme (PSA-TFRC) est très proche, tout en restant en dessous, du débit

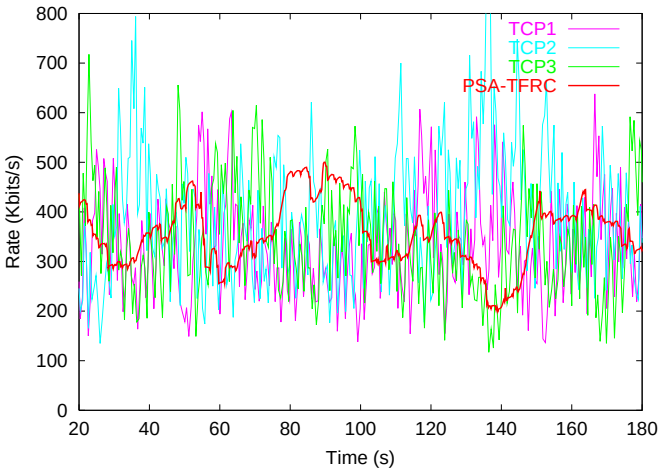

FIG. 2.4 – *Débits respectifs d'un flux PSA-TFRC et de trois flux TCP sur un même lien à 1.5 Mbit/s avec un délai de transmission de 100 ms.*

moyen des flux TCP. Ceci montre clairement que notre protocole partage équitablement la bande passante avec TCP. Comme on peut le voir également, la haute fréquence des rapports, fonction du délai de transmission du lien, accroît la réactivité du mécanisme.

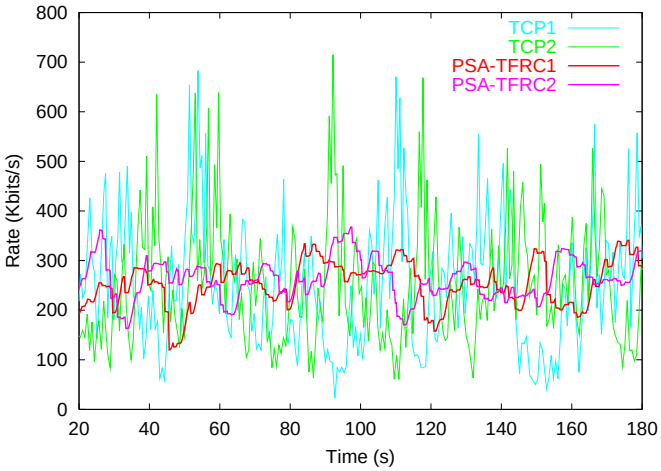

FIG. 2.5 – *Six flux PSA-TFRC et six flux TCP sur ur un même lien à 3 Mbit/s.*

Nous avons également étudié l'équité dans le partage de la bande passante lorsque plusieurs flux PSA-TFRC partage un même lien avec plusieurs connexions TCP. L'expérience réalisée ici consiste à mettre en compétition 6 flux TCP avec 6 flux PSA-TFRC sur un lien bottleneck à 3 Mb/s ayant un délai de transmission de 100 ms. Pour des raisons de clarté, la figure 2.5 montre les résultats obtenus pour 2 des 6 flux PSA-TFRC

et TCP.

On peut ainsi voir que les flux PSA-TFRC partagent équitablement la bande passante et que leur débit moyen est en phase avec le débit moyen des connexions TCP.

2.3.6 Propriétés des variations de débit

Comparativement à TCP qui augmente son débit d'émission d'un MSS par RTT en l'absence de congestion, et qui décroît son débit par deux lorsqu'intervient une perte de paquet, notre protocole montre des variations de débit nettement plus lissées.

Il faut ainsi au moins 5 RTT pour diviser son débit d'émission par 2. En présence, de RTT stables et sans l'utilisation du mécanisme d'*oubli de l'histoire* (basé sur le facteur de dépondération) le débit d'émission augmente d'au plus 0.14 MSS par RTT.

Après une période prolongée de non congestion, le mécanisme d'oubli de l'histoire est activé, et le débit d'émission augmente d'au plus 0.28 MSS par RTT.

Ces propriétés ont été vérifiées dans toutes les expérimentations réalisées et décrites dans ce manuscrit.

2.3.7 Signalisation RTP/RTCP

Afin de connaître la quantité de données perdues, nous proposons d'ajouter un champ spécifique dans la zone d'entête de la zone de données utiles (*payload*) comme suit :

OS (Offset Stream) : Ce champ de 16 bits donne le décalage (offset) en mot de 32 bits, du premier octet de données contenues dans le paquet par rapport au premier octet de ce flux. C'est un compteur cyclique modulo $2^{16} \times 4$. Avec ce champ et la taille de la zone de données utiles, le récepteur peut facilement calculer τ_i.

Toutes les valeurs relatives à l'état du canal sont, quant à elles, transmises à la source dans des rapports RTCP (RR) toutes les δ_{feed} secondes. La bande passante prédite et le délai de transmission (one-way) sont transmises dans des RR augmentés avec les champs suivants:

EB (Expected Bitrate) : Ce champ de 16 bits donne la valeur de la bande passante prédite en kbits/s.

OW (One Way) : Ce champ de 16 bits donne le temps de transmission (dans un sens) mesuré avec les horloges asynchrones de la source et du récepteur. La valeur mise dans le champ OW est mesurée en utilisant le champ "NTP timestamp" (c.f. Annexe B) du rapport d'émission RTCP et s'exprime en millisecondes. L'utilisation de ce champ est décrite dans la section 2.4.4.

2.3.8 Conclusion de la section

Le développement grandissant des applications multimédia sur l'Internet, avec notamment l'arrivée de l'audio et de la vidéo chez les particuliers, ne fait qu'accroître la nécessité de protocoles de contrôle de congestion dédiés à ce type d'applications.

C'est pourquoi, la conception d'un nouveau protocole applicatif appelé DCCP (*Datagram Congestion Control Protocol*) est actuellement à l'étude à l'IETF [KHFP02]. Ce protocole pourrait s'appuyer sur l'algorithme TFRC. Aucun travaux connus à ce jour n'ayant été réalisés et proposés concernant notamment la prise en compte de paquets de tailles variables, notre algorithme semble être une bonne alternative à TFRC. Une proposition à l'IETF est envisageable.

2.4 Régulation de la source

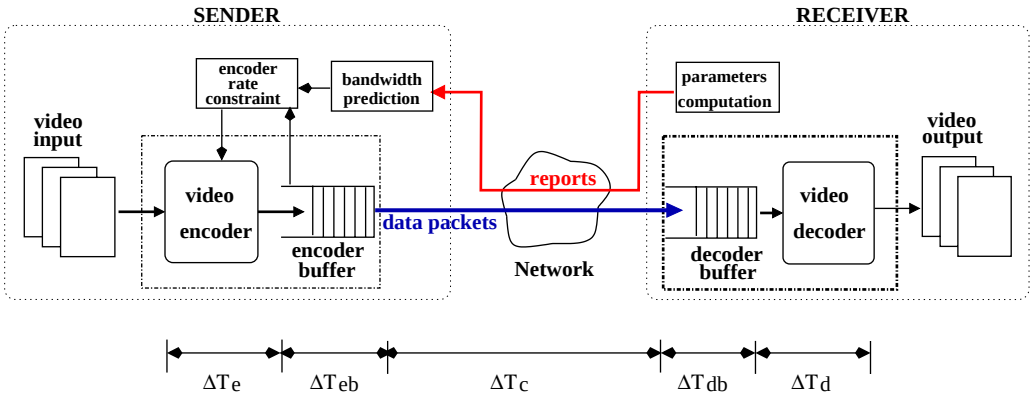

FIG. 2.6 – *Chaîne de transmission vidéo.*

Une fois la bande passante évaluée, le problème consiste à transcrire cette valeur prédite en une contrainte de débit intelligible pour la source vidéo. Deux points principaux sont visés ici : maximiser l'utilisation de la bande passante et minimiser le nombre de retards de paquets dûs aux variations de délai du réseau (i.e. la gigue). Ces retards entraînent une diminution de la qualité de la vidéo reconstruite.

Très peu de travaux ont été réalisés dans ce domaine. De plus, les rares approches proposées dans la littérature font l'hypothèse, soit d'une source à débit constant, soit d'un réseau à débit constant. Les solutions proposées sont des mécanismes uniquement orientés récepteur, qui considèrent donc que c'est au récepteur de régler les éventuels problèmes de retards par un dimensionnement judicieux du *buffer* de réception. Il faut noter, également, que les gammes de valeurs prises pour ces délais restent très élevées et ne sont pas propices à des utilisations dans le cadre d'applications interactives. Enfin, aucun schéma ne considère une source s'adaptant de manière dynamique aux fluctuations de bande passante induites par un mécanisme de protocole de contrôle de congestion.

Dans la suite, nous décrirons trois solutions hybrides émetteur/récepteur, allant de la simple conversion de la bande passante prédite en un budget par image à un modèle "global" tenant à la fois compte de l'état des *buffers* d'émission et de réception et des contraintes temporelles de la source. Ces trois approches considèrent une chaîne de

communication vidéo classique telle que celle illustrée sur la figure 2.6.

2.4.1 Bande passante prédite comme contrainte: injection directe

L'approche la plus simpliste consiste à réintroduire la bande passante prédite notée $R_N(t)$ directement comme une contrainte pour l'encodeur $R_e(t)$. Considérons que les rapports sont reçus par le récepteur avec une périodicité de δ_{feed} définie dans la section 2.3.3.1. Alors, si la source reçoit un rapport à l'instant t_n, le budget alloué à la source vidéo sur l'intervalle de temps $[t_n, t_n + \delta_{feed}]$ est donné par :

$$\int_{t_n}^{t_n+\delta_{feed}} R_e(t) \leq \int_{t_n}^{t_n+\delta_{feed}} R_N(t)dt. \tag{2.12}$$

Soit R_e^i la quantité de bits nécessaire au codage de l'image i et $Ind(t_n, t_n + \delta_{feed})$ l'ensemble des numéros d'images à encoder entre les instants t_n et $t_n + \delta_{feed}$ (i.e. $i \in \{n, n+1, \ldots\}$). Alors, la contrainte de débit de l'encodeur (en bit) sur l'intervalle de temps $[t_n, t_n + \delta_{feed}]$ peut être convertie en une contrainte de débit pour chaque image donnée par

$$\sum_i R_e^i \leq \int_{t_n}^{t_n+\delta_{feed}} R_e(t)dt, \quad \forall i \in Ind(t_n, t_n + \delta_{feed}). \tag{2.13}$$

Comme on le verra dans la section 2.5, ce type d'approche simpliste (mais utilisée) souffre d'un nombre de retards de paquets très élevé. Les données arrivant en retard étant considérées comme perdues, elles ont un impact négatif sur la qualité de vidéo perçue. Une ré-injection directe de la contrainte de réseau n'est pas réaliste pour plusieurs raisons dont une évidente, qui est le délai naturel entre l'instant de prise en compte de la contrainte de débit par l'encodeur et l'instant ou celui-ci produira des données au nouveau débit cible. De plus, cette approche ne prend en compte ni l'état des *buffers*, ni les contraintes de délai de bout-en-bout de la vidéo.

Même si les performances de ce type d'approche peuvent être améliorées en augmentant la taille du pré-chargement initiale dans le *buffer* du côté récepteur, le nombre de retards demeure important. De plus, une taille de *buffer* de réception élévée introduit une latence qui n'est pas compatible avec des applications nécessitant un fort dégré d'interactivité .

Dans la suite, nous appelerons cette approche **DI-SRC** (pour *Direct Introduction - Source Rate Control*).

2.4.2 Considération de l'état des buffers: "flushing buffer"

La non prise en compte de l'état de remplissage du *buffer* de l'encodeur explique en partie le nombre élevé de retards observés. Une méthode simple pour en tenir compte consiste à transcrire la bande passante prédite en une contrainte pour l'encodeur de la façon suivante :

$$\int_{t_n}^{t_n+\delta_{feed}} R_e(t)dt \quad \leq \quad \int_{t_n}^{t_n+\delta_{feed}} R_N(t)dt - B_e(t_n), \tag{2.14}$$

avec $B_e(t_n)$ le nombre de bits restants dans le *buffer* de l'encodeur à l'instant t_n.

Soit $\tilde{R}_e^i$ le nombre de bits appartenant à la $i^{ème}$ image restant dans le *buffer* d'émission et p le nombre d'images dans le *buffer* à un instant donné t_n. Alors, $B_e(t_n)$ peut s'exprimer par

$$B_e(t_n) = \sum_{i=1}^{p} \tilde{R}_e^i. \tag{2.15}$$

Cette approche permet de réduire de manière significative le nombre de retards (cf section 2.5), mais cette diminution se fait au prix d'une utilisation sous-optimale de la bande passante. De plus, elle mène également à des changements brusques, non souhaitables, de contraintes de débit pour l'encodeur. Ces fortes variations de contraintes entraînent une qualité très instable du signal reçu. Enfin, le sous-remplissage périodique du *buffer* d'émission peut durer jusqu'à 40 ms (pour une source à 25 Hz), ce qui mène à une transmission très sporadique influant de manière négative sur l'ensemble de la session.

Dans la suite, nous appelerons cette approche **FB-SRC** (pour *Flushing Buffer - Source Rate Control*).

2.4.3 Prise en compte de la chaîne de communication : modèle global

Bien qu'améliorant les résultats en terme de réduction des retards, l'approche précédente est simpliste et engendre une sous-utilisation de la bande passande disponible. De plus, cette dernière ne prend pas en compte les différentes contraintes de délais liées à la transmission de sources multimédia. C'est pourquoi, nous proposons ici un modèle global tenant compte dans la régulation de la source des différents éléments physiques ou techniques qui composent une chaîne de communication multimédia de bout-en-bout (c.f. figure 2.6).

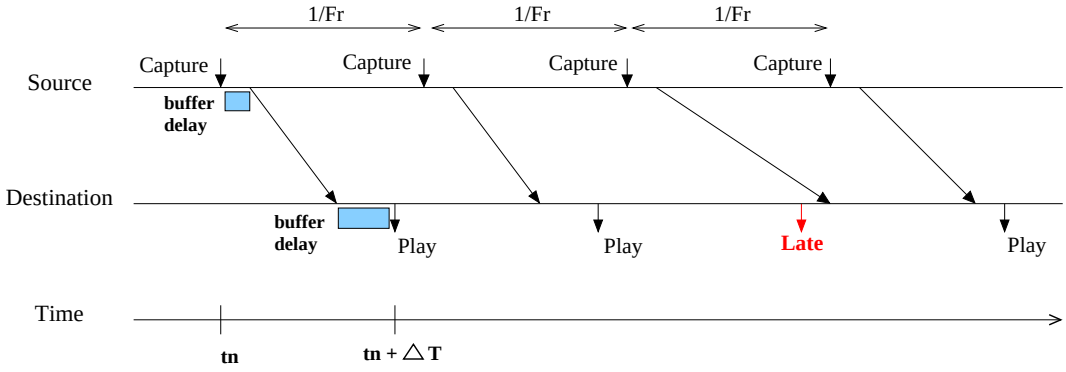

FIG. 2.7 – *Illustration du problème de retard.*

Soit ΔT le délai de bout-en-bout associé à chaque image,

$$\Delta T = \Delta T_e + \Delta T_{eb} + \Delta T_c + \Delta T_{db} + \Delta T_d, \tag{2.16}$$

avec ΔT_e et ΔT_d représentant respectivement les délais d'encodage et de décodage, $\Delta T_e b$ et $\Delta T_d b$ les délais de passage dans les *buffers* d'émission et de réception et ΔT_c le temps de transmission sur le canal.

Ainsi, afin de satisfaire les contraintes relatives au temps-réel, une image capturée à l'instant t_n doit être prête pour affichage à l'instant $t_n + \Delta T$. En effet, les images doivent être affichées du côté récepteur avec la même fréquence à laquelle elles sont capturées du côté émetteur. Par exemple, pour une source à 25 Hz, ce délai inter-image est de 40 ms (cf. figure 2.7). Ce qui implique qu'une image produite à l'instant t_n doit être émise avant un certain instant noté $t_n + \delta_{buff}$ pour ne pas arriver en retard. Le paramètre δ_{buff} représente, en fait, le délai maximal de présence dans les *buffers* (émission + réception) des données. Il peut être calculé de la manière suivante :

$$\delta_{buff} = \Delta T_{eb} + \Delta T_{db} = \Delta T - \Delta T_e - \Delta T_c - \Delta T_d. \tag{2.17}$$

La contrainte de délai concernant l'évitement de retards peut être traduite en une contrainte de débit pour le codeur vidéo. Ainsi, soit δ_{feed} la fréquence de réception des rapports, alors la bande passante prédite nous donne une borne supérieure pour la contrainte associée au codeur qui s'exprime par

$$B_e(t_n) + \int_{t_n}^{t_n+\delta_{feed}} R_e(t)dt \leq \int_{t_n}^{t_n+\delta_{feed}+\delta_{buff}} \tilde{R}_N(t)dt. \tag{2.18}$$

Le terme $B_e(t_n)$ correspond à la quantité de bits restant dans le *buffer* d'émission à l'instant t_n. Le terme $\tilde{R}_N(t)$ représente la bande passante prédite sur l'intervalle inter-rapports courant (i.e. sur l'intervalle $[t_n, t_n + \delta_{feed}]$) extrapolée sur l'intervalle $[t_n + \delta_{feed}, t_n + \delta_{feed} + \delta_{buff}]$ prenant ainsi en compte les délais de mise en *buffer*. Du fait de la stabilité de notre protocole réseau en termes de variations de débit (cf. section 2.3.6), l'erreur induite par cette extrapolation reste faible. De plus, cette erreur, peut être également atténuée en utilisant une stratégie de codage et de gestion du *buffer* d'émission adaptée (cf section 2.4.5).

La contrainte donnée par l'inégalité 2.18 donne la borne supérieure théorique. En pratique, afin de réduire encore les risques de retards dûs à une gigue élévée mal prédite et afin d'éviter un sur-remplissage du *buffer* d'émission, nous utilisons une borne supérieure plus contraignante

$$\int_{t_n}^{t_n+\delta_{feed}+\delta_{\hat{buff}}} \tilde{R}_N(t)dt \leq \int_{t_n}^{t_n+\delta_{feed}+\delta_{buff}} \tilde{R}_N(t)dt \tag{2.19}$$

avec $\delta_{\hat{buff}} = min(\delta_{buff}, \Delta T_{eb}^{max})$ et ΔT_{eb}^{max} représente une borne supérieure souhaitable de délai de présence dans le *buffer* d'émission.

2.4.4 Mise à jour du délai de présence dans les buffers (δ_{buff})

Du fait de la gigue présente naturellement sur l'Internet, le délai de transmission ΔT_c ne peut pas être considéré constant. En conséquence, les paramètres δ_{feed} et δ_{buff},

dépendants de la gigue du réseau, se doivent d'être ré-estimés périodiquement. Le paramètre δ_{feed} sera ré-estimé par l'équation (2.2) en utilisant des valeurs récentes du RTT.

La mise à jour de δ_{buff} est, par contre, moins simple à réaliser, toute la difficulté résidant dans l'estimation du délai de transmission ΔT_c. Une approche simple pour estimer ΔT_c pourrait consister à prendre $RTT/2$. Cependant, du fait de l'asymétrie des liens sous-jacents, en termes de temps de transit, cette valeur n'est qu'approximative [Pax97]. Les méthodes utilisées dans la littérature font l'hypothèse d'horloges émetteur et récepteur synchronisées. La synchronisation de ces horloges n'est pas un problème trivial et reste assez complexe à mettre en oeuvre.

Dans l'algorithme proposé ici, il n'est, en fait, pas nécessaire d'avoir une mesure précise de ΔT_c^t (i.e. ΔT_c à l'instant t). La différence entre le délai de transmission initial ΔT_c^0 et ce délai peut suffire. Pour le montrer, il suffit d'exprimer le délai de bout-en-bout ΔT à $t = 0$ et à t quelconque. Considérons des temps d'encodage et de décodage maximaux, donc égaux à $1/FR$, alors ΔT est donné par

$$\Delta T = \delta_{buff}^t + \Delta T_c^t + \frac{2}{FR} \tag{2.20}$$

et par

$$\Delta T = \delta_{buff}^0 + \Delta T_c^0 + \frac{2}{FR} \tag{2.21}$$

avec ΔT_c^0 et δ_{buff}^0 respectivement les délais de transmission et de présence dans les *buffers* calculés pour la première image. Il faut noter que δ_{buff}^0 est fixé a priori par un paramètre K (i.e. représentant la taille totale des *buffers* en nombre d'images à l'initialisation). En égalisant les équations (2.20) et (2.21), on obtient alors

$$\delta_{buff}^t = \delta_{buff}^0 + (\Delta T_c^0 - \Delta T_c^t) = \frac{K}{FR} + (\Delta T_c^0 - \Delta T_c^t) \tag{2.22}$$

Le calcul de ΔT_c^t (incluant $t = 0$) est réalisé en faisant la différence entre la valeur d'horloge de l'émetteur H_S lorsque le paquet quitte l'émetteur et l'instant d'arrivée H_R mesuré sur l'horloge du récepteur :

$$\Delta T_c^t = H_R - H_S. \tag{2.23}$$

Les horloges n'étant pas synchronisées, leur différence inclut non seulement le délai de transmission, mais également le décalage entre les deux horloges. Comme a priori, le décalage entre les deux horloges est constant et déjà comptabilisé dans ΔT_c^0, alors lorsque l'on applique l'équation (2.22), on peut facilement voir que ce décalage n'a aucun impact sur le calcul de δ_{buff}^t. Cette valeur de ΔT_c^t estimée est ensuite lissée en utilisant une moyenne exponentielle pondérée et transmise à l'émetteur dans le champ OW du RR RTCP décrit dans la section 2.3.7.

Dans la section 2.5, nous montrons l'intérêt d'une telle mise à jour dynamique pour le comportement de l'algorithme.

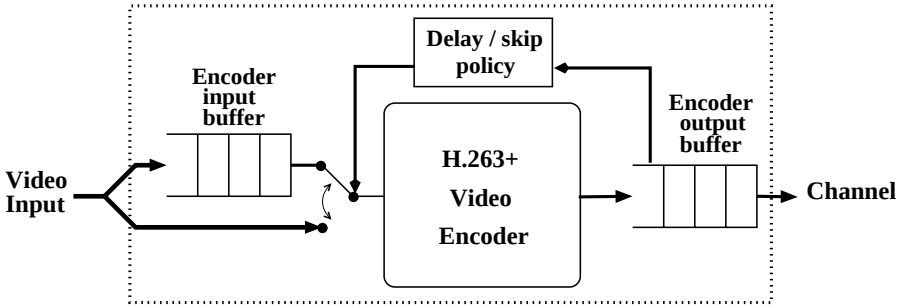

FIG. 2.8 – *Codeur H.263+ modifié.*

2.4.5 Adaptation de la source vidéo

2.4.5.1 Détection des retards potentiels pour garantir le délai de bout-en-bout

Le modèle précédent peut encore être raffiné en adaptant également le débit des images encore présentes dans le *buffer* d'émission afin d'éviter qu'elles arrivent en retard. En effet, le débit de ces images présentes dans le *buffer* peut-être supérieur à la nouvelle contrainte de débit, ce qui peut induire certains retards parmi celles-ci. Afin de limiter ce phénomène, nous avons rajouté un processus vérifiant les violations de délais potentielles des images présentes dans le *buffer* de l'encodeur. Soit $(n-p)$ l'index de l'image la plus ancienne présente dans le *buffer*, alors la contrainte de débit pour l'image $(n-k)$ présente dans le *buffer* est donnée par

$$\sum_{j=n-p}^{n-k} \tilde{R}_e^j \leq \int_{t_n}^{t_{n+1}+\frac{p-k}{FR}+\delta_{buff}} \tilde{R}_N(t)dt, \tag{2.24}$$

avec $k = p, ..., 1$ et $\tilde{R}_e^j$ représentant la quantité de données restantes relatives à la $i^{ème}$ image présente dans le *buffer* de l'encodeur. Si la relation précédente n'est pas vérifiée, le débit de l'image $n-k$ doit être réajusté, en accord avec l'équation (2.18). Cette régulation peut être traitée différement suivant l'algorithme de compression utilisé.

Si l'algorithme de compression s'appuie sur un mécanisme de prédiction temporelle (e.g. standards H.263+, MPEG-4, H.26L), le réajustement du débit de l'image $(n-k)$ affectera les images qui la suivent $(n-k), ..., (n-1)$. Afin d'éviter ce phénomène de dérive, les images $(n-k), ..., (n-1)$ doivent également être modifiées en conséquence. La solution décrite dans [HOK99] basée sur plusieurs *buffers* en parallèle contenant différentes versions quantifiées de groupe de blocs (GOB) ne prend pas en compte ces phénomènes de dérive. Le codec vidéo considéré ici, représenté sur la figure 2.8, s'appuie sur l'utilisation d'un second *buffer*. Ce dernier contient une copie des p images originales restant dans le *buffer* de l'encodeur. Dans le cas où l'image $(n-k)$ ne pourrait pas arriver à temps au décodeur, le codec réencode les images $(n-k), ..., (n-1)$ avec

la nouvelle contrainte de débit calculée avec l'équation (2.18). Le but d'un tel schéma est de simuler le comportement en terme d'adaptation de débit d'un codeur progressif.

La technique basée sur ces deux *buffers* utilisée dans ce codeur H.263+ peut naturellement être évitée avec un codeur scalable à granularité fine (FGS : *Fine Granular Scalability*) [RC99]. La stratégie de codage FGS fournit une solution naturelle au problème d'adaptation de débit d'images déjà codées. Si la relation (2.24) n'est pas satisfaite, la taille de l'images compressée $(n-k)$ peut facilement être réajustée en coupant dans son train binaire afin d'obtenir le débit désiré.

Il est possible que le paramètre δ_{buff} devienne négatif dans le cas où le délai de transmission augmente de manière importante. Dans ce cas, la fréquence temporelle (*frame rate*) de la source doit être réajustée.

2.4.5.2 Contrôle de la fréquence temporelle pour une qualité vidéo constante

Le codeur H.263+ utilisé ici intègre l'algorithme de régulation de débit du TMN 8 que nous avons modifié afin d'autoriser des débits variables. Afin d'ajuster au mieux son débit en accord avec le débit cible, l'algorithme intègre un mécanisme d'adaptation du pas quantification (au niveau des macroblocks) couplé avec une politique de saut d'image (en cas de dépassement de la contrainte). Cette adaptation se base sur les modèles débit-distorsion décrits dans [RCL97].

Cependant, les brusques changements de qualité (PSNR : *Peak Signal to Noise Ratio*) ou de fréquence temporelle engendrés par l'algorithme du TMN8 conduisent à des nuisances visuelles du rendu. Nous avons donc rafiné l'algorithme de sous-échantillonnage temporel afin de prendre en compte des variations "acceptables" de qualité entre images consécutives. Ainsi, soit P_l le PSNR de la dernière image encodée et T_{qual} un seuil donné, alors la qualité de l'image suivante devra se trouver dans l'intervalle $[P_l - T_{qual}, P_l + T_{qual}]$. Ce meilleur compromis entre fréquence temporelle et PSNR permet d'obtenir une plus grande stabilité de qualité et ceci même en présence de contraintes de débit variables. Le débit cible $\tilde{R}_e^n$ de l'image suivante est donc contraint par

$$\tilde{R}_e^n(\tilde{D}_{qual-}) \leq \tilde{R}_e^n \leq \tilde{R}_e^n(\tilde{D}_{qual+}), \tag{2.25}$$

où $[\tilde{D}_{qual-}, \tilde{D}_{qual+}]$ représente l'intervalle de distortion (i.e. contraint par T_{qual}).

Si le budget (en débit) de la prochaine image est inférieur à $\tilde{R}_e^n(\tilde{D}_{qual-})$, l'image n sera non codée et l'on codera l'image $n+1$ avec un débit supérieur. Si au contraire, sa valeur est supérieure à $\tilde{R}_e^n(\tilde{D}_{qual+})$, son budget sera contraint à cette limite afin de ne pas gâcher de la bande passante et également d'augmenter le budget des prochaines images à encoder.

2.4.5.3 Modèle débit-distorsion

Afin d'estimer le débit cible $\tilde{R}_e^n$, nous avons décidé d'utiliser ici une méthode basée sur les modèles débit et distorsion du codec H.263+ proposés dans [RCL97]. A partir de ces modèles, nous déduisons les distorsions et débits moyens de chaque image.

Considérons un pas de quantification moyen Q_{mean} pour chaque macrobloc de l'image j et un poids α_i égal à 1 pour chaque macrobloc, nous obtenons alors la mesure de distorsion suivante pour l'image j:

$$\tilde{D}_j = \frac{Q_{mean}^2}{12}. \tag{2.26}$$

Le modèle de l'encodeur permet ensuite d'obtenir :

$$\tilde{R}_e^i(\tilde{D}_j) = \sum_{i=1}^{N} B_i = A \sum_{i=1}^{N} (C_1 \frac{\sigma_i^2}{12\tilde{D}_j} + C_2) \tag{2.27}$$

où B_i est le coût (en bits) du $i^{ème}$ macrobloc de l'image j, N est le nombre de macroblocs encodés dans une image et A est le nombre de pixels dans un macrobloc (i.e. $A = 16^2$ pixels). Les termes C_1 et C_2 sont des constantes initialisées respectivement à 0.5 and 0 et mises à jour durant le processus d'encodage. Le terme σ_i représente, quant à lui, l'écart-type des valeurs de luminance et de chrominance du macrobloc (compensé en mouvement Intra ou Inter). Ces valeurs approximées de débit et distorsion suffisent dans les expérimentations réalisées ici.

Dans la suite, nous appellerons cette approche, utilisant le modèle global couplé avec le mécanisme d'adaptation de la source vidéo, **GM-SRC** (pour *Global Model - Source Rate Control*).

2.5 Résultats Expérimentaux

Le format de mise en paquet RTP utilisé dans les expérimentations est décrit dans [Ba98] (cf annexe B). Le MSS a été fixé à 1000 octets et le seuil T_{qual} à 1 dB. La séquence vidéo utilisée est la séquence "News" à 25 Hz.

Différentes valeurs ont été considérées pour le délai *initial* de présence dans les *buffers* (émission+réception) des données. Ce délai global est noté K et s'exprime en nombre d'images mises dans les *buffers* à $t = 0$. Les délais envisagés et étudiés sont des délais faibles (i.e. $K = 2, 3, 4$ images) en phase avec les applications nécessitant de l'interactivité (e.g. visioconférence).

Dans les trois approches étudiées ici (DI-SRC, FB-SRC et GM-SRC), seule la méthode simpliste DI-SRC est particulièrement sensible à la répartition initiale des délais de présence dans les *buffers*. C'est pourquoi, afin que l'étude soit complète, nous avons expérimenté, pour cette approche, deux stratégies de répartition initiale des délais de mise en *buffer*: la méthode que l'on nommera DI-SRC1 correspond à $\sim 80\%$ des données initialement mises en *buffer* ($\sim 80\%$ de K) du côté émetteur et $\sim 20\%$ du côté récepteur. Quant à l'approche DI-SRC2, la répartition des délais initiaux de mise en *buffer* est inverse. Cette dernière répartition, plus réaliste, correspond aux méthodes classiques de pré-chargement de données dans le *buffer* de réception (avec un délai constant) utilisées dans les applications de streaming.

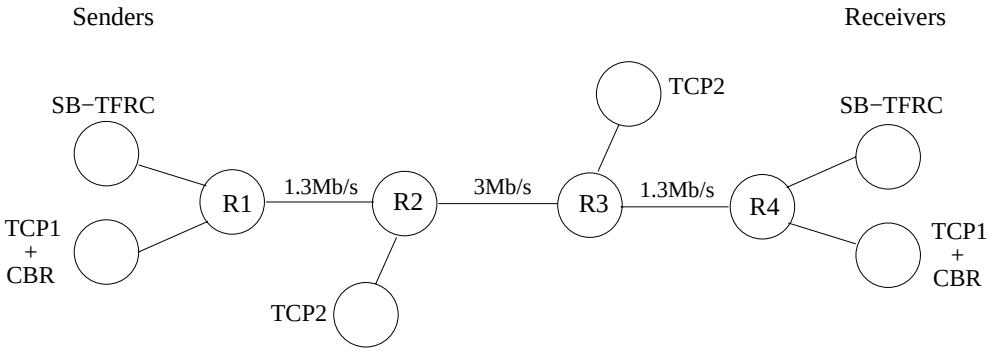

FIG. 2.9 – *Topologie de simulation (linéaire).*

2.5.1 Simulations avec NS2

Le premier ensemble d'expérimentations vise à valider le schéma complet (Estimation de débit TCP-compatible + Modèle global : GM-SRC) avec le simulateur de réseau NS2. Nous avons étudié sa réactivité et son comportement en présence de taux de pertes variables, de gigue et de trafic concurrent. Les résultats reportés résument les observations faites dans les différentes campagnes de tests réalisées.

2.5.1.1 Scénario

Un point important dans la conception de protocoles de congestion réside dans leur réactivité face aux changements de conditions du réseau. Ce comportement a été analysé avec la topologie linéaire montré sur la figure 2.9, comprenant quatre routeurs classiques (*drop-tail*) avec des files d'attente de 15 Koctets. Nous avons choisi des valeurs assez larges afin d'engendrer une gigue assez importante. Cette topologie est composée de liens (R1-R2 et R3-R4) ayant un goulot d'étranglement de 1.3 Mbits/s et d'un lien (R2-R3) avec un goulot d'étranglement de 3 Mbits/s. Le délai de bout-en-bout a été fixé à 40 ms. De plus, différentes combinaisons de trafics concurrents (TCP en "continu" et UDP à débit constant) ont été utilisées.

A t=0, la source vidéo contrôlée par l'algorithme PSA-TFRC est activée pour la durée totale de l'expérimentation. Après 20 secondes, la connexion TCP (TCP1) est activée. Au même instant, une autre connexion TCP (TCP2) est établie entre les routeurs R2 et R3. Ces deux connexions TCP resteront ouvertes jusqu'à la fin. Enfin, afin d'étudier la réactivité du mécanisme face à différents taux de pertes (effectifs), une tranmission UDP à débit constant (CBR: *Constant Bit Rate*) à 1 Mbit/s commence à l'instant t=50 secondes et s'interrompt à t=100 secondes. Un autre flux CBR à 1.1 Mbit/s est ensuite transmis à partir de t=150 secondes jusqu'à la fin de la session.

2.5.1.2 Résultats

Les tables (2.1-2.3) donnent les résultats obtenus, en termes de pourcentages de retards et d'utilisation de bande passante, pour les trois méthodes de régulation de débit (DI-SRC2, FB-SRC, GM-SRC avec ajustement dynamique du δ_{buff}),en considérant des délais initiaux de mise en *buffer* de $K = 2, 3, 4$ images. Le pourcentage d'utilisation de bande passante est calculé comme le ratio entre la quantité de données envoyées (en octets) et la bande passante prédite. La figure 2.10 montre quant à elle la bande passante prédite, la contrainte de débit d'encodage, le taux d'évènement de congestion et le délai de bout-en-bout pour les trois méthodes, considérant le cas $K = 3$ images.

Dans les trois approches, la bande passante est prédite avec le protocole (PSA-TFRC). On peut clairement voir que le protocole réagit rapidement aux changements d'états du réseau. On observe également que la valeur de la bande passante prédite est en étroite corrélation avec l'approche de régulation. Ainsi, l'approche FB-SRC souffre d'un nombre important d'évènement de congestion à l'instant t=150 s. Les sous-remplissages périodiques du *buffer* d'émission, observés avec les méthodes DI-SRC2 et FB-SRC, mènent à une transmission non continue et sporadique. Quand l'émetteur arrête de transmettre des données, les connexions TCP deviennent plus agresives et engendrent, par conséquent, de fortes variations dans les conditions de transmission de la session entière. On observe également dans la table (2.1) que le nombre de retards est très important avec la méthode DI-SRC2. Ces retards s'expliquent par le fait que cette approche ne peut résister ni à une forte gigue, ni à des variations importantes (e.g. forte décroissance) de bande passante. En présence de tels phénomènes, les données restent beaucoup trop longtemps dans le *buffer* d'émission et celui-ci a du mal à se vider. L'approche GM-SRC, quant à elle, fournit un comportement plus stable en terme de transmission (i.e. transmission continue).

Le meilleur compromis entre retard, nombre de pertes et utilisation de la bande passante est clairement obtenu avec la méthode que nous avons proposée : GM-SRC.

	DI-SRC2		
K en nb d'images	**2**	**3**	**4**
% de retards	28.76	28.32	15.20
% bande passante	99.50	99.39	99.68

TAB. 2.1 – *Pourcentage de retards et d'utilisation de bande passante pour l'approche DI-SRC2.*

	FB-SRC		
K en nb d'images	**2**	**3**	**4**
% de retards	0.086	0.08	0.030
% bande passante	96.01	95.37	94.98

TAB. 2.2 – *Pourcentage de retards et d'utilisation de bande passante pour l'approche FB-SRC.*

	GM-SRC		
K en nb d'images	**2**	**3**	**4**
% de retards	0.079	0.08	0.031
% bande passante	99.36	99.69	99.58

TAB. 2.3 – *Pourcentage de retards et d'utilisation de bande passante pour l'approche GM-SRC.*

2.5.2 Résultats sur l'Internet réel

Dans cette partie, nous avons voulu expérimenter les différentes approches sur le réseau Internet afin de voir son comportement en conditions réelles. Nous avons, de plus, intégré ici le codeur vidéo et donnons des résultats en terme de qualité objective mesurée par le PSNR.

2.5.2.1 Scénarios

De très nombreuses expérimentations ont été réalisées entre l'INRIA Rennes (France) et le laboratoire R.U.S (Rechenzentrum der Universität Stuttgart) (Allemagne) et entre l'INRIA Rennes et l'INRIA Sophia-antipolis (France), sur l'Internet, sous différentes conditions de délais, de gigue et de pertes. Afin d'accentuer la congestion potentielle, différents trafics concurrents provenant, suivant les expériences, de l'ENST Bretagne Rennes (France), de l'INRIA Rennes, de l'INRIA Sophia-antipolis ou de Stuttgart ont été activés. Nous avons utilisé un codeur vidéo H.263+ implémentant la méthode de régulation de débit du TMN8 (adaptée aux débits variables). Nous avons activé également les options *slice structured mode* et *Intra forced updating*, couplées avec des mécanismes de dissimulation d'erreurs (*loss concealment*). La séquence vidéo utilisée ici est la séquence "News" (CIF, 25 Hz).

Pour chaque configuration, plus de 50 expérimentations (à différentes heures de la journée) ont été menées afin d'obtenir diverses réalisations de canal.

2.5.2.2 Résultats

Les tables (2.4-2.8) donnent les résultats correspondants à ces expériences en termes de pourcentage de retards, d'utilisation de bande passante ainsi que de PSNR moyen obtenus avec les trois solutions de régulation de débit (i.e. DI-SRC, FB-SRC et GM-SRC). Le pourcentage d'utilisation de bande passante est calculé comme le ratio entre la quantité de données envoyées (en octets) et la bande passante prédite.

On peut assez naturellement voir qu'avec l'approche DI-SRC1 (table (2.4)) le nombre de retards est très élevé. Les performances sont améliorées en utilisant une répartition initiale des délais de mise en *buffer* différente (orientée récepteur). On voit, en effet, qu'avec l'approche DI-SRC2 (table (2.5)) le nombre de retards diminue de manière très importante. Cependant, ceux-ci restent encore très important, notamment en présence d'une forte gigue et de variations importantes de bande passante. Enfin, les retards

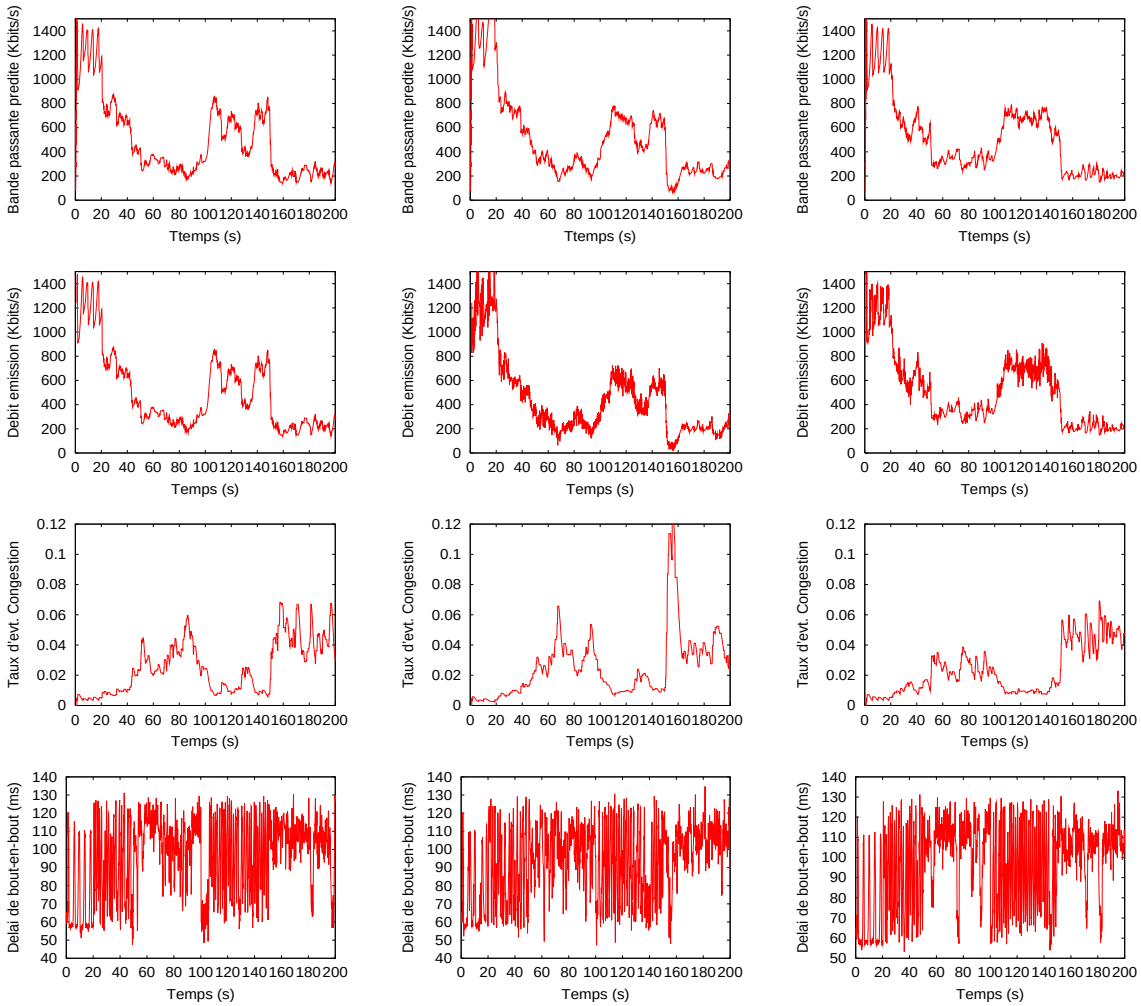

FIG. 2.10 – *Bande passante prédite, contrainte de débit de l'encodeur, taux d'évènement de congestion et délai de bout-en-bout, en considérant un délai initial de présence dans les buffers de $K = 3$ images, pour les algorithmes (a) DI-SRC2, (b) FB-SRC, (c) GM-SRC.*

intervenant en rafales, leur impact sur la qualité visuelle du signal reconstruit est très négatif, comme le montrent les PSNR moyens obtenus.

L'approche FB-SRC (table (2.6)) permet de réduire de manière significative le nombre de retards. Toutefois, cette réduction se fait au prix d'une sous-utilisation de la bande passante et d'une grande instabilité des contraintes de débit renvoyées à l'encodeur. Cette instabilité mène, de plus, à des sous-remplissages périodiques du *buffer* d'émission générant, en conséquence, une transmission sporadique. Enfin, la qualité pâtit de ces comportements.

Dans la suite, nous avons voulu montrer l'intérêt d'utiliser l'approche GM-SRC. De plus, nous montrons également l'intérêt de rafraîchir de manière dynamique la valeur du paramètre δ_{buff}. Pour cela nous avons implémenté la méthode en intégrant un système de mise à jour dynamique et une version fonctionnant à δ_{buff} constant. La table (2.7) fournit les résultats obtenus en utilisant la méthode GM-SRC en considérant une valeur constante pour le délai δ_{buff}. Le nombre de retards obtenus est réduit de manière significative par rapport aux approches DI-SRC. Cependant, le nombre de retards restant relativement important, l'utilisation du mécanisme de mise à jour dynamique du δ_{buff} (cf table (2.8)), permet encore de les réduire. De plus, à l'inverse de FB-SRC le comportement du processus de transmission n'est pas chaotique et surtout la source exploite de manière quasi-optimale la bande passante disponible du réseau.

Les comparaisons des variations des débits d'émission et des taux d'évènements de congestion sont illustrées sur les figures 2.13 et 2.14 pour les trois approches (i.e. les cinq méthodes décrites ci-dessus). Ces courbes sont issues d'échantillons représentatifs des expériences menées en considérant une valeur de $K = 3$ images. Les pourcentages de retards sont calculés sur des intervalles consécutifs d'une seconde. La dernière courbe représente quant à elle, les variations, au cours du temps, du PSNR du signal reconstruit. On retrouve dans ces courbes les comportements décrits plus haut. Ainsi, pour une utilisation comparable de la bande TCP-compatible prédite, les PSNR obtenus pour la vidéo reconstruite avec la méthode GM-SRC sont supérieurs et restent plus stable.

	DI-SRC1					
	INRIA-Sophia			**RUS-Stuttgart**		
K en nb d'images	**2**	**3**	**4**	**2**	**3**	**4**
% de retards	5.86	10.84	22.20	3.08	12.31	31.82
% bande passante	99.40	99.47	99.48	98.96	99.24	99.43
PSNR moyen	35.27	34.08	31.59	36.51	33.99	28.80

TAB. 2.4 – *Pourcentage de retards et d'utilisation de bande passante ainsi que PSNR moyen obtenus entre Rennes et Sophia-Antipolis ($\delta_{feed} = 80ms$) et entre Rennes et Stuttgart ($\delta_{feed} = 120ms$) avec l'approche DI-SRC1.*

La figure 2.11 montre l'impact des retards observés entre les images 400 et 500

	DI-SRC2					
	INRIA-Sophia			RUS-Stuttgart		
K en nb d'images	**2**	**3**	**4**	**2**	**3**	**4**
% de retards	3.98	3.7	2.7	3.40	2.8	2.4
% bande passante	99.31	99.19	99.28	99.42	99.27	99.18
PSNR moyen	35.97	36.07	36.10	36.11	36.22	36.16

TAB. 2.5 – *Pourcentage de retards et d'utilisation de bande passante ainsi que PSNR moyen obtenus entre Rennes et Sophia-Antipolis ($\delta_{feed} = 80ms$) et entre Rennes et Stuttgart ($\delta_{feed} = 120ms$) avec l'approche DI-SRC2.*

	FB-SRC					
	INRIA-Sophia			RUS-Stuttgart		
K en nb d'images	**2**	**3**	**4**	**2**	**3**	**4**
% de retards	0.11	0.08	0.024	0.02	0.05	0.01
% bande passante	95.11	95.30	94.45	97.15	96.21	97.19
PSNR moyen	36.03	36.22	36.42	35.95	36.24	36.32

TAB. 2.6 – *Idem avec l'approche FB-SRC.*

	GM-SRC (constant)					
	INRIA-Sophia			RUS-Stuttgart		
K en nb d'images	**2**	**3**	**4**	**2**	**3**	**4**
% de retards	0.11	0.13	0.15	0.28	0.13	0.31
% bande passante	99.08	99.18	99.38	99.36	99.35	99.36
PSNR moyen	37.03	37.04	37.04	36.97	37.03	37.01

TAB. 2.7 – *Idem avec l'approche GM-SRC (δ_{buff} constant).*

	GM-SRC (dynamique)					
	INRIA-Sophia			RUS-Stuttgart		
K en nb d'images	**2**	**3**	**4**	**2**	**3**	**4**
% de retards	0.031	0.032	0.090	0.049	0.024	0.094
% bande passante	99.08	99.10	99.25	99.34	99.39	99.46
PSNR moyen	37.41	37.45	37.49	37.41	37.34	37.28

TAB. 2.8 – *Idem avec l'approche GM-SRC (δ_{buff} dynamique).*

sur la qualité de la vidéo reconstruite avec l'approche DI-SRC avec un délai de *pré-bufferisation* de $k = 4$ images. Les problèmes induits par les rafales de retards entre ces deux instants sont tout d'abord dûs aux mécanismes de dissimulation de pertes (*Loss Concealment*) entre les images 400 et 450. Ces rafales engendrent ainsi un phénomène de gel de l'image. Le second type de problèmes résulte des propagations de pertes dans la dimension temporelle. La figure 2.12 montre les résultats obtenus dans les mêmes conditions pour l'approche proposée GM-SRC. L'impact visuel des retards sur la vidéo est très réduit (voire nul). De plus, la qualité reste relativement stable durant toute la transmission vidéo.

FIG. 2.11 – *Différentes images clés (entre 400 et 500) de la séquence "News" reconstruites pendant une phase critique de retards avec l'algorithme DI-SRC ($K = 4$).*

FIG. 2.12 – *Différentes images clés(entre 400 et 500) de la séquence "News" reconstruites pendant une phase critique de retards avec l'algorithme GM-SRC avec* δ_{buff} *dynamique* ($K = 4$).

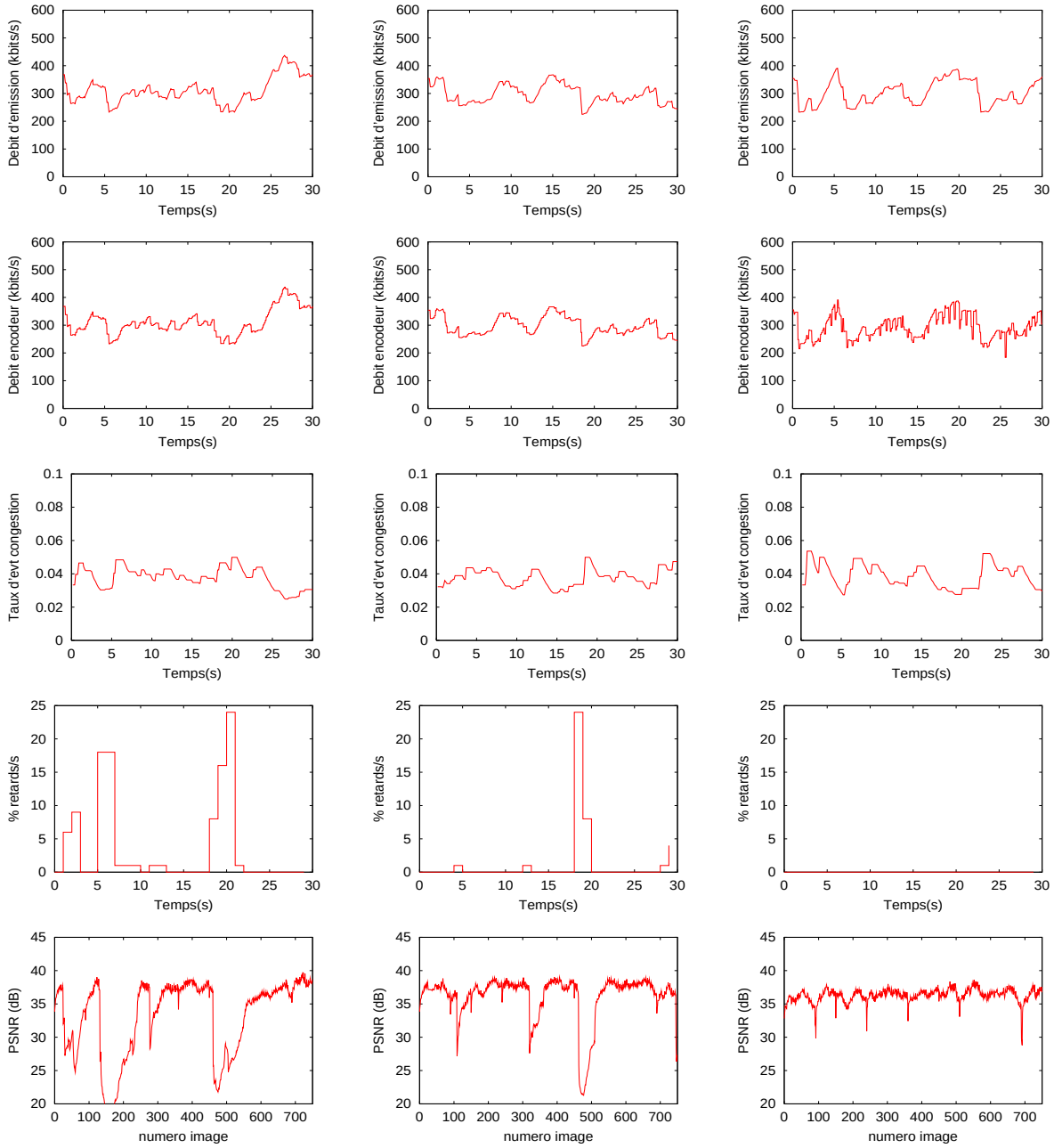

FIG. 2.13 – *Bande passante prédite, contrainte de débit de l'encodeur, taux d'évènement de congestion, pourcentage de retards et PSNR, en considérant un délai de mise en buffer initial de $K = 3$ images entre Rennes et Stuttgart, pour les algorithmes (a) DI-SRC1, (b) DI-SRC2, (c) FB-SRC.*

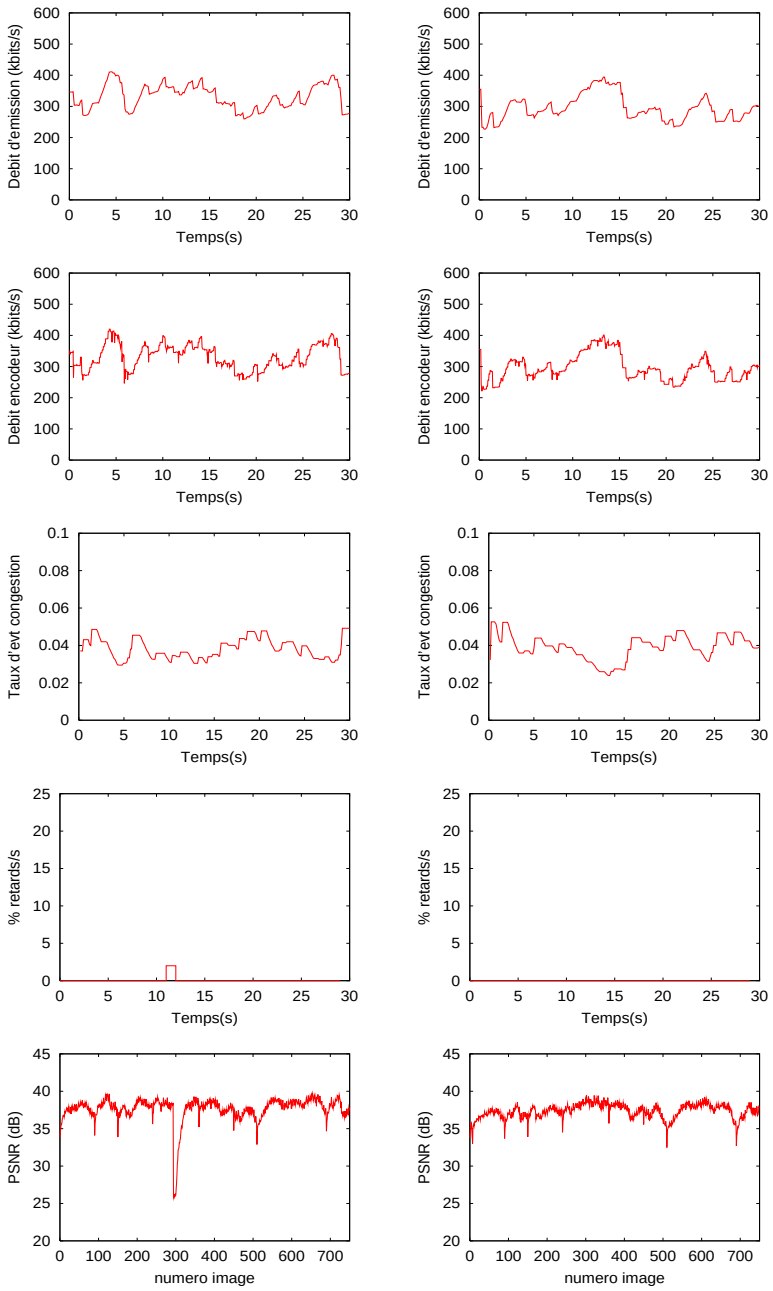

FIG. 2.14 – *Bande passante prédite, contrainte de débit de l'encodeur, taux d'évènement de congestion, pourcentage de retards et PSNR, en considérant un délai de mise en buffer initial de $K = 3$ images entre Rennes et Stuttgart, pour l'algorithme GM-SRC avec (a) δ_{buff} statique, (b) δ_{buff} dynamique.*

2.6 Conclusion

Les communications Internet doivent traditionnellement faire face aux problèmes de congestion du réseau. En ce qui concerne les transmissions multimédia temps-réel, la congestion peut être gérée à l'aide de méthodes complémentaires : contrôle de congestion et régulation de débit. Ces dernières ont pour but de minimiser la quantité de pertes de paquets en accordant le débit de la source (vidéo ou audio) à la bande passante du réseau.

Nous avons proposé ici un nouveau protocole de contrôle de congestion TCP-compatible basé sur RTP prenant en compte, à la différence des protocoles classiques, les caractéristiques des flux multimédia (paquets de tailles variables, délais,...) transportés. Un modèle global de régulation de débit considérant les modèles de *buffer* de la source ainsi que les contraintes de délais de bout-en-bout de flux temps-réels, pour la transmission de vidéo sur l'Internet, est également proposé. Le modèle a été validé par un très grand nombre d'expérimentations réalisées sur l'Internet. Celui-ci permet de réduire de manière significative le nombre de retards et, par conséquent, d'améliorer la qualité du signal décodé, tout en maximisant l'utilisation de la bande passante disponible. Les performances ont encore pu être améliorées en introduisant une nouvelle stratégie d'adaptation de la fréquence temporelle de la source (réduction du nombre d'images codées par seconde) permettant d'obtenir un meilleur compromis entre fréquence temporelle et PSNR, et ceci même en présence de contraintes de débits très variables. Les performances obtenues, en considérant de faibles délais de mise en *buffer*, sont comparables aux quelques approches proposées pour des délais plus de dix fois supérieurs.

Bien que cet algorithme ait été validé avec un encodeur vidéo H.263+, l'approche peut s'appliquer à différents types de codec vidéo incluant les codeurs scalables à grain fin (FGS).

Chapitre 3

Un nouveau protocole TCP-compatible de transmission vidéo multipoint en couches

3.1 Introduction

Dans le chapitre précédent nous avons proposé un schéma global de contrôle de congestion adapté à la transmission de flux multimédia sur Internet. Toutefois, ce type d'algorithme [VG03] bien adapté à la régulation de débit dans le cas d'une source monocouche et en mode de transmission point-à-point, ne peut être transcrit directement à la transmission multicouche multipoint. La situation est nettement plus complexe en multicast et, comme on a pu le voir dans le chapitre (1.4), les différentes solutions proposées dans la littérature ne sont pas pleinement satisfaisantes. Il faut noter, de plus, que les caractéristiques de la source et la qualité perçue par chaque récepteur sont rarement prises en compte dans les mécanismes de régulation de débit.

Le schéma proposé dans ce chapitre est un nouvel algorithme de contrôle de débit multicast hybride orienté émetteur-récepteur prenant en compte les caractéristiques débit-distorsion de la source. Nous faisons l'hypothèse ici que l'émetteur, afin d'adapter ses paramètres de transmission à l'état du réseau, n'a pas besoin de l'ensemble des rapports de chaque récepteur des groupes multicast. Il a plutôt besoin d'un partitionnement des récepteurs en classes homogènes. Chaque couche de la source pourra ainsi par exemple être adaptée à une classe ou un groupe de classes. Chaque classe représente un groupe de récepteurs quasi-homogènes en fonction de variables discriminantes relatives à la qualité du signal reçu (bande passante, taux de pertes,...). La stratégie que nous proposons s'appuie donc sur un mécanisme d'agrégation des rapports des récepteurs dans les noeuds du réseau. Toutefois, à la différence du protocole SAMM [VAS00b], aucune optimisation ne sera réalisée dans les noeuds. Ces derniers ont juste un rôle d'agrégateurs et de mise en forme de l'information. L'autre point clé de notre méthode est que, considérant un nombre limité de niveaux de scalabilité pouvant être générés par la source, les débits source des différents niveaux de scalabilité sont choisis

parmi l'ensemble des débits demandés de manière à maximiser le PSNR vu par chacun des récepteurs. L'approche développée ici permet également pour une couche donnée de déterminer, en fonction de l'état du réseau, la répartition optimale entre données source et données canal (FEC :*Forward Error Correction*). Les données canal permettent en effet de réduire l'impact des pertes de paquets [WZ98, BFPT99, Sal99].

Ce nouvel algorithme de contrôle de congestion multicast est, de plus, couplé à un système de transmission vidéo FGS (*Fine Granular Scalability*) multicouche afin de permettre une adaptation plus aisée du nombre de couches, de leurs débits et de leurs éventuels niveaux de protection.

3.2 Protocole proposé : vue générale

Cette section donne une vue d'ensemble du protocole de contrôle de débit que nous proposons. Sa conception s'appuie sur une structure arborescente (figure 3.1), où les récepteurs sont organisés en arbres hiérarchiques, et où les noeuds internes (i.e. les agrégateurs) agrègent les rapports émis par les récepteurs.

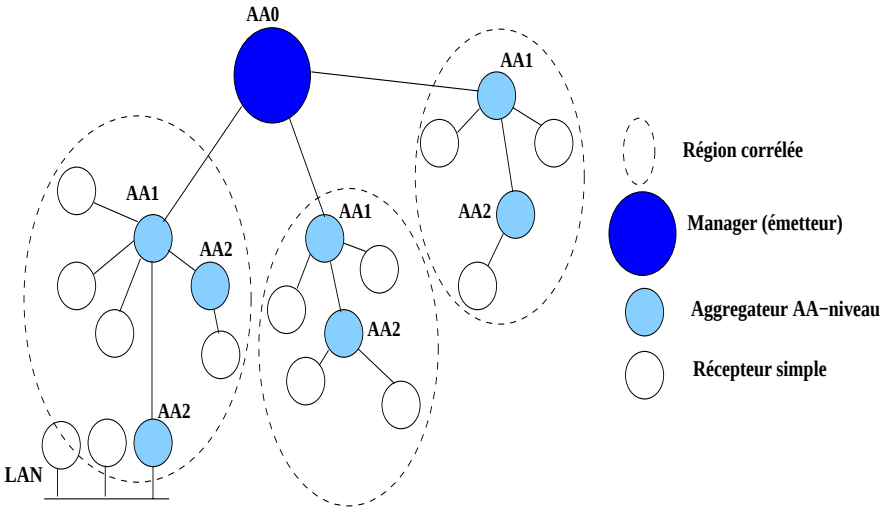

FIG. 3.1 – *Hiérarchie d'agrégateurs multiniveaux.*

Au début d'une session, l'émetteur communique son intervalle de débits $[R_{min}, R_{max}]$, estimé à partir des performances débit-distorsion moyennes de la source. La valeur R_{min} correspond à une valeur de débit en-dessous de laquelle la qualité reçue serait inacceptable, et R_{max} au débit au-dessus duquel l'augmentation de qualité devient insignifiante. Cette information est transmise aux récepteurs au début de la session ou lors de l'arrivée d'un nouveau participant (avec le protocole SDP: *Session Description Protocol* par exemple).

L'intervalle est alors découpé en M sous intervalles, et les valeurs quantifiées résultantes

sont les suivantes:

$$R_i = R_{min} + \frac{i-1}{M}(R_{max} - R_{min}) \ , \ i \in [1, M]. \tag{3.1}$$

Après cette initialisation, le processus de régulation de débit multicast multicouche peut commencer. Ce dernier considère que le temps est divisé en *rounds* de réception. Un round est composé de quatre étapes:

- Au début de chaque round, la source annonce le nombre de couches ainsi que leurs débits, via des rapports RTCP (SR: *Sender Reports*). Chaque couche de la source sera transmise sur un groupe IP multicast différent.

- A la réception de cette annonce, chaque récepteur mesure certains paramètres réseau et estime la bande passante qu'il peut recevoir. Cette bande passante estimée lui sert, ensuite, à prendre une décision quant à son abonnement/désabonnement des couches émises par la source. Différents paramètres réseaux discriminants (bande passante, taux de pertes,...) sont ensuite envoyés à l'agrégateur le plus proche via des rapports RTCP augmentés (RR : *Receiver Reports*).

- Les agents d'agrégation placés à des positions stratégiques au sein du réseau classifient ensuite les récepteurs en fonction de leurs comportements de réception (i.e. variables discriminantes), c'est-à-dire en fonction d'une mesure de distance entre les paramètres renvoyés dans les rapports. Sur la base de cette classification, qui s'apparente à une quantification vectorielle, les agents procèdent à l'agrégation des rapports et des paramètres de réception, fournissant ainsi une représentation de classes homogènes.

- A la réception du dernier rapport agrégé, la source opère alors une adaptation dynamique du nombre de couches à émettre et de leurs débits afin de maximiser la qualité perçue par les différentes classes de récepteurs.

Les sections suivantes décrivent en détails chacune des quatre étapes du protocole proposé.

3.3 Fonctionnalités des récepteurs

Des *caractéristiques discriminantes* doivent être définies afin de pouvoir regrouper des récepteurs. On pourrait ainsi rechercher les corrélations suivant des caractéristiques comme le taux de pertes, les délais, le débit de réception, la bande passante, etc... Parmi ces paramètres, nous avons retenu la paire de variables discriminantes : bande passante, taux de pertes. Ceux-ci sont des paramètres clairement pertinents non seulement au niveau réseau mais également concernant la qualité vidéo.

Deux algorithmes d'estimation de bande passante ont donc été considérés ici: la première approche mesure ce que l'on nommera le *goodput*, et la seconde approche estime la bande passante TCP-compatible du lien.

Cette section décrit les fonctionnalités supportées par le récepteur afin de mesurer ces paramètres, ainsi que la politique d'abonnement/désabonnement retenue. Les champs dédiés, rajoutés dans les rapports RTCP RR, sont également décrits.

3.3.1 Estimation de débit basée "Goodput"

Une notion de **goodput** a été exploitée dans l'algorithme SAMM décrit dans [AVS99]. Basé sur de la différenciation de services entre les différentes couches, le *goodput* est défini dans [AVS99] comme le débit cumulé des couches reçues sans perte. Si une couche souffre d'une perte, celle-ci n'est pas prise en compte dans l'estimation. Le problème d'une telle mesure réside dans le fait que la bande passante prédite est directement dépendante du débit d'émission et par conséquent, ne permet pas une estimation précise de la capacité du lien. Lorsqu'aucune perte n'intervient, afin d'approcher la capacité du lien, l'algorithme SAMM considère que le *goodput* est légèrement supérieur à la valeur mesurée. Cependant, un taux de pertes de 0% n'est pas réaliste sur l'Internet. Les expériences montrent que cette notion de *goodput* ainsi définie, sur un réseau fournissant un service "au mieux" comme l'Internet, mène à une estimation de bande passante décroissant vers zéro durant la session.

Nous avons donc légèrement modifié la définition. Le *goodput* est plutôt défini comme le débit global reçu par le récepteur. Basé sur cette définition nous avons développé un mécanisme simple afin d'estimer au mieux la bande passante disponible du lien sous-jacent. Ainsi, lorsque le taux de pertes est inférieur à un seuil donné T_{loss}, la bande passante B_t estimée à l'instant t se calcule de la façon suivante :

$$B_t = min(B_{t-1} + \Delta, S_{rate} + T_{rate}) \tag{3.2}$$

où B_{t-1} représente la dernière valeur estimée et $\Delta = \alpha \times MSS/RTT$ avec $\alpha \in]0, 1]$. S_{rate} représente le débit auquel le récepteur souscrit (débit cumulé des couches). T_{rate} est un seuil choisi afin de limiter l'augmentation entre deux requêtes, il est égal à un multiple d'intervalles de débit défini dans la section 3.2 (i.e. $T_{rate} = K * (R_{max} - R_{min})/M$ avec $K \in N$ une constante fixée).

A l'inverse, lorsque le taux de pertes devient supérieur à T_{loss}, B_t est mis à g_t, avec g_t le *goodput* observé à l'instant t.

3.3.2 Estimation de la bande passante TCP-compatible

La seconde stratégie considérée, afin d'estimer la bande passante disponible sur le lien, s'appuie sur le modèle analytique de débit TCP proposé dans [PFTK98], adapté et utilisé dans le chapitre précédent (2.1). Il faut cependant noter que l'utilisation de ce modèle dans le cadre de transmission multicast n'est pas directe.

3.3.2.1 Modèle de débit TCP

Nous redonnons brièvement le modèle :

$$B = \frac{MSS}{RTT\sqrt{\frac{2p}{3}} + T_o 3\sqrt{\frac{3p}{8}}p(1 + 32p^2)}, \tag{3.3}$$

avec MSS la taille maximum d'un paquet (cf. chapitre précédent). RTT représente le temps d'aller-retour entre l'émetteur et le récepteur, et p le taux d'évènement de congestion. Le terme T_o représente le délai avant retransmission de l'algorithme TCP (délai après lequel un paquet non acquitté est considéré comme perdu).

3.3.2.2 Estimation des paramètres

Afin de pouvoir utiliser le modèle analytique 3.3, il est nécessaire que chaque récepteur estime le RTT entre lui et la source. Cependant, si chaque récepteur fait le calcul tel que décrit dans la section B.2 en envoyant un RR en mode point-à-point, la voie de retour implosera. C'est pourquoi, nous avons défini un nouveau paquet RTCP que nous avons appelé Probe-RTT. Afin d'éviter une implosion de la voie de retour, seuls les agrégateurs feuilles sont autorisés à envoyer des paquets Probe-RTT à la source. En effet, ceux-ci estiment leur RTT en utilisant l'algorithme proposé dans MLDA [SW00b]. Dans le cas où les récepteurs ne sont pas localisés sur le même réseau local (LAN: *Local Area Network*) que leur agrégateur feuille, ceux-ci doivent ajouter leur propre RTT entre eux et leur agrégateur. Cette estimation peut facilement être faite localement sans générer de trafic indésirable. La source envoie périodiquement des rapports contenant le RTT mesuré (en ms) pour le dernier paquet Probe-RTT reçu avec les SSRCs associés. Ainsi chaque récepteur peut mettre à jour son estimation de RTT en utilisant les résultats relatifs à son agrégateur feuille (dont il connait l'adresse et le SSRC).

L'estimation du taux d'évènement de congestion p est réalisé avec la nouvelle méthode définie dans la section 2.3.3.3 et le paramètre T_o est estimé comme dans la section 2.3.3.2.

Le taux de pertes de paquets réel p_{real} est également calculé sur une fenêtre temporelle glissante.

3.3.2.3 Récepteurs non représentatifs

Dans un environnement très hétérogène, en présence d'un nombre contraint de classes de récepteurs, le débit reçu par certains récepteurs peut fortement différer de leurs requêtes et donc des valeurs de bandes passantes TCP-compatibles estimées. Ces récepteurs "non pleinement satisfaits" sont dits **non représentatifs** et sont contraints (momentanément ou non) à recevoir significativement moins de données que ce qu'ils ont estimé pouvoir recevoir. Un problème non traité (mais présent) dans la littérature peut alors survenir pour ces récepteurs. En effet, dans un tel cas les pertes observées s'avèreront très faibles et par conséquent le taux d'évènement de congestion sera également très (trop) faible. Cela mènera à une sur-estimation de la bande passante et donc à des tentatives d'abonnement infructueuses. Cette non-représentativité du taux d'évènement de congestion conduit à une forte instabilité au niveau de ce type de récepteurs, et nuisent à la stabilité globale de la session.

Afin de pallier ce type de problème, nous estimons donc la bande passante B_t par

$$B_t = min(B, max(S_{rate} + T_{rate}, B_{t-1})) \tag{3.4}$$

où S_{rate} est le débit souscrit et T_{rate} un seuil choisi afin de limiter l'augmentation entre deux requêtes (c.f. section 3.3.1). B_{t-1} est la dernière requête. Quand l'estimation de bande passante B n'est pas fiable, l'historique de pertes servant à estimer le taux d'évènement de congestion est ré-initialisé en utilisant la méthode décrite dans la section 2.3.3.3. Nous verrons dans les expérimentations que l'algorithme reste aussi réactif aux changements d'état du réseau.

3.3.2.4 Phase d'initialisation : Slow-start

La phase d'initialisation adoptée ici diffère du mécanisme dit de *slow-start* décrit dans [FHPW00] et [WH01]. Au début de la session ou lorsqu'un nouveau récepteur rejoint l'arbre de transmission multicast, le débit initialement requis est mis à R_{min}. Puis, après avoir obtenu une première estimation du RTT, p sera estimé et T pourra être calculé par la formule (3.3). La requête résultante B_t^{slow} est alors donnée par

$$B_t^{slow} = max(B, g_t + K \times MSS/RTT) \tag{3.5}$$

avec g_t le valeur de *goodput* à l'instant t et K la même constante que celle utilisée dans la section 3.3.1. L'estimation donnée par (3.5) est utilisée jusqu'à l'observation de la première perte. Après la première perte, la fenêtre d'historique est ré-initialisée en prenant g_t comme bande passante disponible et l'on utilise alors l'équation (3.4).

3.3.3 Politique d'abonnement/désabonnement

Chaque récepteur estime la bande passante B_t disponible sur son lien et décide de s'abonner ou de se désabonner en conséquence. Soient $\{r_1, \ldots, r_L\}$ l'ensemble des débits alloués aux L couches transmises. Le récepteur, qui dispose de ce tableau, s'abonne aux l^* premières couches parmi les L flux transmis, où l^* est tel que:

$$l^* = \arg\min_l \left| B_t - \sum_{i=1}^{l} r_i \right|. \tag{3.6}$$

Cependant, il est nécessaire que le mécanisme de désabonnement prenne en compte le délai entre l'instant d'émission du RR et l'instant où la source adapte le débit de ses couches en conséquence. Des oscillations indésirables peuvent survenir, si les récepteurs décident de se désabonner dès que le débit courant est annoncé comme supérieur à la bande passante estimée. Il est impératif de laisser assez de temps à la source pour adapter son débit d'émission, et ensuite, de décider, éventuellement, son désabonnement si la requête n'a pas été satisfaite.

C'est pourquoi la source, lors de son annonce donne le numéro de round auquel elle répond. Ainsi, nous avons choisi d'attendre que la source fasse une annonce en réponse au round précédent avant de décider de quitter une couche. Afin de rester réactif, le récepteur se désabonne immédiatement lorsque le taux de pertes observé est supérieur à un seuil acceptable T_{loss} (et ceci quel que soit le numero de round annoncé). Ces mécanismes couplés permettent d'éviter de gâcher de la bande passante avec le trafic du protocole d'abonnement/désabonnement IGMP et stabilise d'avantage la qualité de réception.

3.3.4 Protocole de signalisation

Les informations de réception agrégées (i.e. bande passante estimée, taux de pertes) sont périodiquement envoyées à la source dans des RR augmentés par le mécanisme d'extension des rapports RTCP. Ici nous avons rajouté les champs suivants dans le RR :

EB (Expected Bitrate) : un champ de 16 bits donnant la valeur de la bande passante estimée en a kbit/s.

LR (Loss Rate) : Un champ de 16 bits donnant la valeur estimée du taux de pertes réel.

NB (NumBer client) : un champ de 16 bits donnant le nombre de clients demandant ce débit (i.e. EB). Cette valeur est mise à 1 par les récepteurs classiques.

3.4 Agregation des rapports basée sur un algorithme de clustering distribué

Les transmissions multicast exhibent de fortes corrélations spatiales [YKT96]. Un algorithme de classification peut tirer avantage de ces corrélations afin de regrouper des comportements similaires dans des classes homogènes et donc compresser au maximum l'information contenue dans l'ensemble des paquets d'informations de retour (feedbacks). On peut alors voir l'opération d'agrégation comme une procédure de quantification vectorielle des rapports des récepteurs.

Dans notre schéma, l'arbre multipoint est organisé sous la forme d'une hiérarchie de régions locales (c.f. figure 3.1). Dans chaque région, un agrégateur reçoit les informations de retour (feedback) provenant des récepteurs, calcule des statistiques sur les données reçues et renvoie le résultat en point à point vers un agrégateur de hiérarchie plus élevée (agrégateur père). Un agrégateur final (appelé manager), placé à la racine de l'arbre multipoint collecte tous les rapports agrégés.

Cette architecture nécessite une légère modification par rapport à l'architecture RTP classique. De même que pour PIM-SM (*Protocol Independent Multicast-Sparse Mode*) , les RR ne sont pas envoyés en multicast à la session entière, mais sont envoyés en point-à-point vers l'agrégateur père. Puisque ces rapports sont locaux, ils peuvent être émis plus souvent sans pour autant briser la contrainte de 5% spécifiée dans le standard RTP [SCFJ96].

3.4.1 Organisation des agrégateurs au sein du réseau

Les agents d'agrégation (AAs) doivent être placés stratégiquement au coeur du réseau afin de minimiser le surplus de bande passante des rapports RTCP RRs. Plusieurs méthodes ont été proposées afin d'organiser les récepteurs dans une session multicast afin de réaliser un protocole multicast scalable [LPGLA98]. Nous avons choisi une approche multiniveau hiérarchique telle que celle décrite dans le protocole RMTP (*Reliable Multicast Transport Protocol*) [PSLB97]. Toutefois, dans notre approche il n'y a pas de récepteurs désignés : tous les récepteurs envoient leurs rapports à l'agrégateur qui leur est associé.

La racine de la hiérarchie de l'arbre d'agrégation (appelé le *Manager*) est basée à l'émetteur et reçoit tous les résumés de rapports. Il est recommandé dans [EMD98] de ne pas utiliser un arbre hiérarchique de taille supérieur à 3. Dans notre approche, les rapports agrégés fournissent une classification contenant le nombre de récepteurs et le comportement moyen des différentes classes. Le mécanisme d'agrégation est décrit dans la section 3.4.3.

Dans les expériences reportées ici, les agrégateurs ont été manuellement placés. Cependant, si certaines fonctionnalités additionnelles (proposées dans la littérature) sont permises par les routeurs, différentes approches peuvent être utilisés afin de placer automatiquement les agrégateurs. Par exemple, nous pouvons implémenter la fonction des agrégateurs en utilisant un *custom concast* [CGM+01]. Les adresses *concast* sont l'opposé des adresses multicast, i.e. l'adresse représente un groupe d'émetteurs plutôt qu'un groupe de récepteurs. Ainsi un datagramme *concast* contient une sorte d'adresse multicast de source et une adresse de destination unicast.

Avec un tel schéma, tous les récepteurs envoient leurs RRs en utilisant l'adresse de groupe de source RTCP, et seulement un paquet agrégé est délivré à la source. L'interface de signalisation *custom concast* permet à l'application de fournir au réseau une description de l'algorithme de fusion des rapports.

3.4.2 Mesure de similarité

Deux types de mesure de similarité peuvent être utilisées: une mesure de similarité entre deux rapports x et y ($\delta(x, y)$) et une mesure de similarité entre un rapport x et une **classe** (ensemble de rapport (et donc de récepteurs) regroupés selon des caractéristiques similaires) C ($\delta(x, C_l)$).

La première mesure de similarité peut s'appuyer sur une simple distance L^p (i.e. $\delta(x, y) = \sqrt[p]{\sum_i (x_i - y_i)^p}$) ou sur toute autre mesure définie par l'application.

La méthode la plus simple, pour mesurer la similarité d'un rapport vis-à-vis d'une classe, consiste à choisir un représentant $x\hat{}_{C_l}$ pour chaque classe et d'assigner la distance $\delta(x, x\hat{}_{C_l})$ pour représenter la distance entre le nouveau rapport et la classe. Il est aussi possible de définir la distance par rapport à une classe comme la distance par rapport au point le plus proche ou le plus éloigné de la classe (e.g. $\delta(x, C_l) = \min_{y \in C_l} \delta(x, y)$ avec $\delta(x, C_l) = \max_{y \in C_l} \delta(x, y)$). Nous utiliserons ici un représentant.

Dans le cas de notre application, l'utilisation d'une simple distance L^p, peut mener à des regroupements non adéquats. Ainsi, par exemple, deux récepteurs ayant le même taux de pertes mais ayant requis des débits significativement différents peuvent aisément être regroupés. Dans un tel cas, la représentativité de la classe est clairement mise en doute. Afin d'éviter ce type de problème, nous avons défini une nouvelle notion que l'on nomme **seuil de similarité**. Chaque composante (i.e. taux de pertes, bande passante) possède un seuil de similarité. Ce seuil correspond à la distance L^2 maximale entre deux représentants d'une composante pour que ces représentants soient considérés comme *similaires*.

La distance retenue alors entre deux rapports, deux classes, un rapport et une classe

est donnée par :

$$\delta(x, y) = \max_i \frac{abs(x_i - y_i)}{dt_i} \tag{3.7}$$

avec dt_i le seuil de similarité choisi pour la composante i.

3.4.3 Algorithme de clustering

Chaque classe est représentée par un point et une pondération. Le point représentatif peut être vu comme un vecteur dont les composantes sont données par les caractéristiques discréminantes considérées. Nous utilisons ici un algorithme de clustering au plus proche voisin. Le point représentant un nouveau RR rejoindra la classe qui lui est le plus proche. La métrique utilisée est celle définie dans la section précédente. Le nouveau point modifie le point représentant et remet à jour le facteur de pondération de la classe. Si la distance à toutes les classes existantes est supérieure à 1, une nouvelle classe sera formée. Toutefois, le nombre de classe est limité à N_{max}. Le point représentant est obtenu en utilisant une formule de moyennage pondéré sur tous les points de la classe. L'algorithme complet est donné par :

- $N_{\max}$ =*Nombre maximal de classes*
- $\mathbf{r}$ = *RR reçu*
- Recherche de la classe la plus proche $\delta(\mathbf{r}, \hat{C}_l) = \min_{C_l} \delta(\mathbf{r}, C_l)$
- **si** $(\delta(\mathbf{r}, \hat{C}_l) \geq 1)$ et (Nombre de classe existant $< N_{\max}$)
  - Ajouter un nouveau classe C_{new} et mettre à jour $\hat{C}_l = C_{\text{new}}$
  - Mise à jour du point représentatif de la classe $\hat{C}_l$,
    $\hat{x}_{\hat{C}_l} = \frac{weight(\hat{C}_l)\hat{x}_{\hat{C}_l} + \mathbf{r}}{weight(\hat{C}_l)+1}$
  - Augmenter le poids de la classe $\hat{C}_l$

Au terme de chaque phase de clustering, le manager reçoit un vecteur représentant les caractéristiques de réception de l'ensemble des récepteurs. Notons que cet algorithme permet de classifier les récepteurs suivant leurs caractéristiques courantes mais peut aussi prendre en compte le passé récent des récepteurs. On initialisera donc une phase de classification avec les classes formées durant la phase précédente. Les nouveaux points remettront à jour les classes existantes. Il arrivera au cours d'une session que certaines classes formées dans le passé ne soient plus représentatives d'aucun récepteur. Dans ce cas ces classes pourront être supprimées.

3.5 Régulation de la source vidéo multicouche

Les rapports de canal générés par l'algorithme de classification offrent périodiquement à l'émetteur des informations concernant l'état du réseau. Plus précisément, ce mécanisme délivre un taux de pertes, une bande passante limite et le nombre de récepteur présent

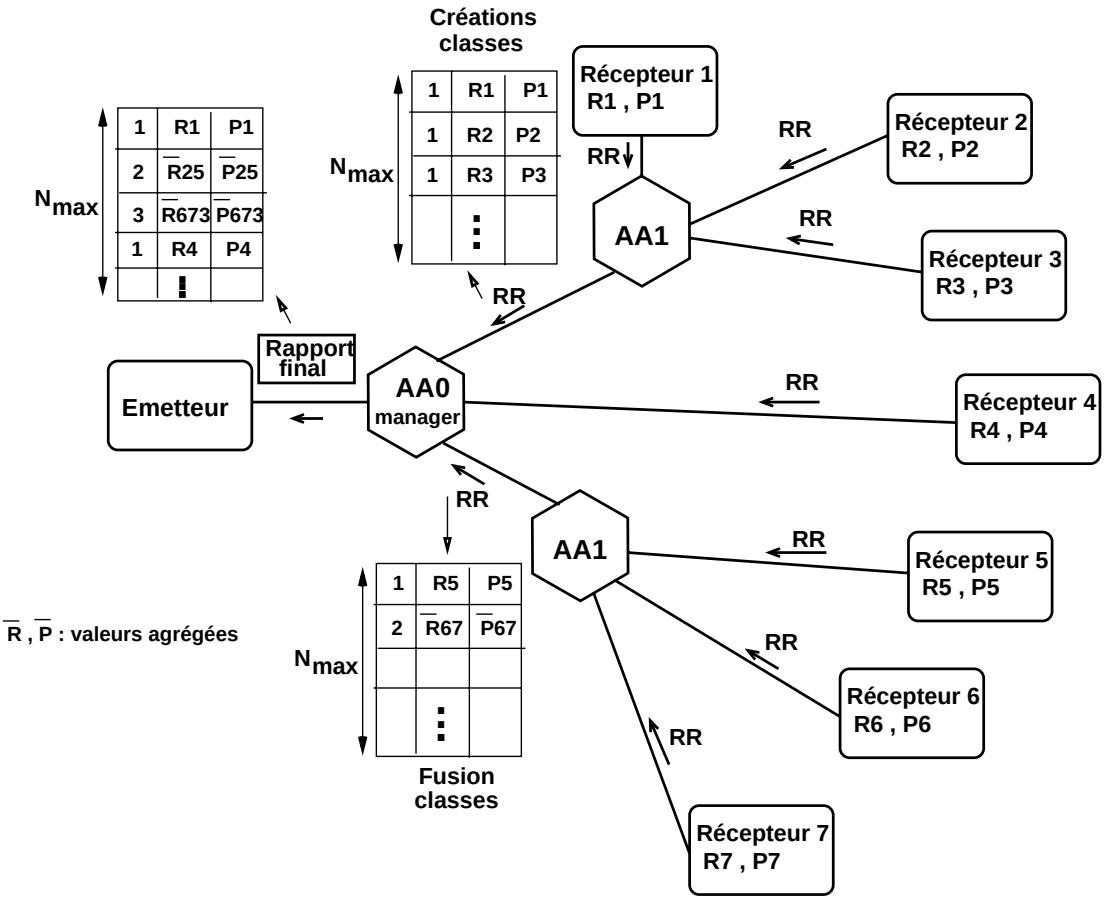

FIG. 3.2 – *Exemple de scénario d'agrégation des informations de retour lors d'une session multipoint.*

pour une classe de récepteurs donnée. Ces informations sont ensuite exploitées afin d'optimiser le nombre de couches, leur débit ainsi que leur niveau de protection.

Cette section décrit, tout d'abord, l'algorithme de régulation de débit données/ protection prenant en compte l'état du réseau et les caractéristiques débit-distorsion de la source. Le système d'encodage de la source vidéo finement scalable (FGS) utilisé ainsi que le serveur de streaming multicast considérés sont ensuite décrits.

3.5.1 Sélection optimale des débits transmis et du taux de protection

Nous considérons ici l'utilisation de mécanismes de protection appelés FEC (*Forward Error Correction*). Dans le contexte de transmission sur l'Internet, la détection d'erreur est généralement fournie par les protocoles des couches basses. Ainsi, les couches hautes ont à gérer principalement les effacements ou les paquets perdus. La position exacte des données manquantes étant connue, une bonne capacité de cor-

rection peut être obtenue en utilisant des codes MDS (*Maximal Distance Separable*) systématiques. Un code MDS (n,k) prend k paquets de données et produit $n-k$ paquets de redondance. La propriété de ces codes permet de résister jusqu'à $n-k$ pertes dans un groupe de n paquets. La probabilité effective de perte $P_{eff}(k)$ d'un code MDS après décodage de canal est donnée par

$$P_{eff}(k) = P_e \sum_{j=0}^{k-1} \left(\begin{array}{c} n-1 \\ j \end{array} \right) P_e^{n-1-j}(1-P_e)^j, \tag{3.8}$$

où P_e est la probabilité moyenne de perte du canal. Une question à résoudre est ensuite, connaisant la probabilité de perte, comment répartir de manière optimale la bande passante disposnible entre données et redondance. Cela revient à trouver le niveau de protection (ou le paramètre k/n) pour chaque couche. L'optimisation conjointe de la répartition de débits entre données et FEC est décrite dans la suite.

3.5.1.1 Formulation du problème

Pour un nombre maximum de couches L supportées par la source, le nombre de couches effectivement envoyées, leurs débits et leurs niveaux de protection sont choisis afin de maximiser le PSNR global vu par les récepteurs. Il faut noter que les débits sont choisis dans un ensemble de N débits requis (informations de réception). Le problème peut être formulé de la façon suivante:

$$(\Omega_1, \ldots, \Omega_l) = \arg \max_{(\Omega_1,\ldots,\Omega_l)} G, \tag{3.9}$$

où $\Omega_i = (r_i, \frac{\kappa_i}{n}), i = 1, \ldots, l$ avec r_i représentant le débit cumulé source et canal, et $\frac{\kappa_i}{n}$ le niveau de protection de chaque couche i.

La mesure de qualité G à maximiser est définie par

$$G = \sum_{j=1}^{N} \left(\sum_{i=1}^{l} \mathrm{PSNR}(\Omega_i).\mathbf{P}_{eff_{j,i}} \right).C_j \tag{3.10}$$

$$\text{avec } l = \arg \max_{k \in [1,\ldots,L]} \left\{ \sum_{i=1}^{k} r_i \leq R_j \right\}. \tag{3.11}$$

Les termes R_j et C_j représentent respectivement le débit requis et le nombre de récepteurs dans la classe j. Le terme $\mathrm{PSNR}(\Omega_i)$ dénote l'augmentation de PSNR associée à la réception de la couche i. Le PSNR de la couche i dépend directement des couches inférieures. Le terme $\mathbf{P}_{eff_{j,i}}$ représente la probabilité, pour les récepteurs de la classe j, que i couches soient correctement décodées et peut s'exprimer par

$$\mathbf{P}_{eff_{j,i}} = \prod_{k=1}^{i} \left(1 - \bar{p}_{eff_{j,k}} \left(\frac{\kappa_k}{n} \right) \right), \tag{3.12}$$

où $\bar{p}_{eff_{j,k}}$ est la probabilité effective de pertes observée par tous les récepteurs de la classe j recevant les k couches considérées.

Les valeurs PSNR(Ω_i) sont obtenues en estimant les performances débit-distorsion $D(R)$ de l'encodeur de source sur un ensemble de séquences d'entrainement. Le modèle peut ensuite être raffiné durant le processus d'encodage de la séquence, si il est réalisé en temps-réel, ou stocké sur le serveur dans le cas d'applications de streaming.

Une procédure d'optimisation doit donc être mise en place, afin de maximiser la fonctionnelle ci-dessus (c.f. Equation (3.10)).

3.5.1.2 Combinaison optimale

A la réception du rapport final agrégé, l'émetteur opère une quantification de ces différentes valeurs sur l'intervalle de valeurs possibles définies dans la section 3.2. Les requêtes sont ensuite triées en ordre croissant. C'est à partir de cet ensemble ($R_1, ..., R_N$ avec $R_1 < R_2 < R_3 .. < R_N$) que la recherche des différentes combinaisons, satisfaisant potentiellement les récepteurs, va avoir lieu. On recherche alors toutes les combinaisons de j éléments $j \in [1, ..., L]$ parmi les N requêtes. Puis, pour une combinaison de débit donnée, les différentes combinaisons de répartition de débit données/FEC sont considérées. En pratique, pour un code MDS (n,k) on ne recherche les k_i de chacune des couches que pour $k_i \in [n/2; n]$ (on rappelle que ici n est fixé). Une illustration de ce processus est proposée sur la figure 3.3.

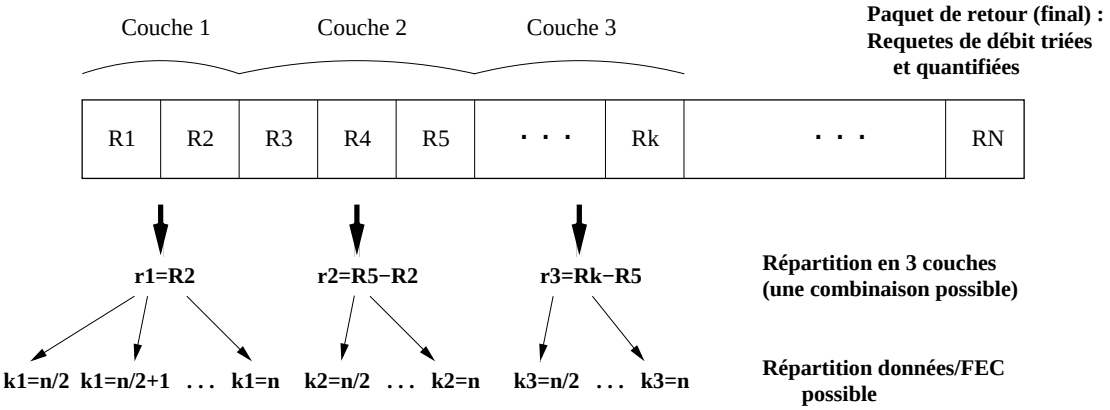

FIG. 3.3 – *Illustration d'une combinaison possible de couches et allocation de débit dans ces couches, à partir des requêtes de débit agrégées, triées et quantifiées. Si la combinaison illustrée ici est retenue, alors 3 couches seront transmises, avec les débits respectifs r_1, r_2 et r_3. Au sein de ses trois couches, il reste à déterminer la proportion entre données et redondance (FEC).*

Dans le cas d'un nombre maximum de couches réduit L, une recherche exhaustive de la combinaison optimale est satisfaisante en terme de complexité. En revanche, pour une plage de débits possibles et un codeur scalable très granulaire, un algorithme d'optimisation accélérée est nécessaire pour garder une complexité raisonnable. Le sous-ensemble

maximisant (3.9) peut ainsi être calculé en utilisant un algorithme de programmation dynamique [Koo77].

3.5.2 Source vidéo scalable à grain fin

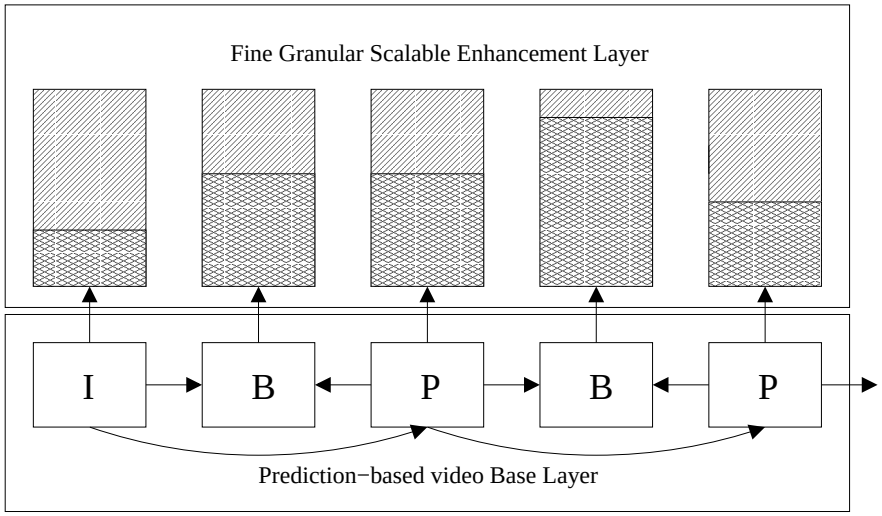

FIG. 3.4 – *Structure du schéma de codage vidéo MPEG4-FGS.*

Les couches sont générées par un codeur MPEG-4 FGS (Fine Granular Scalability)[RC99] (cf section 4.3.2.3). Le codage FGS a été introduit afin de faciliter l'adaptation de débit des sources aux variations de bande passante du réseau dans le cas d'applications de transmission de flux pré-encodés (e.g. vidéo à la demande).

En effet, même si les schémas de codage scalables classiques (i.e. SNR, spatial , temporel) fournissent une réponse au problème d'adaptation de débit à la bande passante du réseau, ces approches souffrent de limitations en termes de granularité d'adaptation. La structure de l'approche FGS est décrite sur la figure 3.4. La couche de base est encodée à un débit noté R_{BL}, en utilisant une approche hybride basée sur une prédiction temporelle compensée en mouvement suivie par un schéma de compression basé sur la DCT. La couche d'amélioration est, quant à elle, encodée de manière progressive jusqu'à un débit maximum R_{EL}. Le train binaire résultant est progressif et peut être tronqué en tous points, au moment de la transmission, afin de s'adapter aux variations de bande passante. La troncature est contrôlée par l'optimisation debit-distorsion décrite plus haut.

L'encodeur compresse le contenu sur l'intervalle de débit $[R_{min} = R_{BL}, R_{max}]$. Ensuite, le même flux compressé peut être utilisé pour des applications unicast ou multicast.

3.5.3 Serveur de streaming vidéo FGS multicast

Les expériences reportées dans ce papier ont été réalisées en considérant un serveur de streaming FGS. La figure 3.5 montre la structure du système de streaming multicast considéré en incluant le contrôleur de débit multicouche et le module de calcul de FEC.

Pour chaque séquence vidéo pré-stockée sur le serveur, il existe deux trains binaires séparés (i.e. un pour la couche basse BL et un pour la couche d'amélioration EL), chacun d'eux étant couplés avec un descripteur. Ces descripteurs contiennent diverses informations concernant la structure des flux. Ainsi, il contient le décalage (en octets) du premier bit de chaque image par rapport au début du train binaire d'une couche donnée. Le descripteur de la couche de base contient également le décalage du débit de chaque "*slice*" (ou "*video packet*") d'une image. Une estampille temporelle (CTS: *Composition TimeStamp*) de chaque image utilisée comme instant de présentation du côté décodeur est également présent dans le descripteur.

A la réception d'une nouvelle liste $(r_0, r_1, \ldots, r_L)$ de contraintes de débit, le contrôleur de débit FGS calcule un nouveau budget par image (pour chaque couche) tenant compte de la fréquence temporelle de la source vidéo. Puis, au moment de la transmission, le contrôleur de débit FGS partitionne la couche d'amélioration FGS en un nombre correspondant de "sous-couches". Chaque nouvelle couche sera envoyée sur un groupe IP multicast différent.

Il est intéressant de noter que quel que soit le nombre de couches d'amélioration FGS auquel le client souscrit, le décodeur ne décode qu'une unique couche (i.e. les "sous-couches" fusionnent du côté du décodeur).

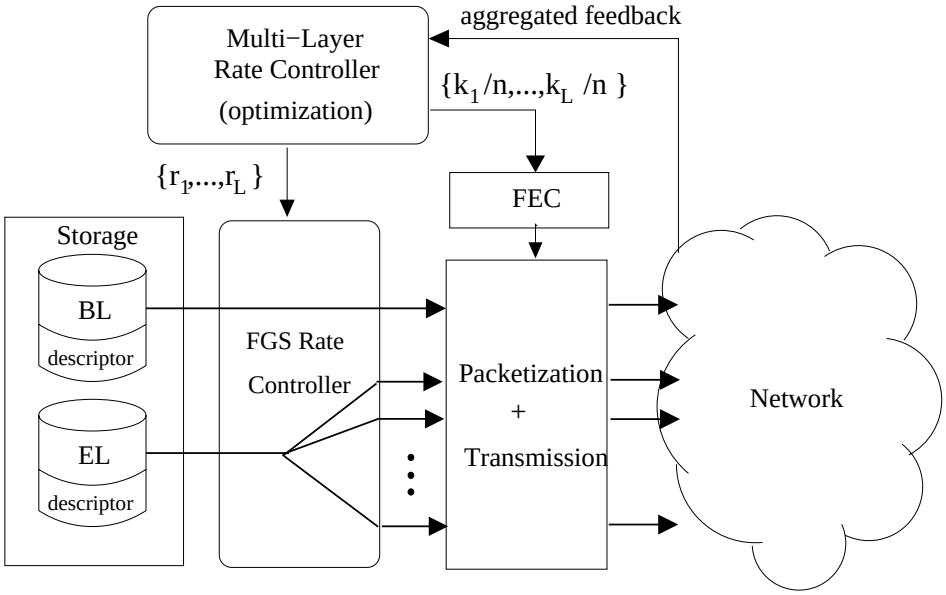

FIG. 3.5 – *Serveur de streaming vidéo FGS multicast.*

3.5.4 Signalisation

En plus du RTT calculé pour les paquets *Probe-RTT*, les rapports d'émission RTCP envoyés périodiquement contiennent des informations au sujet des couches émises i.e., leur nombre, leur débit et leur niveau de protection, selon la syntaxe suivante :

NL (Number Layer) : un champ de 8 bits donnant le nombre de couche d'amélioration.

BL (Base Layer) : un champ de 16 bits donnant le débit de la couche de base.

EL_i (Enhancement Layer i) : un ensemble de champs de 16 bits donnants le débit de la couche d'amélioration i, $i \in 1, \ldots, NL$.

K_i : un ensemble de champs de 8 bits contenant le niveau de protection de la couche i $i \in 0, \ldots, NL$ [1].

3.6 Expérimentations

Dans cette partie, nous évaluons les performances de notre algorithme basé sur une optimisation débit-distorsion globale ont également été testées en considérant les deux méthodes d'estimation de bande passante décrites plus haut : basée *goodput* et TCP-compatible.

Les expérimentations présentées ici ont été réalisées avec le simulateur de réseau $NS2$ [NS2]. Dans un premier temps, nous comparerons le protocole SAMM [VAS00a] à notre approche basée *goodput*. Puis, nous montrerons les limites de l'approche basée *goodput* notamment dans le cadre d'un réseau réel où les flux sont en compétition avec TCP. Enfin, nous montrerons les performances obtenues avec l'approche dite TCP-compatible, ainsi que l'apport des mécanismes de correction d'erreur de type FEC sur notre algorithme. Dans la suite, la méthode basée sur le *goodput* sera nommée **GB-MRC** et la méthode TCP-compatible **TCPF-MRC** (respectivement *Goodput Based* et *TCP Friendly Multicast Rate Control*).

La figure 3.6 représente la topologie principale de simulation que nous avons retenue. Etant donnée la topologie de l'arbre multicast, nous avons considéré ici une représentation de la source en 3 couches, chaque couche étant transmise sur une adresse IP multicast. La couche de base est encodée à un débit constant de $256kb/s$. Le débit global (couche de base plus couche d'amélioration) peut prendre toutes les valeurs comprises entre $256Kbit/s$ et $1Mb/s$. A $t = 0$ chaque client souscrit aux 3 couches émises aux débits initiaux de $R_{BL} = 256kb/s$, $R_{EL1} = 100kb/s$ et $R_{EL2} = 0kb/s$. Les caractéristiques débit-distorsion de la source FGS considérée sont données sur la figure 3.7.

Dans les algorithmes décrits plus haut, les multiples expérimentations menées (non reportées ici) conduisent à un choix de $K = 4$ (plusieurs valeurs de $K \in [1, 7]$ ont été testées). Le seuil T_{loss} est quand à lui fixé à 2%.

1. Nous considérons ici un code de Reed-Solomon de rapport k/n. La valeur n est fixée au début de la session et seul le paramètre k est adapté dynamiquement durant la session. Cependant, nous pouvons également considérer l'adaptation du paramètre n.

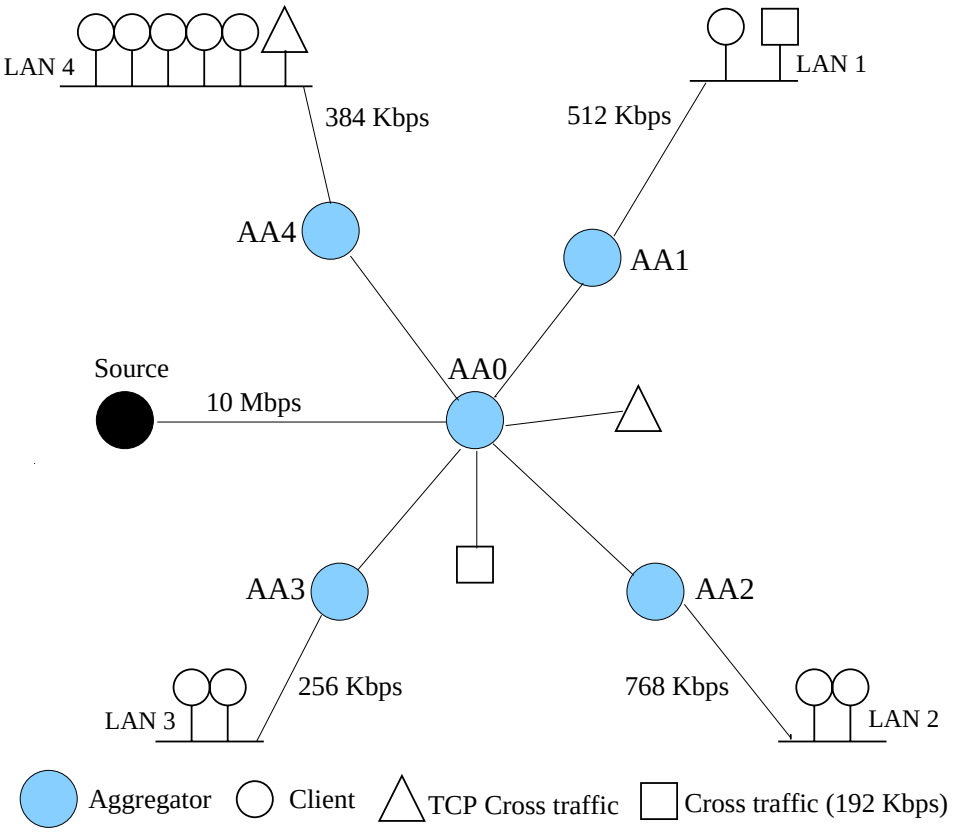

FIG. 3.6 – *Topologie principale de simulation.*

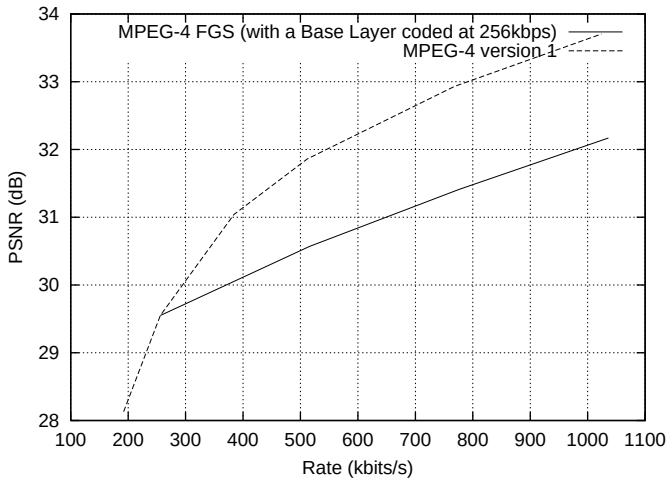

FIG. 3.7 – *Modèle débit-distorsion de la source vidéo FGS (Thomson Multimedia).*

3.6.1 Le protocole SAMM-like versus GB-MRC

Nous étudions ici les comportements des protocoles SAMM (noté ici **SAMM-like**) et GB-MRC dans le cas où aucun trafic concurrent n'est activé (**scénario 0 :aucun trafic concurrent**). L'algorithme SAMM-like est implémenté et prenant comme valeur de *goodput* la quantité globale de données reçues sans perte. De plus, l'incrément choisi, lorsque le taux de pertes est nul est égal à un intervalle de débit de la source. C'est pourquoi, l'algorithme est nommé SAMM-like puisque le véritable algorithme SAMM ne peut fonctionner en l'état. Rien n'a été modifié concernant l'optimisation du *goodput* global.

Les figures 3.8, 3.10 et 3.11 montrent les résultats obtenus avec l'algorithme SAMM-like pour les différents récepteurs. On voit clairement que l'approche SAMM-like ne permet pas de faire une utilisation efficace de la bande passante. En effet, prenons, par exemple, les clients du LAN 2 (avec un lien d'une capacité de 768 kb/s) n'ont pas reçu à plus de 300 kb/s sur ce lien. Des observations similaires peuvent être faites sur les récepteurs des autres LANs. De plus, il faut noter que si aucune borne basse R_{min} n'est fixée, le *goodput* des différents récepteurs converge irrémédiablement vers une valeur très faible. En plus de cette utilisation nettement sous-optimale de la bande passante, l'algorithme engendre un comportement très instable en termes d'abonnements/désabonements aux groupes multicast, provoquant par conséquent une congestion locale inutile.

Les figures 3.9, 3.12 et 3.13 montrent les résultats obtenus avec l'algorithme GB-MRC. Ce dernier permet de satisfaire la majorité des récepteurs et approche au mieux la bande passante disponible des différents liens. De plus, on arrive assez rapidement à une situation stable assurant ainsi, dans cette configuration, un nombre d'abonnements/désabonnements quasi nul. Il faut remarquer le taux de pertes instantané qui est logiquement plus élevé que celui de la méthode SAMM-like.

On voit clairement ici le bénéfice de la solution GB-MRC par rapport à la solution SAMM-like. Ceci souligne l'intérêt d'une mesure globale prenant également en compte les caractéristiques de la source (et en particulier les caractéristiques débit-distorsion) par rapport à une simple optimisation du *goodput* global. De plus, la mise en place d'une politique d'abonnements/désabonnements intelligente est également importante.

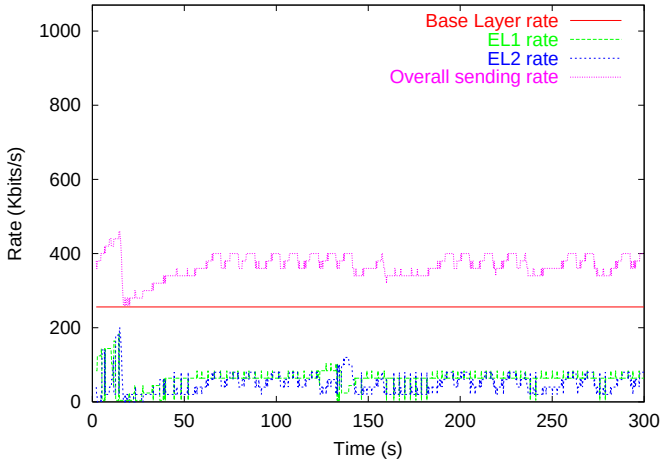

FIG. 3.8 – *scénario 0 : Variations de débit de chaque couche de la source vidéo avec l'approche SAMM-like (sans trafic concurrent).*

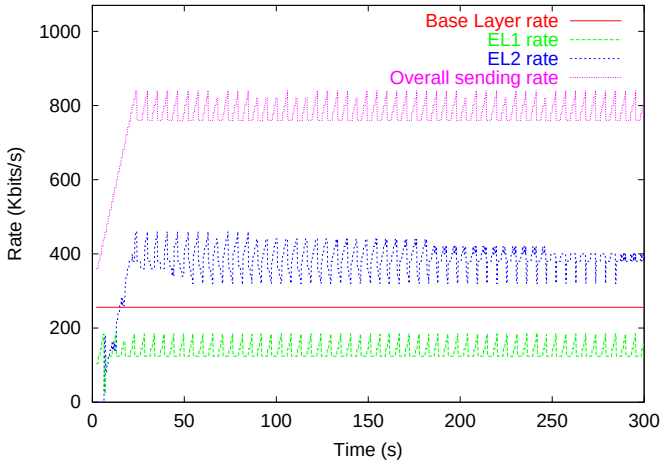

FIG. 3.9 – *scénario 0 :Variations de débit de chaque couche de la source vidéo avec l'approche GB-MRC.*

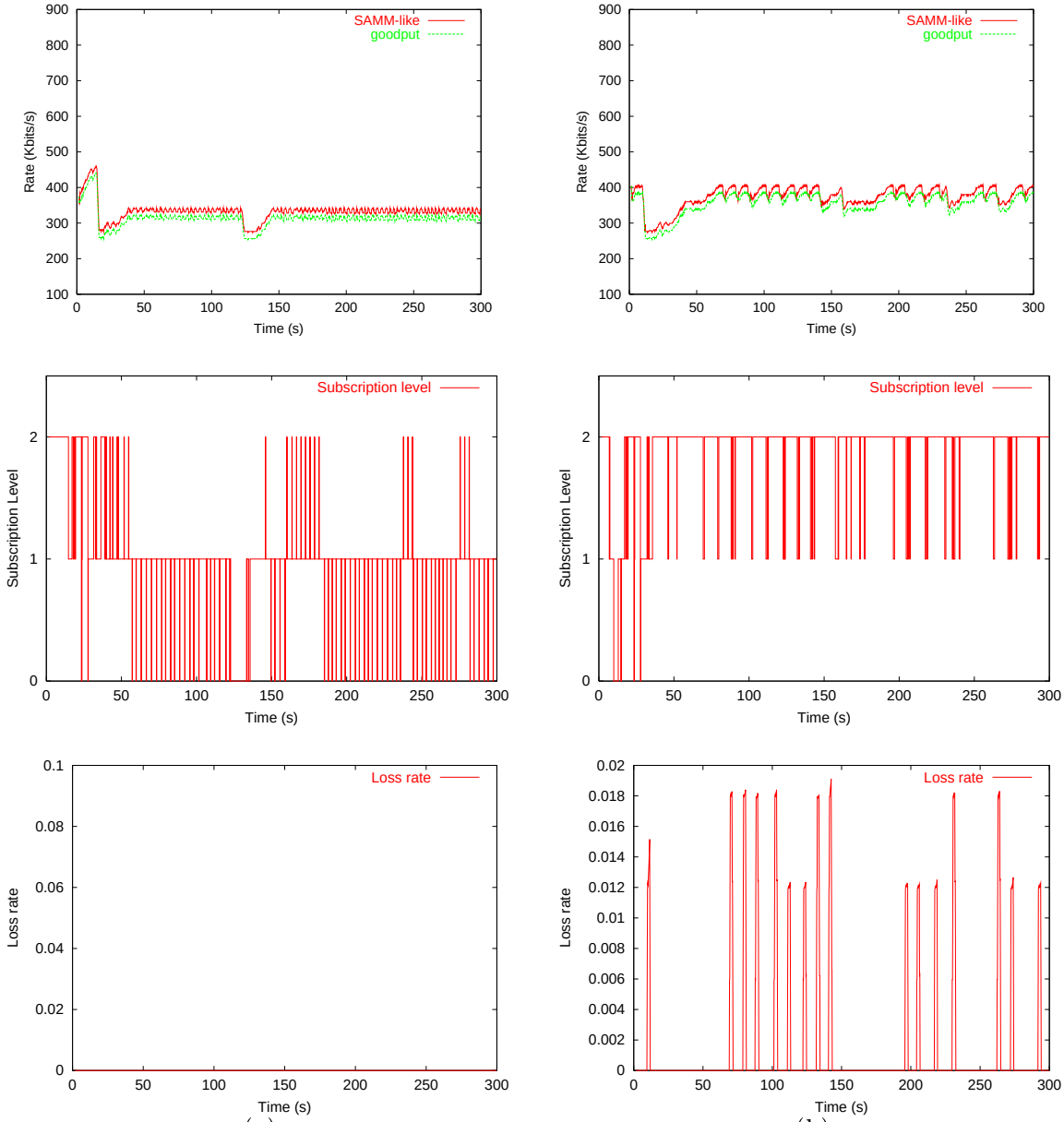

FIG. 3.10 – *scénario 0 : Débit requis avec SAMM-like vs mesure de goodput, niveau de souscription et taux de pertes instantané pour les clients (a) du LAN 2 (lien à 768kb/s) (b) du LAN 4 (lien à 384kb/s).*

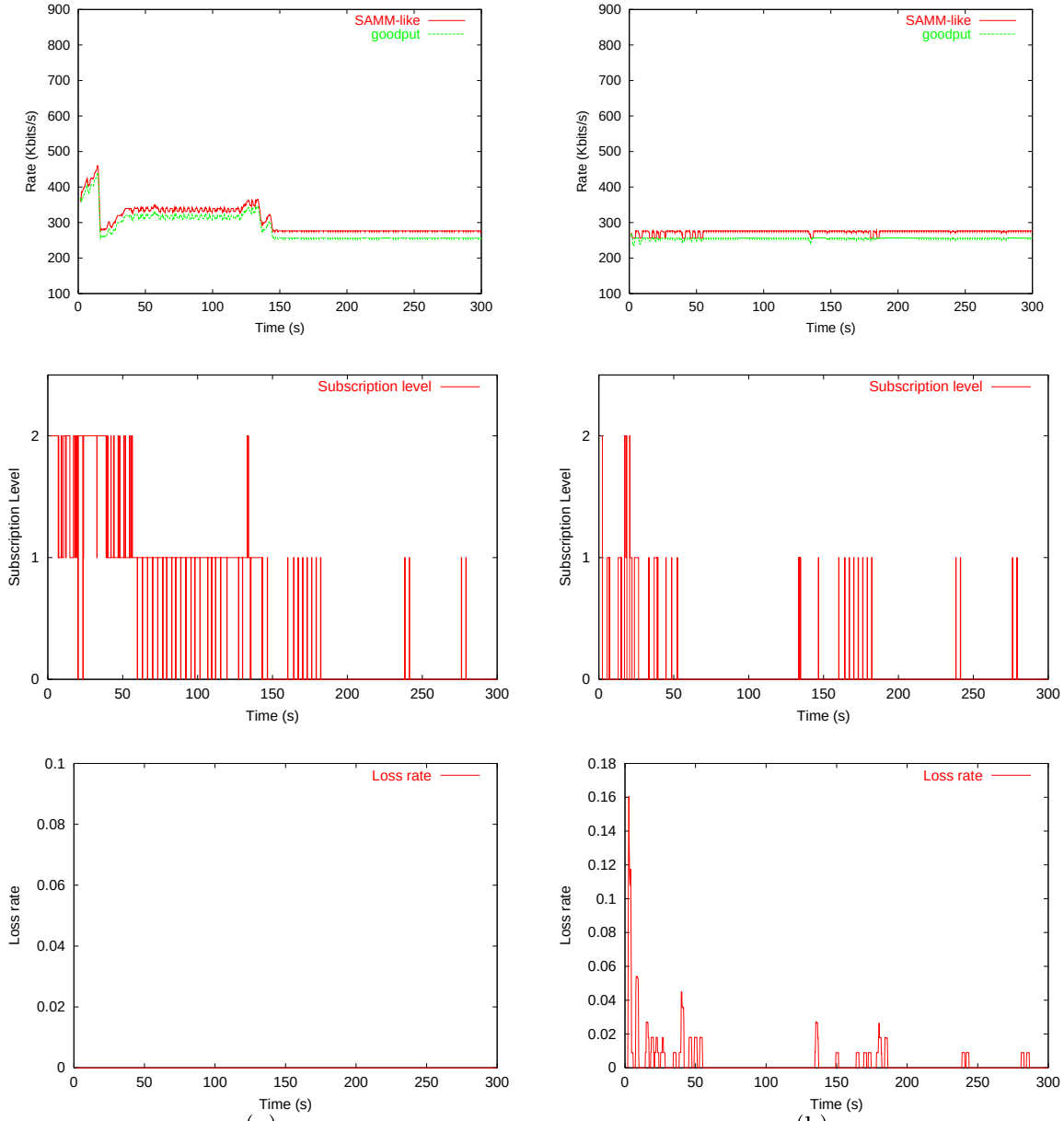

FIG. 3.11 – *scénario 0 : Débit requis avec SAMM-like vs mesure de goodput, niveau de souscription et taux de pertes instantané pour les clients (a) du LAN 1 (lien à 512kb/s) (b) du LAN 3 (lien à 256kb/s).*

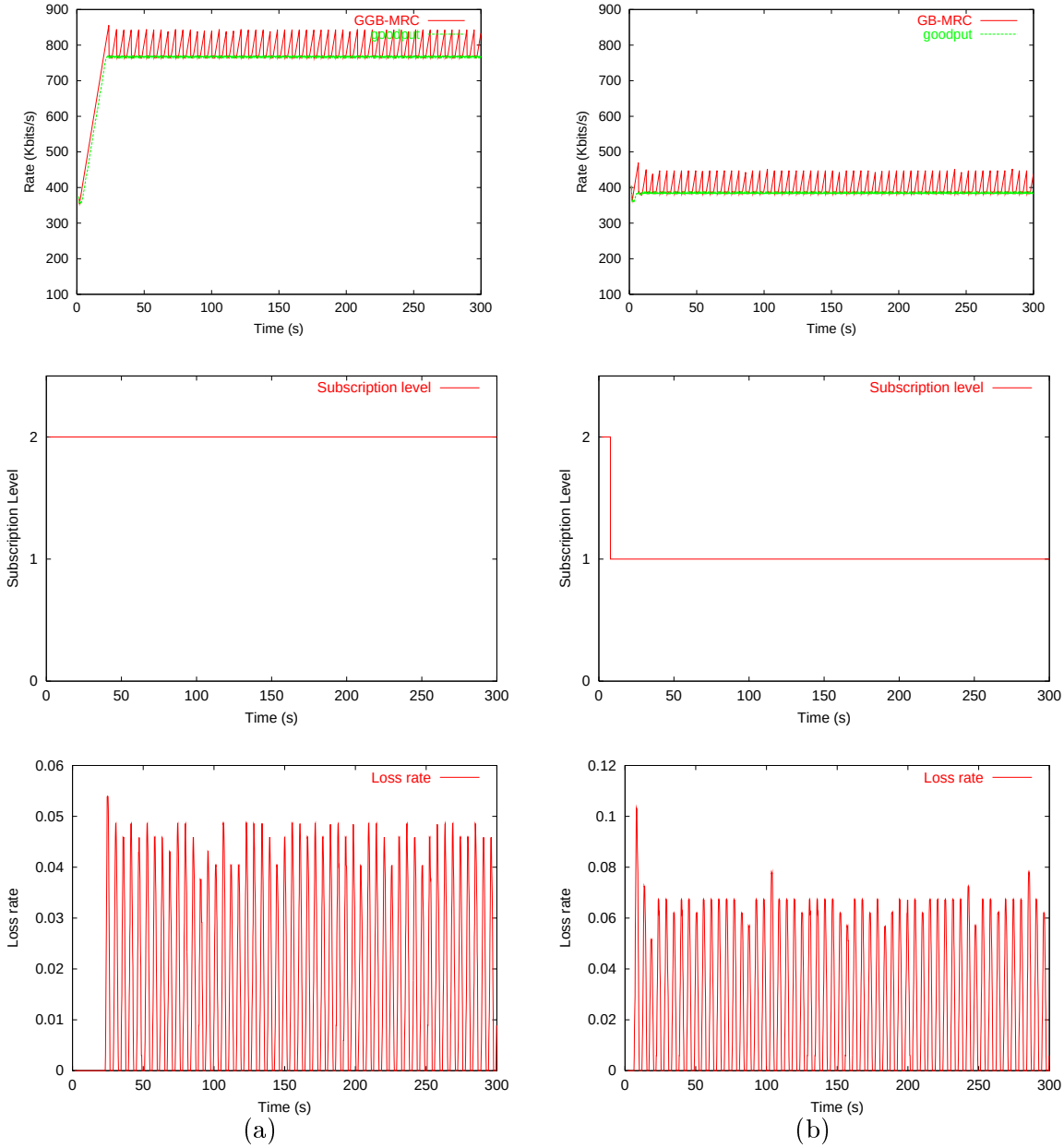

FIG. 3.12 – *scénario 0 : Débit requis par l'approche GB-MRC vs mesure de goodput, niveau de souscription et taux de pertes instantané pour les clients (a) du LAN 2 (lien à 768kb/s) (b) du LAN 4 (lien à 384kb/s).*

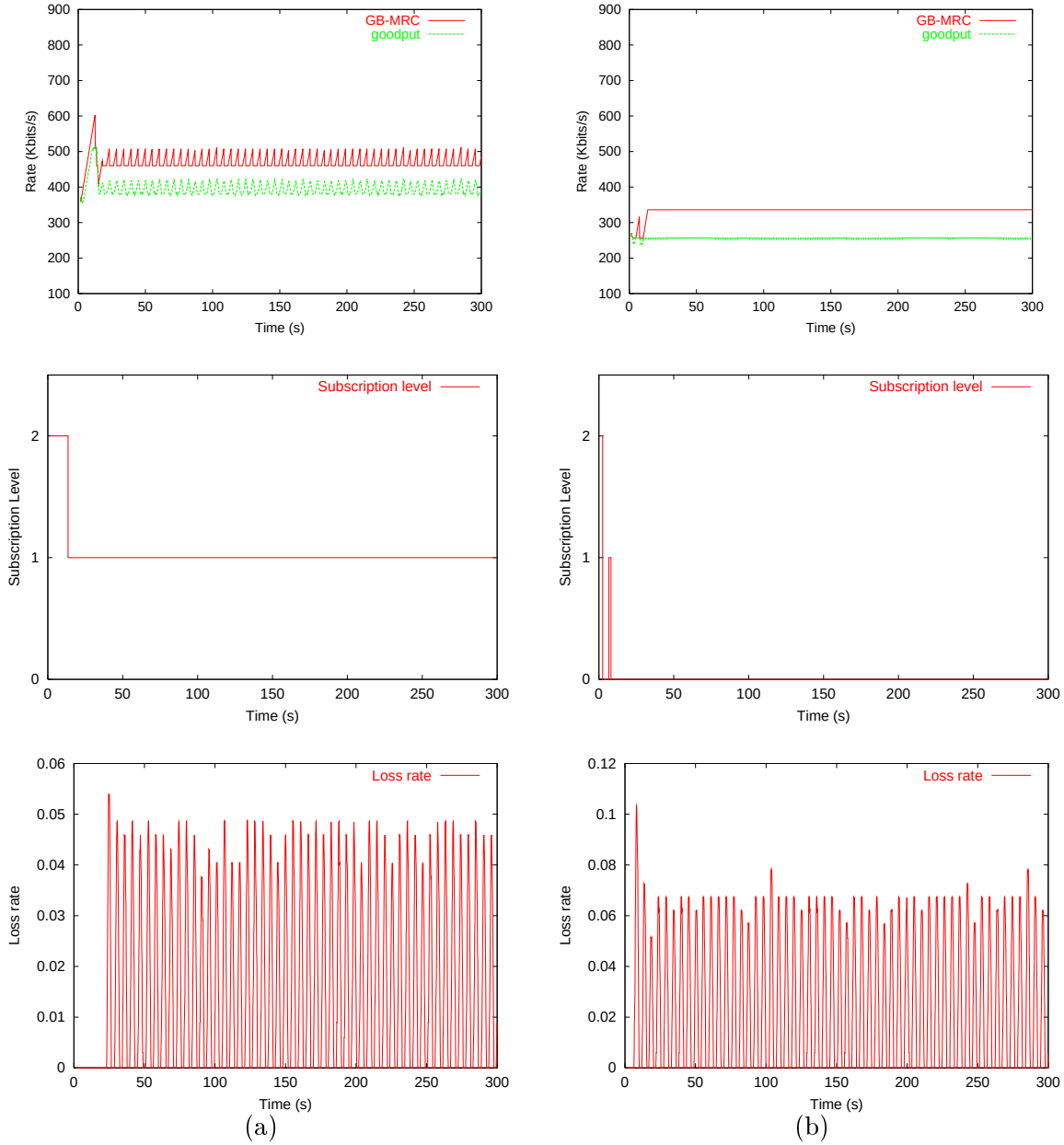

FIG. 3.13 – *scénario 0 : Débit requis par l'approche GB-MRC vs mesure de goodput,niveau de souscription et taux de pertes instantané pour les clients (a) du LAN 1 (lien à 512kb/s) (b) du LAN 3 (lien à 256kb/s).*

3.6.2 Le protocole GB-MRC versus TCPF-MRC

Nous étudions ici les performances en termes de PSNR et de taux de pertes pour les deux mécanismes de régulation de débit GB-MRC et TCPF-MRC. Nous considérons également ici la topologie multicast décrite sur la figure 3.6. La périodicité des rapports de réception sera égale au RTT maximum de l'ensemble des récepteurs. La séquence utilisée dans les expériences reportées ici est la séquence "Brest" fournie par Thomson Multimedia. Elle a une durée de $300s$ (25 Hz, 6700 images).

Durant la session, les flux vidéo seront en compétition avec deux types de trafic concurrents : des flux UDP à un débit constant (CBR) de $192kb/s$ et des flux TCP. Ces flux contribuent à diminuer le goulot d'étranglement du lien. L'activation du trafic concurrent entre les clients représentés par des "carrés" sur la figure 3.6, pendant l'intervalle de temps $[100s, 200s]$, limite ainsi la bande passante du lien correspondant (i.e. des clients du LAN 1) à environ $320kb/s$. De manière similaire, le trafic TCP concurrent activé entre les clients représentés par des "triangles", pendant l'intervalle de temps $[140s, 240s]$, conduit à une limitation du lien (i.e. des clients du LAN 4) à une valeur moyenne de $192kb/s$. Il faut remarquer qu'en présence de trafic concurrent la bande passante disponible sur le lien est inférieure au débit de la couche de base, qui dans le cas de notre application, est maintenu constant en moyenne (e.g. $256Kb/s$). Ce scénario de test sera appelé **scénario 1**.

3.6.2.1 Résultats avec la méthode GB-MRC

La figure 3.14, donne les variations de débit des différentes couches de la source FGS, au cours de la session, obtenues avec l'approche GB-MRC. Les figures (3.16 et 3.17, montrent, quant à elles, les débits requis versus la mesure réelle du *goodput*, le taux de pertes, le nombre de couches reçues et les PSNR obtenus pour les clients des différents LANs. On peut voir que même en présence de trafic concurrent l'algorithme est capable de faire une utilisation quasi-optimale de la bande passante. En effet, les estimations de bande passante suivent parfaitement les variations des différents liens au cours du temps. De plus, le comportement en termes d'abonnements et désabonnements non pertinents reste également intéressant.

Cependant les taux de pertes instantanés observés restent encore assez élévés. De plus, les pertes interviennent de manière sporadique mais régulière. Par conséquent, la qualité de la vidéo reçue souffre de ces taux de pertes et les PSNR obtenus sont relativement perturbés. Enfin, une autre limitation de ce mécanisme est la latence à l'initialisation de la session, il faut au moins 30 secondes avant que l'algorithme converge.

3.6.2.2 Résultats avec la méthode TCPF-MRC

La méthode TCPF-MRC (cf Fig. 3.15, 3.18) et 3.18) permet de pallier les limitations de l'approche GB-MRC tout en conservant les mêmes propriétés. Ainsi, les débits d'émission des différentes couches suivent les variations de bande passante des différents liens. L'algorithme proposé permet par contre d'obtenir une session stable avec des taux

de pertes faibles et un nombre d'abonnements/désabonnements non pertinents limité. La comparaison des courbes de PSNR (figure 3.18) révèlent un gain d'au moins 2 dB pour les récepteurs du LAN 2 par rapport aux récepteurs du LAN 4. Ceci montre clairement l'intérêt d'un mécanisme *hybride émetteur-récepteur* de régulation de débit dans un environnement multicast hétérogène.[2] Enfin, l'algorithme converge en moins de 10 secondes dans le scénario décrit ici grâce au mécanisme de slow-start proposé.

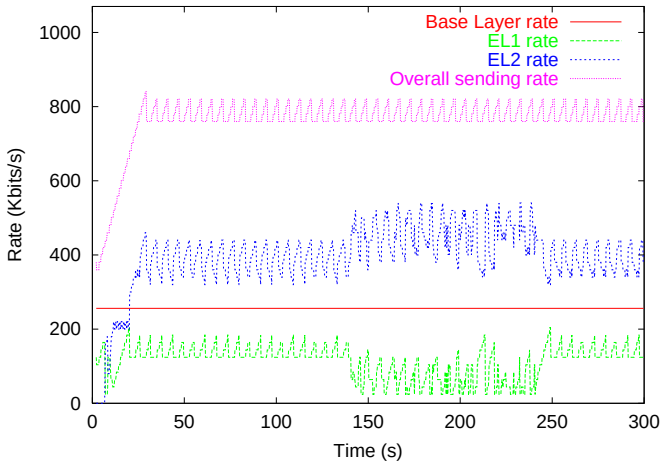

FIG. 3.14 – *scénario 1 : Variations de débit de chaque couche de la source vidéo avec l'approche GB-MRC.*

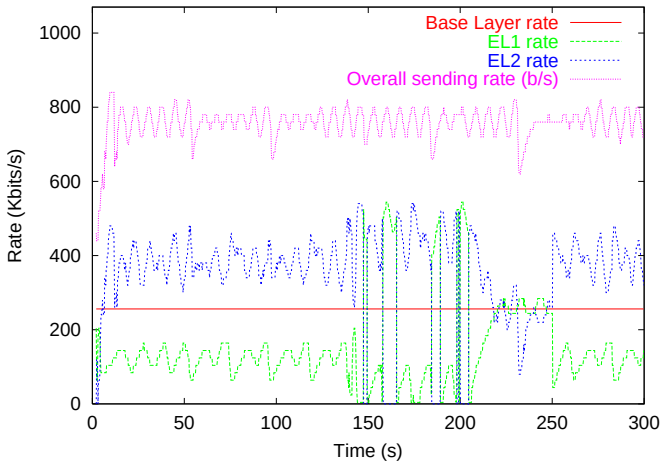

FIG. 3.15 – *scénario 1 : Variations de débit de chaque couche de la source vidéo avec l'approche TCPF-MRC.*

2. Notons que les "pics" de taux de pertes observés pour l'approche TCPF résultent d'une prédiction de débit TCP-compatible pouvant occasionnellement excéder (de peu) la bande passante disponible du lien et ceci dû notamment à une inadéquation entre les paramètres réseaux pour des raisons de délai.

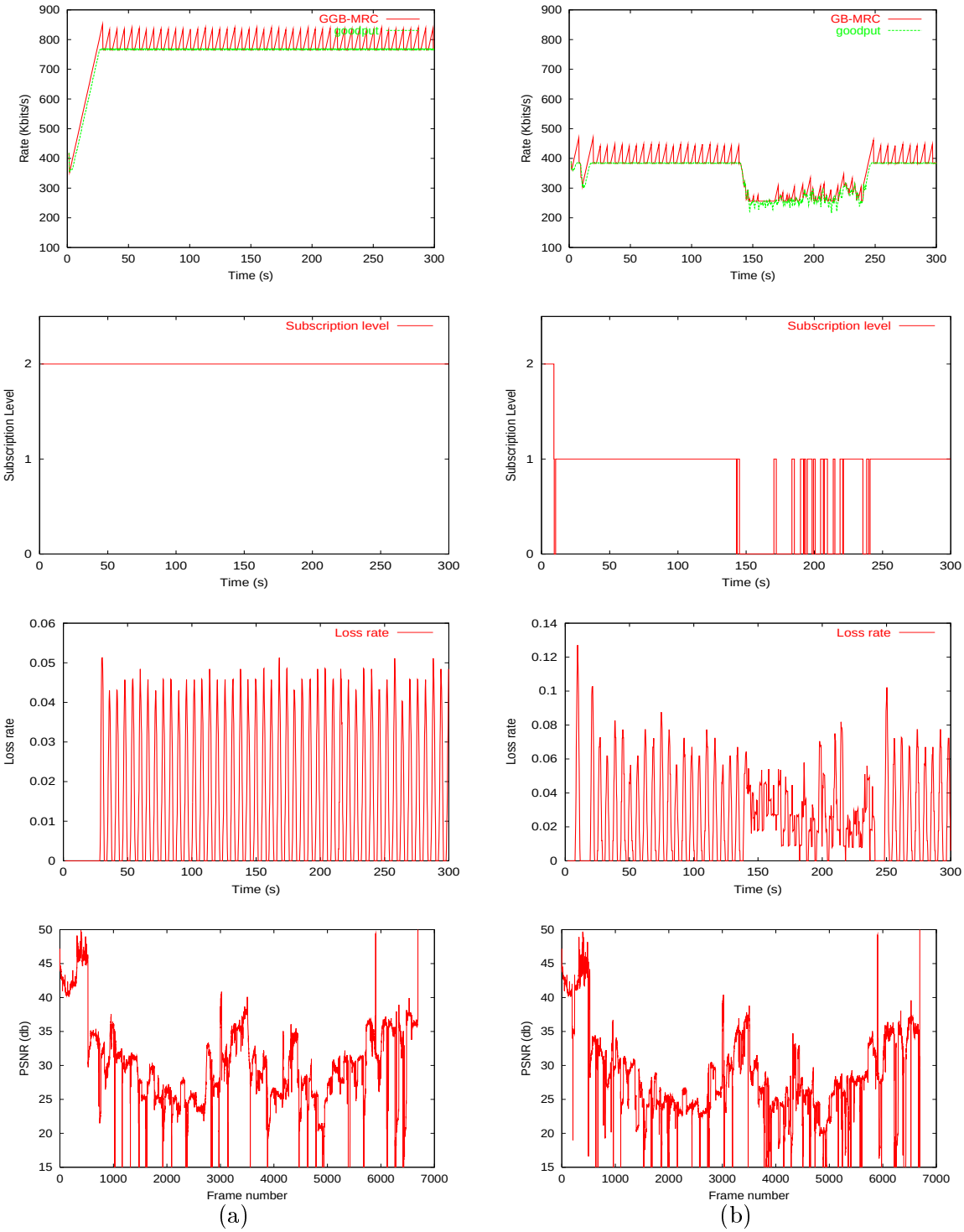

FIG. 3.16 – *scénario 1 : Débit requis par l'approche GB-MRC vs mesure de goodput, taux de pertes et niveau de souscription pour les clients (a) du LAN 2 (lien à 768kb/s) (b) du LAN 4 (lien à 384kb/s).*

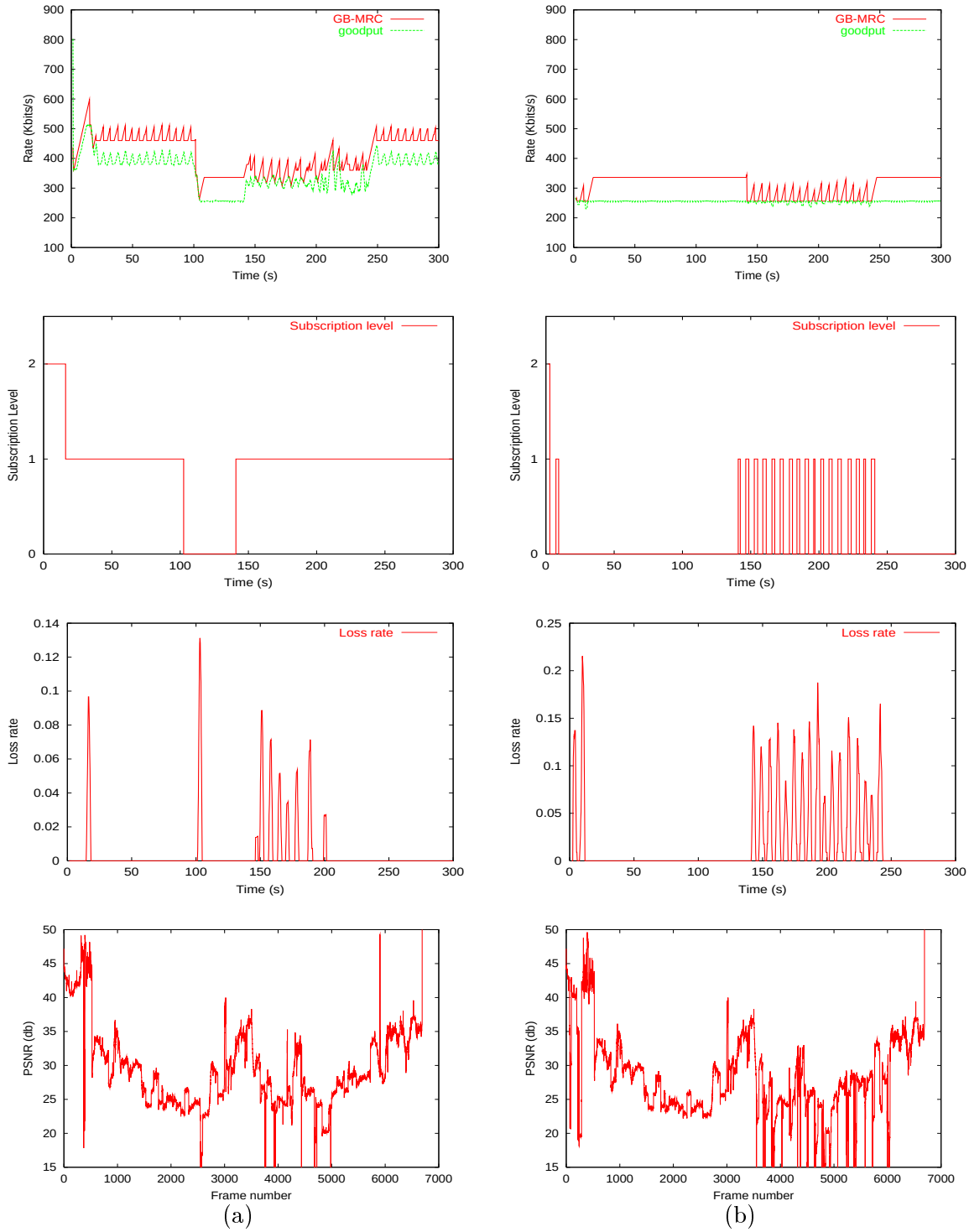

FIG. 3.17 – *scénario 1 : Débit requis par l'approche GB-MRC vs mesure de goodput, taux de pertes et niveau de souscription pour les clients (a) du LAN 1 (lien à 512kb/s) (b) du LAN 3 (lien à 256kb/s).*

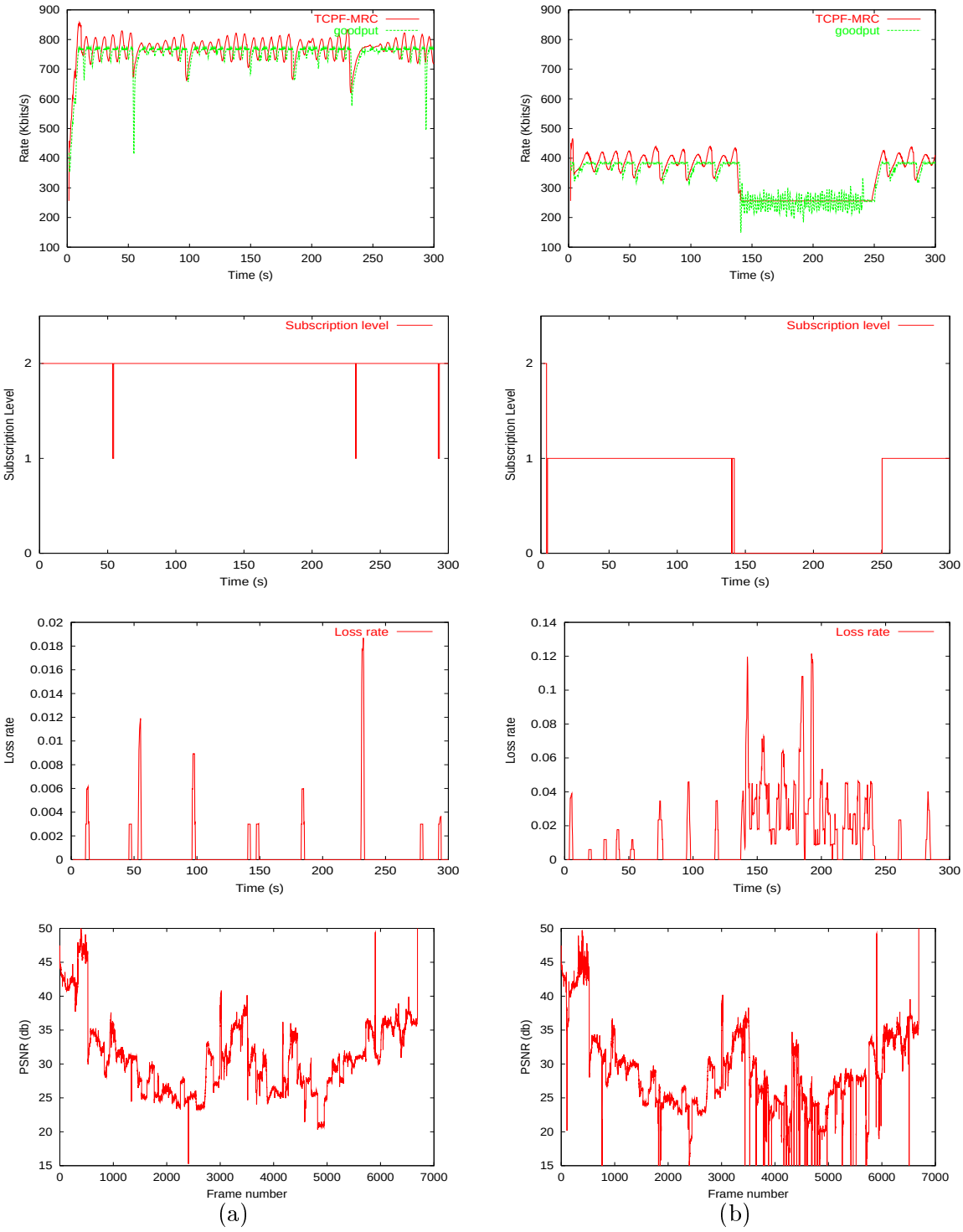

FIG. 3.18 – *scénario 1 : Débit requis par l'approche TCPF-MRC vs mesure réelle de goodput, taux de pertes et niveau de souscription pour les clients (a) du LAN 2 (lien à 768kb/s) (b) du LAN 4 (lien à 384kb/s).*

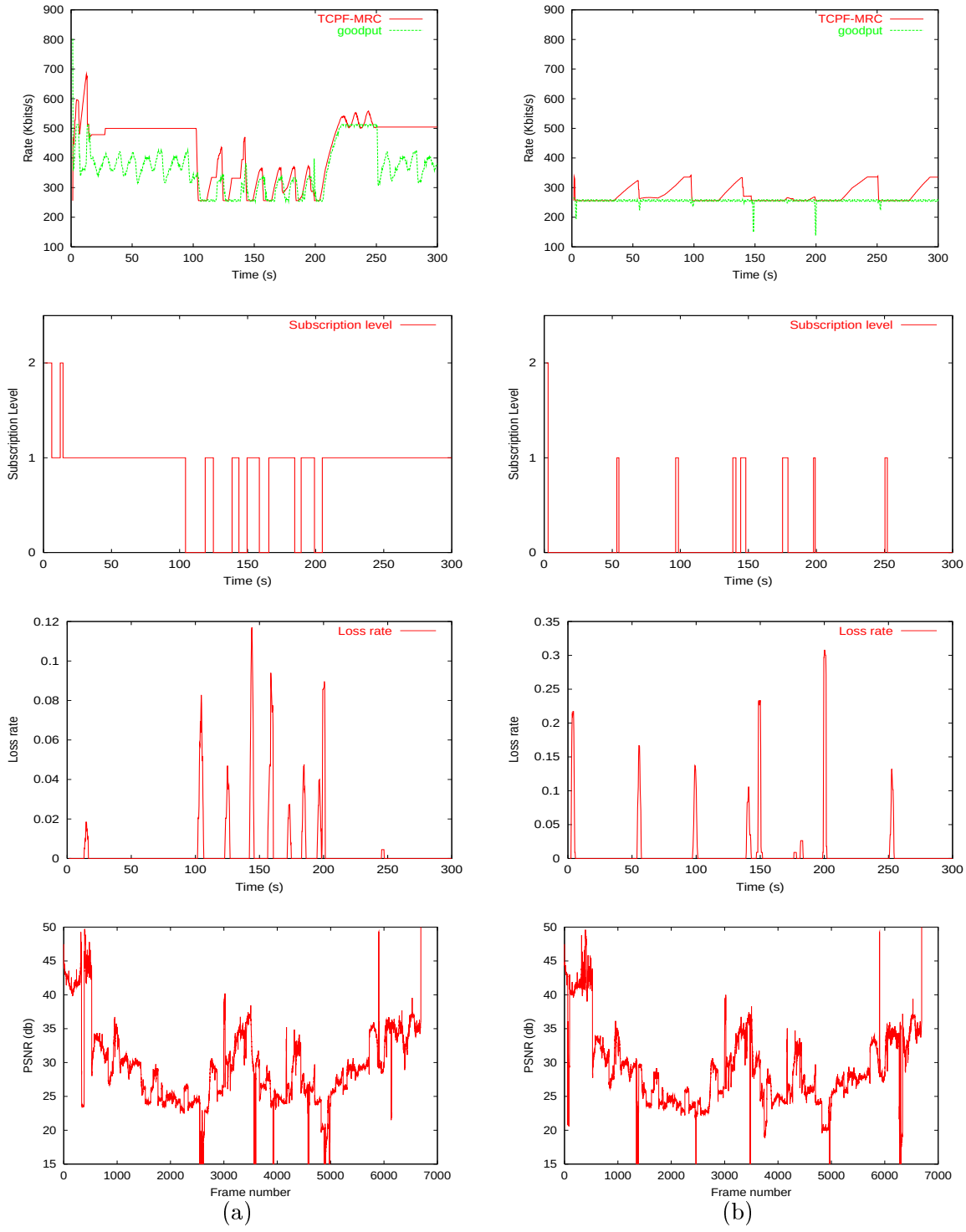

FIG. 3.19 – *scénario 1 : Débit requis par l'approche TCPF-MRC vs mesure réelle de goodput, taux de pertes et niveau de souscription pour les clients (a) du LAN 1 (lien à 512kb/s) (b) du LAN 3 (lien à 256kb/s).*

3.6.2.3 Analyse de l'équité vis-à-vis de TCP

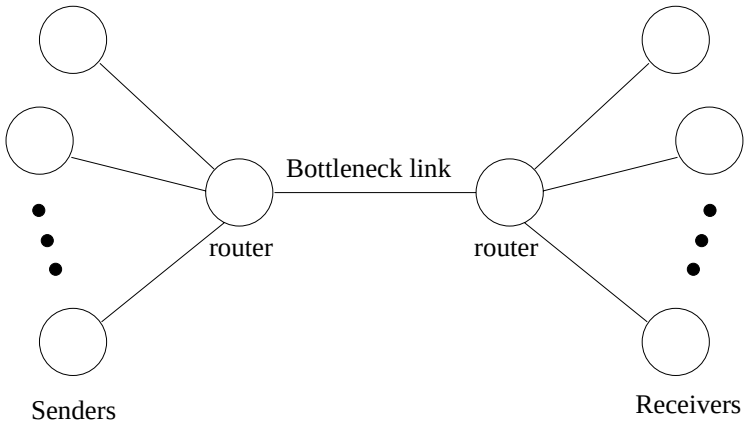

FIG. 3.20 – *Topologie de simulation de Goulot d'étranglement (bottleneck).*

Nous analysons ici l'équité du protocole proposé (GB-MRC et TCPF-MRC) face à des flux TCP concurrents quant au partage de bande passante. Pour cela, nous utilisons la topologie de réseau, représentée sur la figure 3.20, où un certain nombre de noeuds source sont connectés à des noeuds récepteurs via un lien commun à 8 Mb/s possédant un délai de transmission de 50ms. Les flux vidéo contrôlés par notre algorithme partagent le lien avec 15 connexions TCP.

La figure 3.21 montre le débit d'un flux vidéo contrôlé en utilisant l'algorithme GB-MRC face à 2 des 15 flows TCP concurrents. La figure 3.22 montre les résultats obtenus en utilisant la mesure TCPF-MRC. Les différentes expériences menées montrent que la régulation de flux utilisant la mesure GB-MRC ne permet pas un partage équitable de la bande passante. En présence de trafic concurrent important, la bande passante estimée décroit inéxorablement vers 0 (ou vers $R_{min} = 256$Kb/s ici puisque l'on force une borne min). Ceci est dû au fait que de part l'utilisation du *goodput*, l'algorithme n'est pas assez agressif et subit par contre l'agressivité des flux TCP. A l'inverse, comme on peut le voir sur la figure 3.22, le débit moyen du flux régulé par la mesure TCPF-MRC est en phase avec le débit moyen des flux TCP. Il apparait clairement ici qu'une approche basée *goodput* n'est pas réaliste et ne peut être mise en oeuvre sur une architecture réseau fournissant un service 'au mieux' (Best-effort) comme l'Internet actuel.

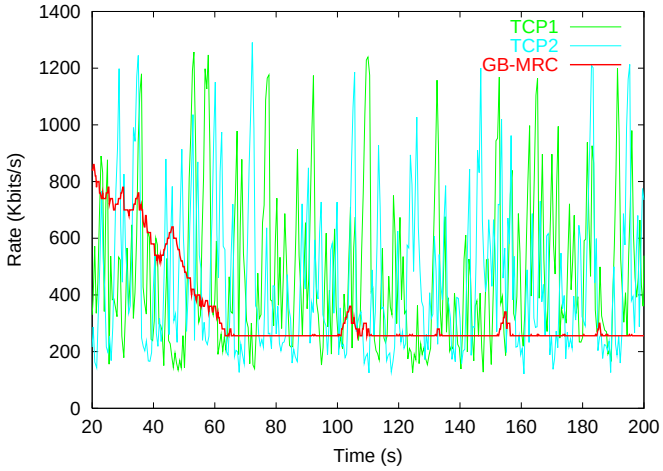

FIG. 3.21 – *Débits respectifs de 2 flux TCP (parmi 15) et d'un flux vidéo contrôlé par GB-MRC.*

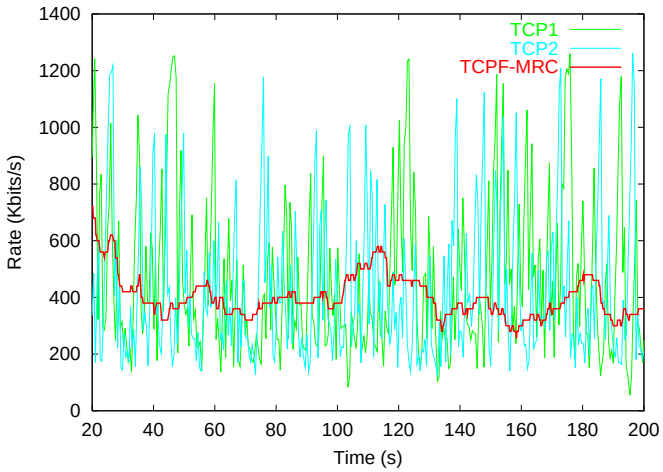

FIG. 3.22 – *Débits respectifs de 2 flux TCP (parmi 15) et d'un flux vidéo contrôlé par TCPF-MRC.*

3.6.2.4 Résultats avec la méthode TCPF-MRC avec ajout de FEC

De plus, afin d'évaluer l'impact des FEC sur le résultat du rendu, nous avons considéré l'approche TCP-compatible (TCPF-MRC) avec ajout de FEC. Le cas sans FEC est un cas particulier du cas avec FEC, il correspond au cas où le paramètre k_i de chaque couche i est fixé à n (i.e. 10 dans les expériences présentes). Ce scénario sera le **scénario 2**.

Les figures 3.23, 3.24 et 3.25 illustrent les résultats obtenus. Les FEC permettent d'améliorer la qualité, et notamment pour les récepteurs du LAN 4 (cf. figure 3.24-(b)). Cependant, on peut voir sur la figure 3.23 que l'utilisation de FEC mène à un comportement légérement moins stable. Elle induit, en effet, de plus fortes fluctuations de débit des différentes couches de la source FGS.

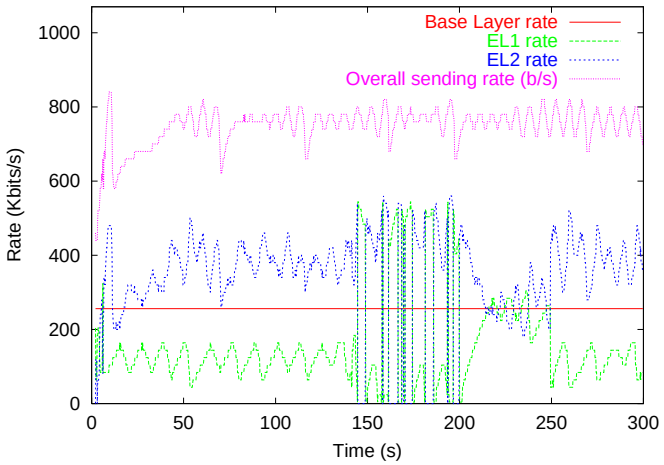

FIG. 3.23 – *scénario 2: Variations de débit de chaque couche de la source vidéo avec l'approche TCPF-MRC et des FEC additionnels (TCPF-MRC+FEC).*

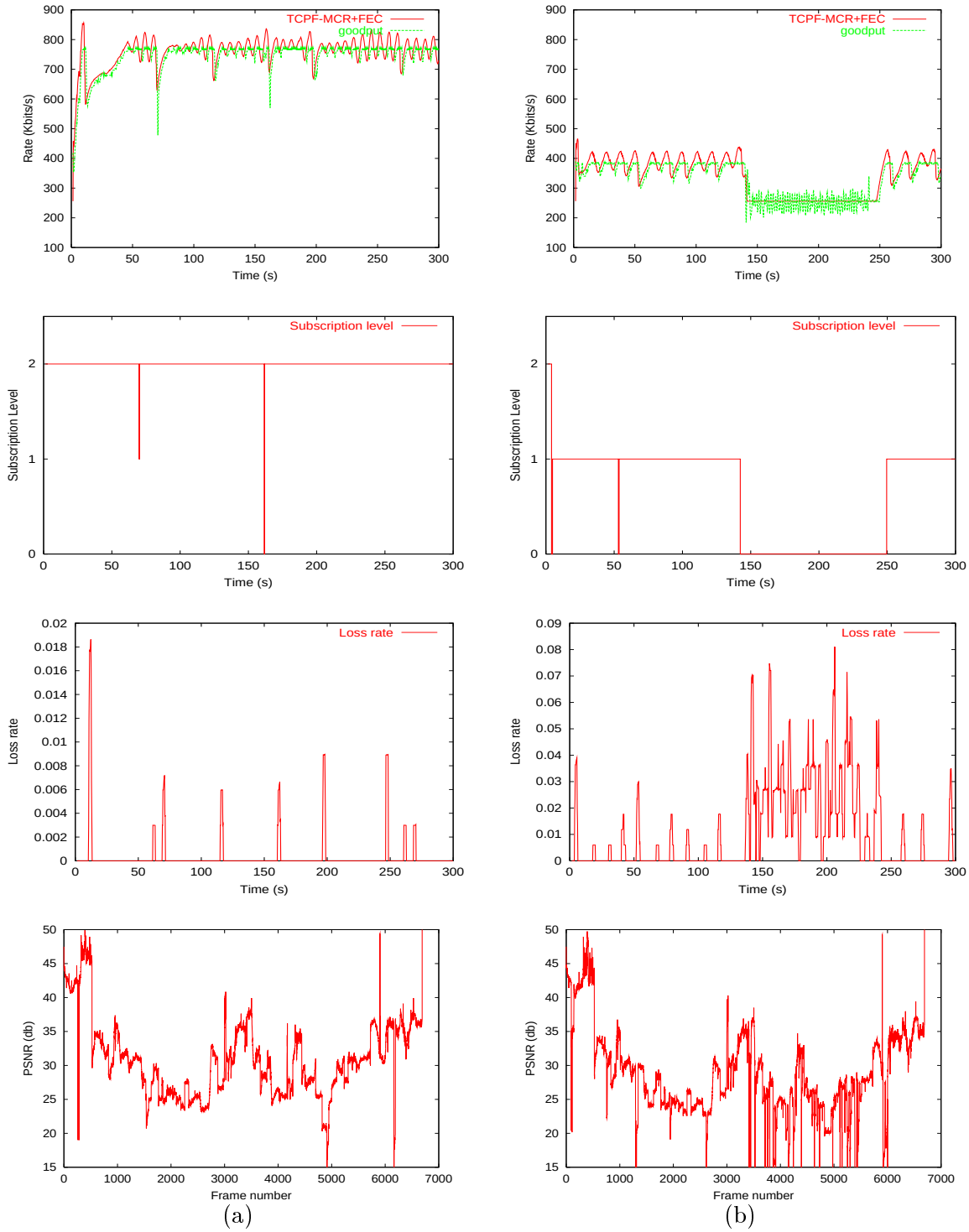

FIG. 3.24 – *scénario 2 : Débit prédit par l'approche TCPF-MRC+FEC vs mesure réelle de goodput, taux de pertes et niveau de souscription pour les clients (a) du LAN 2 (lien à 768kb/s) (b) du LAN 4 (lien à 384kb/s).*

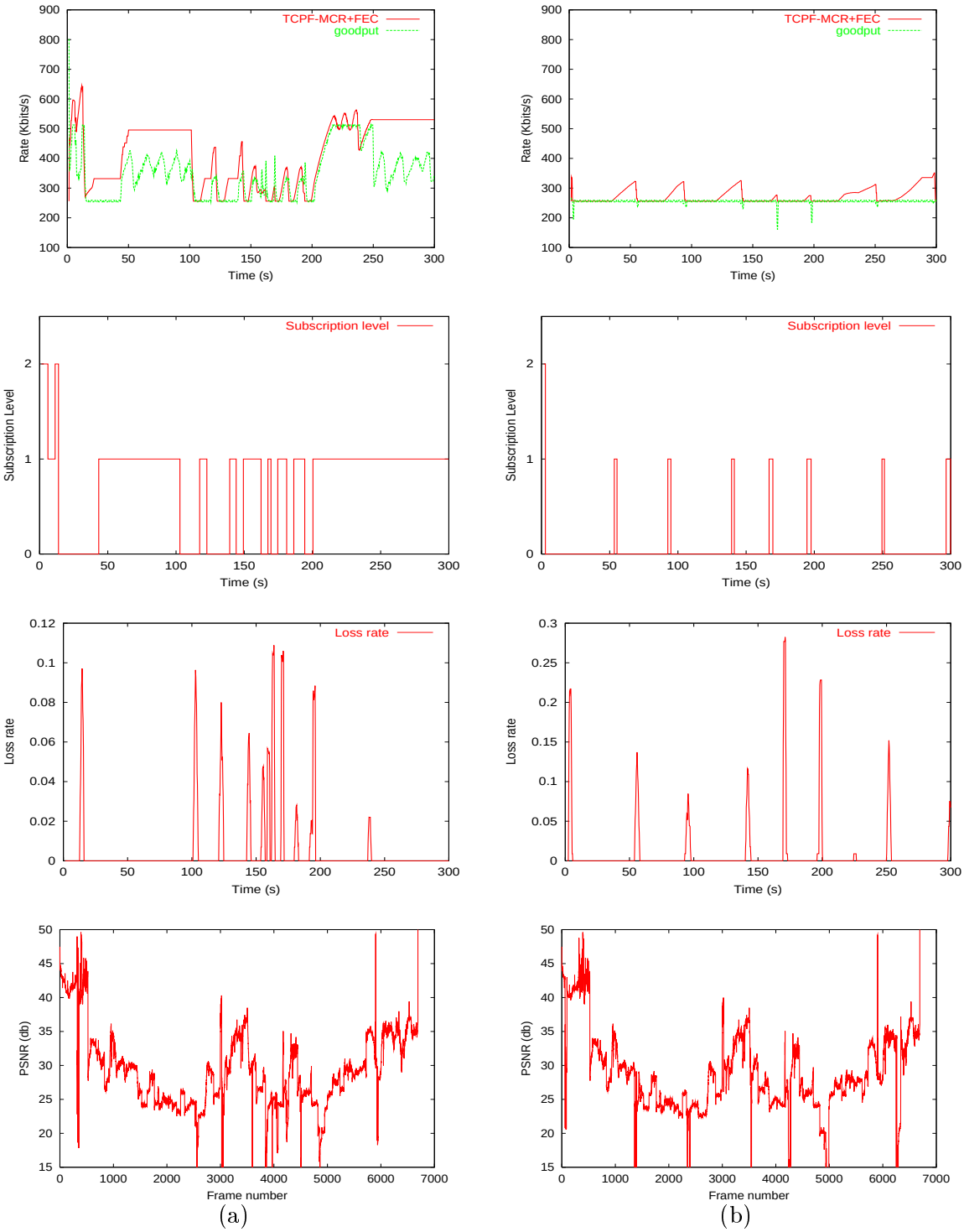

FIG. 3.25 – *scénario 2 : Débit prédit par l'approche TCPF-MRC+FEC vs mesure réelle de goodput, taux de pertes et niveau de souscription pour les clients (a) du LAN 1 (lien à 512kb/s) (b) du LAN 3 (lien à 256kb/s).*

3.7 Conclusion

Dans ce chapitre, nous avons présenté un nouveau système de diffusion vidéo FGS multicouche dans lequel le nombre de couches, leur débit, ainsi que leur niveau de protection sont dynamiquement adaptés afin d'optimiser la qualité de service de la session multimédia multicast. Un mécanisme de groupement distribué est utilisé afin de classifier les récepteurs en fonction de leur taux de pertes et la bande passante estimée sur leur lien. Les résultats ont montré la capacité du mécanisme à s'adapter aux fluctuations de la bande passante disponible dans l'arbre multicast, et dans un même temps sa capacité à gérer des taux de pertes fluctuants. Nous soulignons également l'intérêt de mettre en oeuvre une politique intelligente d'abonnements/désabonnements afin d'éviter un gaspillage de bande passante mais aussi une instabilité de la session. Nous avons montré également que l'utilisation du taux de pertes et d'une estimation de bande passante TCP-compatible comme variables discriminantes dans le mécanisme de classification mène à une qualité globale supérieure à celle obtenue en utilisant le couple taux de pertes et *goodput*. Enfin, fort des propriétés précédentes, le protocole se comporte de manière équitable avec TCP en terme de partage de bande passante.

II Codage vidéo scalable nouvelle génération

Chapitre 4

Codage scalable : état de l'art

4.1 Introduction

La transmission de données multimédia sur l'Internet doit faire face aux caractéristiques hétérogènes et variant dans le temps du réseau sous-jacent mais également aux capacités de traitement des récepteurs. Les applications de diffusion vidéo unicast ou multicast, telles que la vidéo-conférence ou la vidéo à la demande, s'en trouvent fortement pénalisées. Pour permettre une adaptation des données multimédia aux capacités et besoins des clients, il pourrait être intéressant de fournir une certaine flexibilité de manipulation des données dans le domaine compressé. Ceci est notamment vrai dans le cadre de sources pré-encodées. Le codage scalable ou hiérarchique apporte une solution élégante à ce problème.

Définition 4.1.1 (Scalabilité) *La scalabilité désigne ici l'aptitude d'un algorithme de compression à représenter une source hiérarchiquement sur plusieurs trains binaires. Parmi ceux-ci, une couche de base est indépendamment décodable des autres et permet la reconstruction des données à un niveau de qualité minimum. La prise en compte de sous-flux binaires de raffinement par le décodeur permet d'obtenir des incréments de qualité de reconstruction. Ces sous-flux peuvent être emboîtés dans un même train binaire ou diffusés sur des canaux distincts.*

Plusieurs schémas de codage scalable ont été proposés dans la littérature. Nous essaierons ici d'en montrer un large panel en nous focalisant sur notre finalité qui est la compression vidéo. Nous définirons les différents types de scalabilité étudiées par le passé. Il sera ensuite question des mécanismes de codage scalable proposés dans les standards vidéo actuels (H.263+ et MPEG-4). Puis, nous présenterons une alternative aux solutions actuelles utilisant une décomposition multirésolution basée sur une transformation en ondelettes. Dans ce cadre, nous proposons, tout d'abord, une étude portant sur différents schémas de compression progressif d'images fixes basés sur une décomposition en ondelettes 2D proposées dans la littérature. Puis, il sera question de l'extension des outils de transformation 2D à la dimension temporelle dans le cas de séquences d'images. Enfin, nous présentons les principaux codeurs de la littérature se basant sur une décomposition spatio-temporelle (2D+t).

4.2 Scalabilité : définitions

4.2.1 Définitions

Différentes natures de scalabilité de source vidéo existent et ont été étudiées. Dans les définitions qui suivent, nous désignons par couche de base ou de référence un sous-flux de la hiérarchie du codage scalable, destiné à être raffiné. Nous désignons par couche de raffinement la couche directement supérieure qui augmente donc d'un niveau de qualité sa couche de référence.

Définition 4.2.1 (Scalabilité en débit) *La scalabilité en débit propose une découpe multiniveau sur la base de budgets incrémentaux en débit.*

Définition 4.2.2 (Scalabilité temporelle) *La scalabilité temporelle définit une hiérarchie de résolutions temporelles.*

Définition 4.2.3 (Scalabilité SNR) *La scalabilité SNR (Signal to Noise Ratio) ou en qualité consiste à augmenter le rapport signal à bruit d'une couche donnée de la hiérarchie, c'est-à-dire à réduire la distorsion de quantification entre images originales et reconstruites. Pour cela, une couche de raffinement transporte de l'information de correction des erreurs de quantification de sa couche de référence.*

Définition 4.2.4 (Scalabilité spatiale) *La scalabilité spatiale ou en résolution définit une hiérarchie de résolutions spatiales. La résolution désigne ici la taille en pixels des images reconstruites.*

Définition 4.2.5 (Scalabilité de complexité) *La scalabilité de complexité porte sur la complexité de l'algorithme de décodage des trains de bits reçus.*

La section suivante présente les techniques de scalabilité adoptées dans les standards vidéo H.263+ et MPEG-4.

4.3 Scalabilité dans les standards vidéo actuels

4.3.1 Scalabilité dans la norme H263+

Trois des types de scalabilité définis en section 4.2 sont prévus dans la norme H263+: les scalabilités temporelle, SNR et spatiale [Sul99]. La norme définit 3 types d'images dans une couche de raffinement: les images B, EI et EP.

4.3.1.1 Scalabilité temporelle

La scalabilité temporelle dans H263+ consiste en l'insertion d'images B entre des paires d'images de référence, ne pouvant être que des images I, P, EI ou EP (voir figure 4.1). Le train binaire formant une couche de raffinement contient donc des informations de mouvement et d'erreur de compensation en mouvement bidirectionnelle.

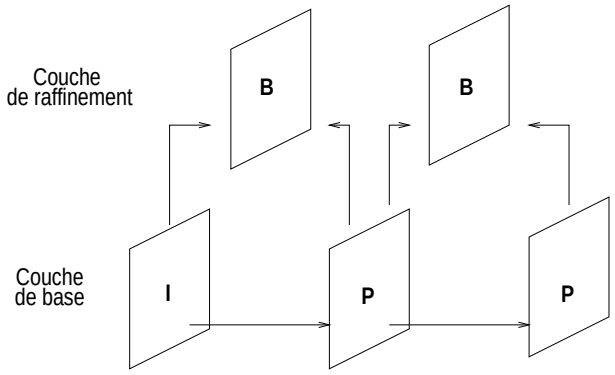

FIG. 4.1 – *Scalabilité temporelle dans la norme H.263+*

4.3.1.2 Scalabilité SNR

Dans le cadre de la scalabilité SNR, la couche de raffinement est constituée d'images EI et EP.

- Une image EI (*Enhancement I-picture*) n'est prédite qu'à partir de l'image correspondante dans la couche de référence (voir figure 4.2). Une image EI peut-être prédite à partir d'une image I ou P de la couche de base.
- Une image EP (*Enhancement P-picture*) est prédite à la fois à partir de son image de référence et de l'image EI ou EP qui la précède dans la couche de raffinement (voir figure 4.2). Le choix entre prédiction intra-couche ou inter-couche est établi pour chaque macrobloc via l'énergie (variance) du macrobloc d'erreur de prédiction.

La prédiction inter-couche n'utilise pas de vecteur de mouvements, au contraire de la prédiction entre images de même couche.

4.3.1.3 Scalabilité spatiale

La scalabilité spatiale mise en place dans la norme H.263+ prévoit le sur-échantillonnage et l'interpolation des images de la couche de référence pour prédire les images de la couche de raffinement. Cette interpolation d'un facteur deux peut être horizontale, verticale ou les deux. La figure 4.3 illustre la scalabilité spatiale avec l'exemple d'une interpolation à la fois horizontale et verticale. Une image de référence utilisée dans la prédiction des images de la couche de raffinement doit être une image I, P ou la partie P d'une image PB. Le train binaire d'une couche de raffinement contient donc l'erreur de prédiction spatiale par sur-échantillonnage et interpolation des images de référence.

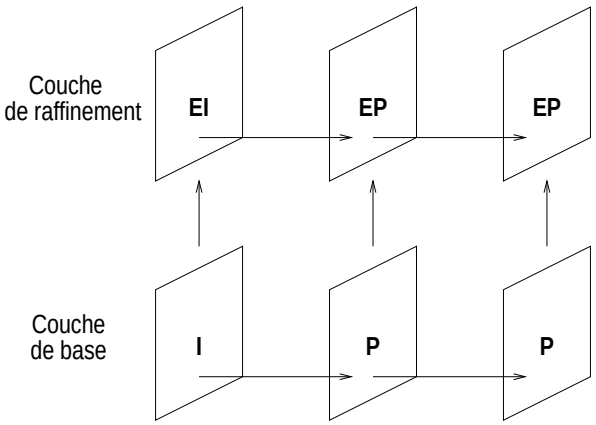

FIG. 4.2 – *Scalabilité SNR dans la norme H.263+*

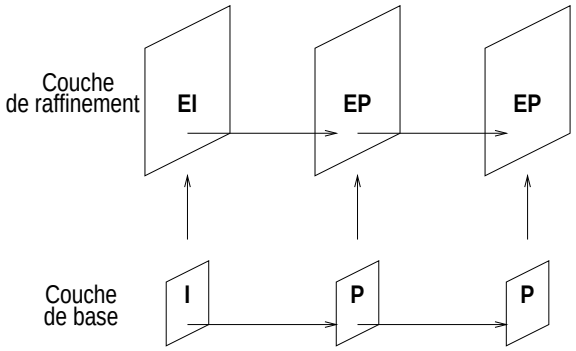

FIG. 4.3 – *Scalabilité spatiale dans la norme H.263+*

4.3.1.4 Scalabilité hybride

La norme H.263+ prévoit une scalabilité hybride, qui combine les différents modes de scalabilité présentés ci-dessus. Ainsi, un flux vidéo scalable hybride peut combiner des couches de raffinement SNR et spatiales. De plus, des images B peuvent éventuellement être insérées entre des images EI et EP. La scalabilité hybride est illustrée figure 4.4. Quelques règles y sont associées :

- La résolution ne peut pas décroître d'une couche de référence à une couche de raffinement.
- Toute image d'une couche de raffinement correspondant temporellement à une image B d'une couche plus basse doit être une image B. Les résolutions spatiales de ces images B sont alors nécessairement les mêmes.

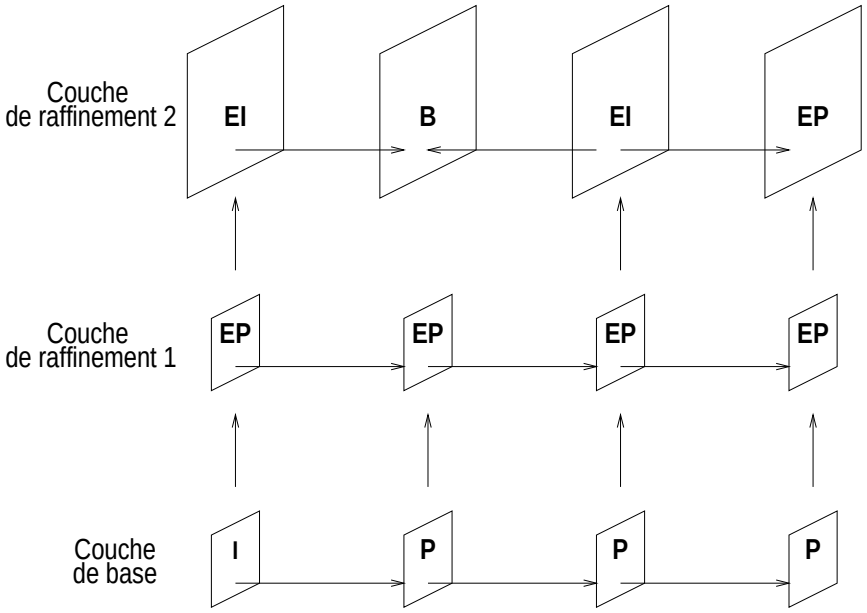

FIG. 4.4 – *Scalabilité hybride dans la norme H.263+*

4.3.2 Scalabilité dans la norme MPEG-4

Dans la norme MPEG-4 [14498], on distingue le **mode rectangulaire** du *mode objet*.[1] Dans le mode rectangulaire, la vidéo est considérée comme une suite d'images rectangulaires de tailles fixes. Le mode objet permet, quant à lui, de coder des objets vidéo (VO : *Video Object*) de formes arbitraires préalablement extraits par segmentation de la séquence originale. A chaque objet de la scène est associé une ou plusieurs couches (VOL : *Video Object Layer*). Chacune de ces VOL contient un VOP (*Video Object Plane*). Une image est ainsi potentiellement formée de plusieurs VOP. Dans le cas du mode rectangulaire, on assimile une image à un VOP.

La norme MPEG-4 prévoit les scalabilités spatiale, temporelle ainsi qu'une certaine forme de scalabilité SNR par l'intermédiaire de la scalabilité FGS.

4.3.2.1 Scalabilité temporelle

Contrairement à la norme H.263+, la norme MPEG-4 prévoit en scalabilité temporelle l'insertion d'images I, P et B dans une couche de raffinement.

Mode rectangulaire

– Images I: les images Intra d'une couche de raffinement sont les mêmes que celles de la couche de base.

1. Il existe également un autre mode appelé *mode sprite*.

- Images P: le décodage des images P dans une couche de raffinement est identique au décodage dans la couche de base. Chaque image P est prédite à partir d'une image de référence qui peut être :
  - l'image décodée précédente au même niveau de la hiérarchie,
  - l'image précédente, en ordre d'affichage, appartenant à la couche de base,
  - la prochaine image, en ordre d'affichage, appartenant à la couche de base.
- Images B: comme pour les images prédites, le codage-décodage des images B des couches de raffinement est similaire à celui réalisé dans la couche de base. Le couple d'images de référence des images B peut être constitué de:
  - l'image précédente décodée dans la même couche et l'image précédente, en ordre d'affichage, appartenant à la couche de base,
  - l'image précédente décodée dans la même couche et la prochaine image, en ordre d'affichage, dans la couche de base,
  - l'image précédente et l'image suivante dans la couche de base.

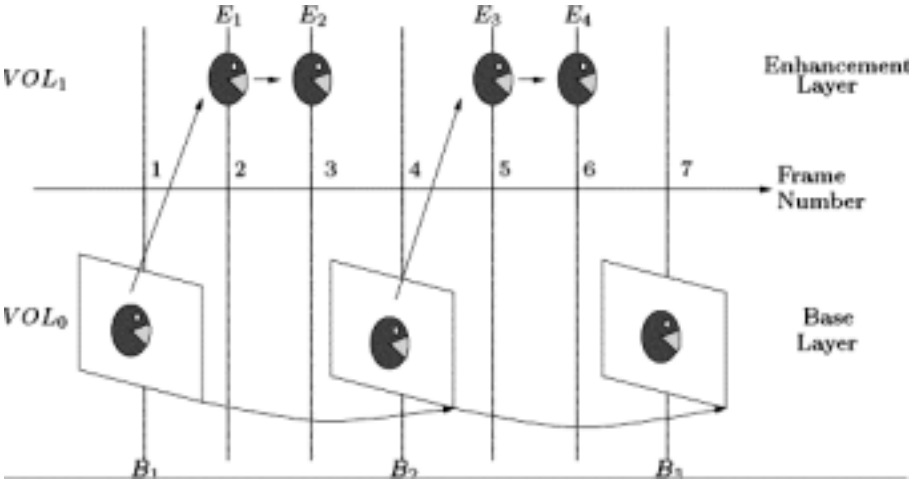

FIG. 4.5 – *Scalabilité temporelle de type 1 pour le mode objet dans la norme MPEG-4.*

Mode objet Dans le mode objet, deux types de scalabilité temporelle sont envisagés dans MPEG-4:

- type 1 : seule une région d'intérêt, extraite de la VOL de base, voit sa résolution temporelle augmenter (cf. figure 4.5);
- type 2 : une VOL de raffinement augmente la fréquence d'images des VOP entiers de la VOL de base (cf. figure 4.6).

Les techniques de codage employées en scalabilité temporelle basée objet MPEG-4 sont très similaires aux techniques utilisées en mode rectangulaire. Une description détaillée de la norme est disponible dans [14498].

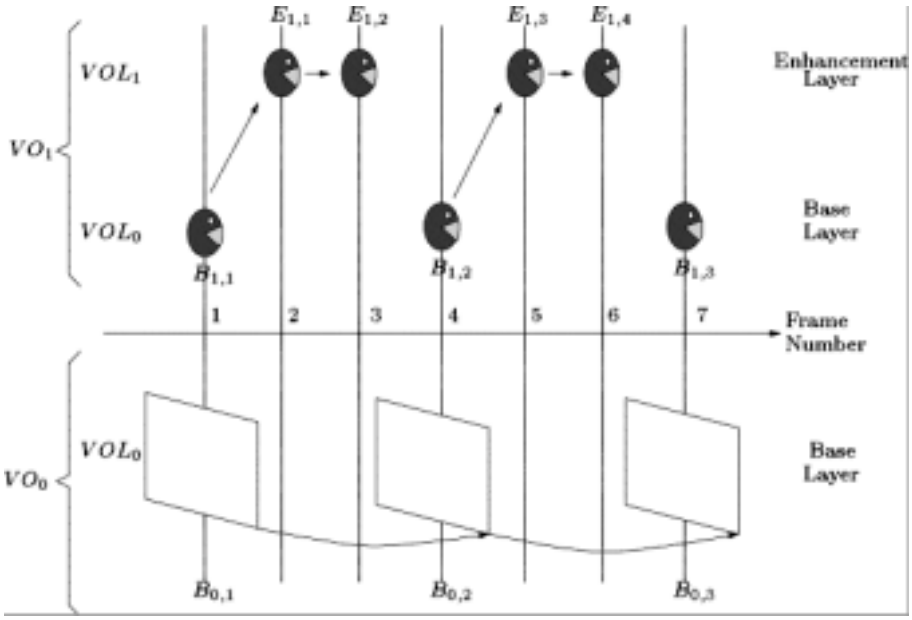

FIG. 4.6 – *Scalabilité temporelle de type 2 pour le mode objet dans la norme MPEG-4.*

4.3.2.2 Scalabilité spatiale

La scalabilité spatiale MPEG-4 consiste en une prédiction temporelle par compensation en mouvement d'une image de référence dans la couche de raffinement, et d'une prédiction spatiale depuis l'image décodée dans la couche inférieure. Ces prédictions sont soit sélectionnées individuellement, soit combinées pour former la prédiction effective. On distingue deux types de VOP :

- *P-VOP*: ceux-ci sont prédits spatialement depuis l'image correspondante dans la couche de référence.
- *B-VOP*: ceux-ci sont prédits de façon bidirectionnelle, de la même manière que dans H.263+.

4.3.2.3 Scalabilité à grain fin

La scalabilité à grain fin ou FGS (*Fine Granular Scalability*)[RC99], a été introduite afin de faciliter l'adaptation de débit des sources aux variations de bande passante du réseau notamment dans le cas d'applications de streaming avec des flux pré-encodés. Cette approche a pour but de fournir une granularité d'adaptation plus fine que celle permise par les représentations scalables classiques. Dans ce schéma, deux couches sont produites: une couche de base (codée selon un schéma MPEG-4 non scalable classique) et une couche de raffinement FGS. Cette couche de raffinement code l'erreur de quantification obtenue dans la couche de base de manière progressive en utilisant des plans de bits. La qualité évolue selon le décodage de plus ou moins de plans de bits

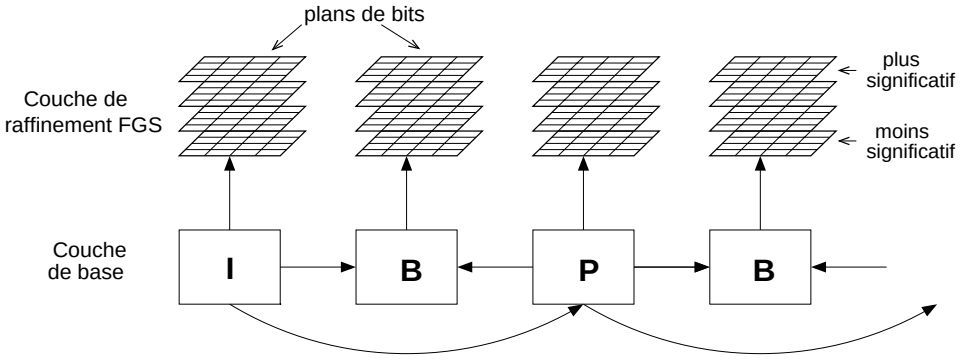

FIG. 4.7 – *Scalabilité FGS dans la norme MPEG-4.*

de cette couche d'amélioration. Le train binaire de la couche FGS peut être tronqué en tous points selon les besoins et les fluctuations de la bande passante.

4.3.3 Limites des standards et nouvelles orientations

La granularité d'adaptation permise par les représentations scalables proposées dans les standards restent encore très grossière. L'approche FGS décrite ci-dessus propose donc une alternative intéressante. Toutefois, ses performances débit-distorsion, ainsi que celles des approches précédentes, restent très médiocres.

La principale raison pour laquelle ces performances restent réduites sont dûes au fait que les algorithmes de codage vidéo standards ne sont pas naturellement scalables. Ils se contentent de corriger l'erreur de quantification de la couche de base, ou bien de dupliquer la boucle de prédiction temporelle dans les couches de raffinement.

L'idée est donc de mettre en oeuvre des schémas de codage progressif menant plus facilement à l'obtention d'une scalabilité à granularité fine. L'utilisation de méthodes basées sur une quantification progressive directe des données est envisageable. Toutefois, dans ce type de schéma la scalabilité n'est pas vraiment naturelle. C'est pourquoi, d'autres approches, se basant sur l'utilisation d'une représentation multirésolution fournie par une décomposition en ondelettes, ont été proposées. Ces représentations fournissent des voies naturelles vers l'obtention conjointe des scalabilités spatiale, temporelle et SNR. Dans les sections suivantes, nous nous intéresserons plus particulièrement à ce type d'approche.

4.4 Représentation multirésolution : transformation en ondelettes

Le concept de multirésolution est un concept dont la philosophie semble assez naturelle en soi. L'idée est d'examiner le signal à différentes échelles et résolutions afin d'extraire certaines caractéristiques spatiales ou fréquentielles par transformation. De nombreuses techniques, étudiées indépendamment en mathématiques appliquées, en

physique théorique et en traitement du signal, ont été développées pour y parvenir. Dans [BA83], les auteurs proposent une transformation d'une image sous forme pyramidale, par application de banc de filtres. De nombreux travaux ont alors été entrepris dans la même voie [WO86, ASH87] afin de formaliser mathématiquement les propriétés particulières d'orthogonalité de la transformation ou de reconstruction parfaite. Dans [Mal89], Mallat montre que la transformation en ondelettes discrètes [LM86], qui n'est autre qu'une projection sur des fonctions discrètes orthonormales, peut également être vue comme une manière de réaliser un analyse multirésolution. L'analyse multirésolution et la transformée en série d'ondelettes apparaissent alors comme deux manières de formaliser le même concept, et dans le cas discret rejoignent la théorie des bancs de filtres. En effet, la transformée en ondelettes peut-être vue comme décomposition dyadique en sous-bandes réalisée par filtrage linéaire à l'aide d'une paire de filtres d'analyse (passe-haut, passe-bas).

Cette transformation en ondelettes devient alors un nouvel outil de décorrélation présentant les propriétés de bonnes localisations spatiale et fréquentielle, qui sont des propriétés très intéressantes dans le domaine du traitement de l'image et notamment de la compression. Dans la suite cette transformée sera notée DWT (*Discrete Wavelet Transform*).

4.4.1 Principe

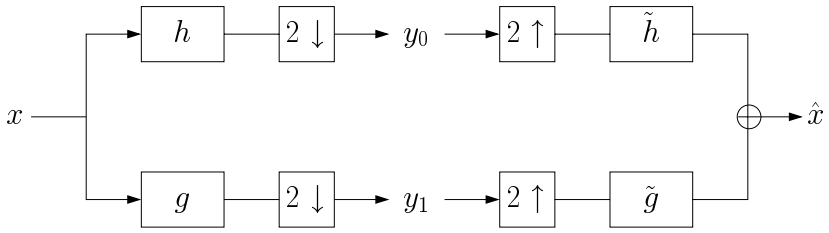

FIG. 4.8 – *Analyse-Synthèse par ondelettes.*

Le principe de base d'une analyse multirésolution par ondelettes est de décomposer un signal $x[n]$ monodimensionnel en deux sous-bandes : basse fréquence et haute fréquence. Pour cela, la décomposition est réalisée à l'aide d'une paire de filtres d'analyse séparables g, h, l'un étant passe-bas (g) et l'autre passe-haut (h). Après chaque phase de filtrage, les signaux sont décimés afin de rester à échantillonnage critique. Soit $y[n]$ les coefficients entrelacés des sous-bandes obtenues, avec $n \in [0, N]$. Les coefficients de la sous-bande passe-bas sont les coefficients pairs $y[2n]$ et ceux de la sous-bande haute fréquence sont les coefficients impairs $y[2n+1]$. Ainsi les opérations d'**analyse** peuvent s'écrire :

$$\begin{cases} y[2n] &= \sum_k h[k].x[2n+k], \\ y[2n+1] &= \sum_k g[k].x[2n+1+k]. \end{cases} \tag{4.1}$$

L' équation de **synthèse** correspondante s'écrit :

$$\hat{x}[n] = \sum_k y[2k].\tilde{h}[n-2k] + y[2k+1].\tilde{g}[n-(2k+1)] \tag{4.2}$$

avec $\tilde{h}$ et $\tilde{g}$ les filtres passe-bas et passe-haut de synthèse.

4.4.2 Pourquoi des filtres biorthogonaux?

La définition des filtres de synthèse et d'analyse découle des différentes contraintes de *design* que l'on impose au banc de filtres. Ainsi, en l'exprimant par l'intermédiaire de la transformée en z, une reconstruction parfaite est assurée si les deux équations suivantes sont vérifiées:

$$\begin{cases} \frac{1}{2}\left[\tilde{g}(z)g(z) + \tilde{h}(z)h(z)\right] = z^{-d} \\ \frac{1}{2}\left[\tilde{g}(z)g(-z) + \tilde{h}(z)h(-z)\right] = 0 \end{cases} \tag{4.3}$$

avec d le retard du banc de filtres. Les premiers bancs de filtres mis en oeuvre ne font qu'approcher cette reconstruction parfaite: ils s'attachent uniquement à minimiser l'erreur de reconstruction et font appel à des filtres QMF (*Quadrature Mirror Filters*) [Joh80]. Puis, assurant l'équation de reconstruction parfaite, tout en garantissant l'orthogonalité, une nouvelle famille de filtres, les filtres CQF (*Conjugate Quadrature Filters*) [SB86], a été exploitée.

En traitement d'images, la phase est très importante. il est donc essentiel que les filtres d'analyse/synthèse soient symétriques, c'est-à-dire à phase linéaire. Ceci permet d'éviter notamment l'apparition de distorsions gênantes pour l'oeil. Il est préférable, de plus, d'utiliser des filtres stables et peu longs pour des raisons de coût de calcul. Malheureusement, toutes ces conditions ne peuvent être satisfaites simultanément par une ondelette à l'exception des filtres particuliers de Haar [Haa10]. Ainsi, la phase linéaire impose de relâcher la contrainte d'orthogonalité en utilisant des *filtres biorthogonaux*. Ces derniers ont été proposés indépendamment dans [CDF89, CDF92] et [VH90]. Les schémas communément utilisés en traitement d'images s'appuient sur des décompositions en ondelettes biorthogonales.

4.4.3 Le schéma Lifting

Le but du schéma *lifting*, introduit par Sweldens [Swe95], est de proposer une transformée en ondelettes par un procédé simple, réversible et rapide. C'est une alternative intéressante au schéma convolutif de la transformée en ondelettes classique. Ce schéma proposé initialement pour la compression sans perte, est également, du fait de sa rapidité très compétitif pour la compression avec pertes. Ce schéma s'effectue en trois étapes présentées sur la figure 4.9. Le signal original x est partitionné en deux sous-signaux x_e (l'ensemble des échantillons d'indice pair) et x_o (l'ensemble des échantillons d'indice impair). Un opérateur de prédiction P est appliqué au sous-ensemble x_e. Il s'exprime par :

$$d = x_o - P(x_e). \tag{4.4}$$

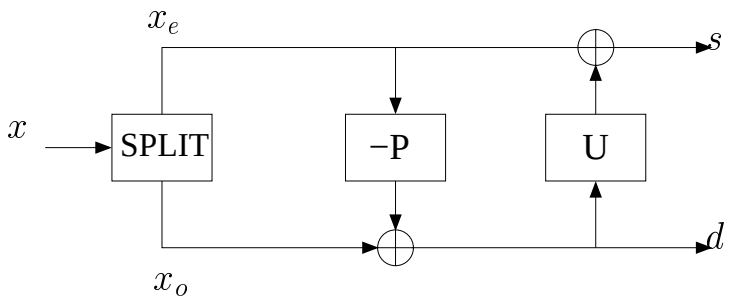

FIG. 4.9 – *Principe du schéma de lifting.*

On obtient alors un signal différentiel d qui représente l'erreur de prédiction sur x_o. Cette étape définit ce que l'on appelle un **pas** de lifting.

Le pas de lifting dual U effectue une mise à jour du signal pair à partir du signal différentiel afin de pallier les problèmes d'*aliasing* (conservation d'énergie). Le signal mis à jour obtenu s s'exprime par

$$s = x_e + U(d). \tag{4.5}$$

Suivant le filtre de transformée en ondelettes utilisé, nous pouvons avoir plusieurs étages de pas lifting et de pas lifting dual, ce qui se traduit, pour M par:

$$\begin{cases} d_i[n] &= d_{i-1}[n] - \sum_k P_i[k].s_{i-1}[n+k], \\ s_i[n] &= s_{i-1}[n] + \sum_k U_i[k].d_i[n+k] \end{cases} \tag{4.6}$$

avec i le numéro d'étages courant et d_i la valeur de d après l'étage i.

Une fois que tous les pas de lifting nécessaires sont appliqués, les nouveaux signaux s et d sont normalisés par un facteur positif. Le signal s peut être identifié comme le signal basse fréquence et d comme le signal haute fréquence. La transformée inverse est simplement donnée en inversant l'ordre des équations ainsi que des signes. L'avantage de cette méthode est que l'algorithme est simple et l'inversion facile.

Remarque :
Une description de la construction du schéma de lifting à partir des filtres convolutifs est donnée dans l'annexe C et de manière plus détaillée dans [DS98].

4.5 Codeurs finement scalables d'images fixes basés ondelettes 2D

L'apparition des transformées en ondelettes a permis une évolution des opérations de codage de par les propriétés particulières des coefficients. En effet, leur organisation

sous forme de pyramide multirésolution est particulièrement adaptée à un codage progressif. Malgré le travail de décorrélation opéré par la transformation, il existe tout de même des dépendances entre les coefficients des différentes sous-bandes fréquentielles. De plus, au sein même des sous-bandes, les coefficients voisins ne sont pas complètement indépendants. C'est pourquoi, de nombreux travaux ont porté sur la définition de schémas de compression tirant parti de ces dépendances.

Dans cette section, nous décrivons quatre des algorithmes de progressivité, basés sur une représentation multirésolution de l'image, les plus utilisés dans la littérature. Les algorithmes EZW [2][Sha93] et SPIHT [3][SP96] exploitant les corrélations inter sous-bandes. Puis, l'algorithme EBCOT [4] [Tau00] exploitant quant à lui les corrélations intra sous-bandes. Enfin, l'algorithme EZBC [5][HW00a] qui tend à exploiter à la fois les dépendances inter et intra sous-bandes.

Nous introduisons ces quatre approches dans cette section car la plupart des algorithmes de codage vidéo finement scalable reposent sur l'utilisation de ces techniques de compression progressive d'images fixes . Ceci ne constitue pas un état de l'art exhaustif sur les codeurs progressifs basés ondelettes 2D mais prépare à la lecture de la section suivante, consacrée au codage vidéo finement scalable nouvelle génération.

4.5.1 Représentation multirésolution d'une image : ondelettes 2D

La théorie des ondelettes peut facilement se généraliser en plusieurs dimensions. Nous étudierons particulièrement l'approche bidimensionnelle. L'approche la plus utilisée repose sur une analyse dyadique multirésolution. Les filtres d'analyse g, h sont des filtres séparables qui sont successivement appliqués sur les lignes, puis sur les colonnes de l'image, leurs sorties respectives subissant un sous-échantillonnage par un facteur deux. Pour un niveau de décomposition, nous obtenons ainsi quatre sous-bandes. Ainsi, nous avons une sous-bande passe-bas notée LL relative à la moyenne de l'image et trois sous-bandes passe-haut LH, HL et HH, représentant respectivement les contours horizontaux, verticaux et diagonaux. Dans le cas d'une décomposition dyadique à N niveaux, l'opération est renouvelée N fois sur la sous-bande basse fréquence du niveau précédent (cf figure 4.10). Nous obtenons ainsi une représentation multirésolution de l'image de départ. La transformée en ondelettes est, de plus, dans le domaine du codage d'image, un outil de décorrélation présentant des propriétés de bonnes localisations spatiale et fréquentielle.

L'élaboration d'une hiérarchie de codage d'une image ainsi décorrélée est naturelle. A chaque niveau de résolution, l'ajout de sous-bandes de fréquences supérieures vient augmenter la résolution de l'image reconstruite. Ainsi, la couche de base contient l'image à la résolution la plus grossière (i.e. la sous-bande LL_2). Cette couche est ensuite raffinée progressivement par les sous-bandes HL_2, HL_2, HH_2 afin de reconstruire l'image à une résolution deux fois supérieures notée LL_1. Le processus peut-être répété jusqu'à obtenir

2. Embedded Zerotree of Wavelet
3. Set Partitionning in Hierarchical Trees
4. Embedded Block Coding with Optimized Truncation
5. Embedded ZeroBlocks of wavelet coding based on Context modeling

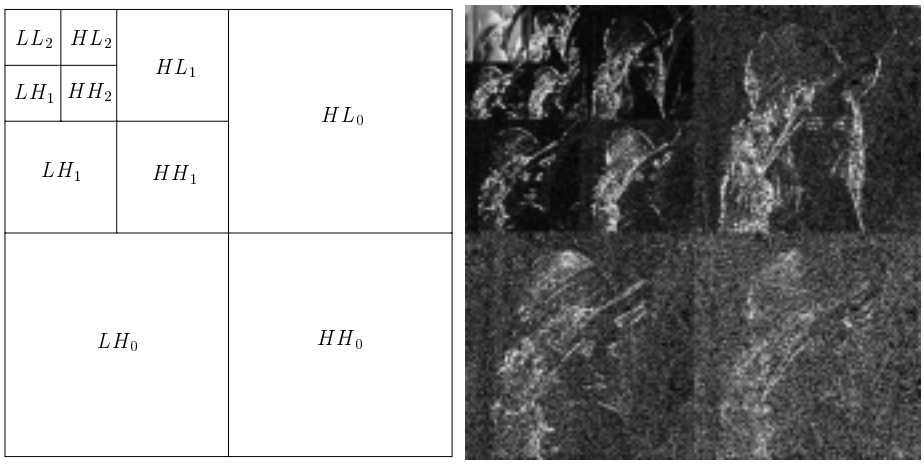

FIG. 4.10 – *Pyramide issue de la transformée en ondelettes.*

l'image pleine résolution.

4.5.2 Approche Inter sous-bandes : EZW et SPIHT

Les premiers algorithmes de compression se basant sur une représentation en ondelettes de l'image ont consisté à exploiter les dépendances statistiques existantes entre coefficients de mêmes localisations spatiales et orientations, mais appartenant à des sous-bandes de résolutions différentes. Nous décrivons dans cette partie les deux algorithmes fondateurs que sont EZW (*Embedded Zerotree of Wavelet*) et SPIHT (*Set Partitionning in Hierarchical Trees*).

4.5.2.1 Algorithme EZW (Embedded Zerotree Wavelet)

L'algorithme EZW[Sha93] définit des relations de dépendance parents-enfants, comme indiqué figure 4.11, où les flèches pointent des parents vers les enfants. L'hypothèse de décroissance du spectre exploitée dans EZW dit que si un coefficient d'ondelettes à une résolution grossière est insignifiant vis-à-vis d'un seuil, alors tous les coefficients de même orientation dans la même zone spatiale sont susceptibles d'être insignifiants vis-à-vis du même seuil.

L'algorithme EZW définit alors un premier seuil $T_0 = max\left\{\frac{|x|}{2}, x \in \{coef.DWT\}\right\}$. Un coefficient x est dit significatif par rapport à T_i si $|x| \geq T_i$. Le seuil courant utilisé sera divisé par deux à chaque itération de l'algorithme: $T_{i+1} = \frac{T_i}{2}$. Au cours de l'algorithme EZW, deux listes sont utilisées et mises à jour :

- une liste dominante $\mathcal{D}$ contient les coordonnées des coefficients non significatifs vis-à-vis du seuil T_i.
- une liste subordonnée $\mathcal{S}$ contient les amplitudes des coefficients ayant déjà été trouvés comme significatifs.

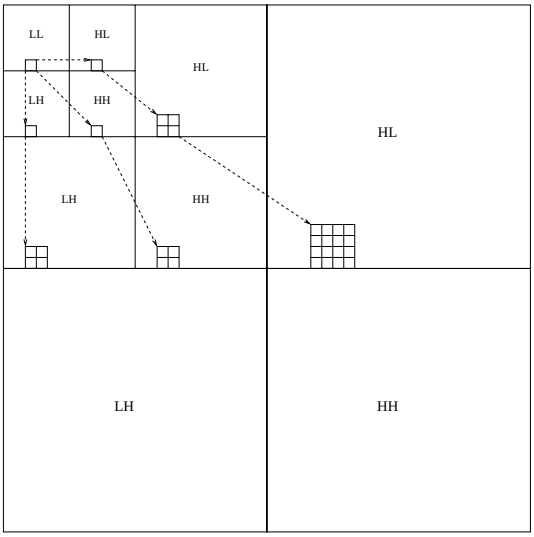

FIG. 4.11 – *Dépendances parents-enfants entre coefficients de différentes sous-bandes (EZW).*

L'algorithme se décompose alors en deux étapes.

1. Le parcours des coefficients dans l'ordre indiqué dans $\mathcal{D}$, et la formation d'une carte de significativité $\mathcal{C}$ sont effectués comme suit. Pour chaque coefficient d'ondelettes x :
   - si x est significatif et positif, le symbole P est noté dans $\mathcal{C}$. L'amplitude de x est insérée en fin de liste $\mathcal{S}$ et ses coordonnées sont retirées de $\mathcal{D}$;
   - si x est significatif et négatif, le symbole N est noté dans $\mathcal{C}$. L'amplitude de x est insérée en fin de liste $\mathcal{S}$ et ses coordonnées sont retirées de $\mathcal{D}$;
   - si x et tous ses descendants sont insignifiants on dit que x est la racine d'un *zerotree* (arbre de 0). Le symbole $\mathcal{Z}_t$ est noté dans $\mathcal{C}$;
   - si x est insignifiant mais un de ses descendants est significatif alors x est un zéro isolé, le symbole Z_i est inséré dans $\mathcal{C}$;
   - Si x est le fils d'un coefficient déjà marqué comme racine de *zerotree*, alors on le laisse de côté.

   La carte de significativité ainsi formée avec l'alphabet $\{P, N, Z_t, Z_i\}$ est codée avec un codeur entropique adaptatif et transmise.
2. Dans la deuxième étape de l'algorithme, un codage par plans de bits permet de combiner les deux étapes suivantes.
   - Le raffinement des amplitudes de coefficients déjà présentes dans $\mathcal{S}$. Un indice de quantification à '1' ou '0' permet de réduire l'intervalle d'incertitude dans lequel se trouve le coefficient traité.
   - Le codage des amplitudes des nouveaux arrivants. Un indice de quantification positionne le nouveau coefficient par rapport au milieu de l'intervalle

$[T_i, T_{i-1}[$

Notons que cet ordre assure l'emboîtement du train binaire généré. Un codage arithmétique de ces plans de bits est effectué. L'algorithme reprend ensuite ces deux étapes avec un seuil $T_{i+1} = \frac{T_i}{2}$. Le codage s'arrête lorsqu'une condition d'arrêt est atteinte, par exemple le budget de débit.

L'algorithme EZW génère un train binaire emboîté, permettant ainsi la transmission progressive d'une image fixe. Il est expliqué plus en détail dans [Sha93].

4.5.2.2 Algorithme SPIHT (Set Partitionning in Hierarchical Trees)

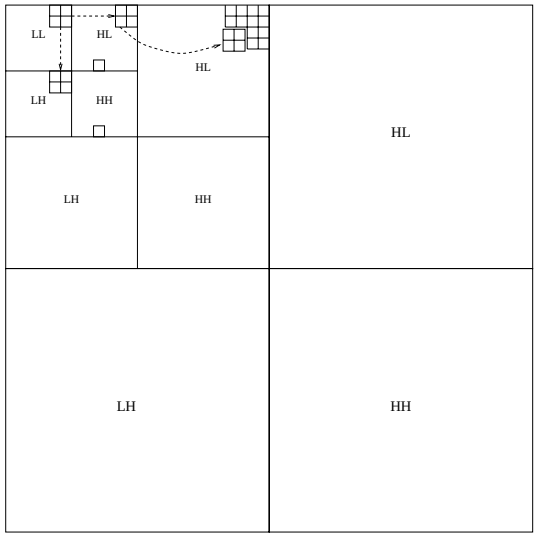

FIG. 4.12 – *Dépendances parents-enfants entre coefficients de différentes sous-bandes (SPIHT).*

L'algorithme SPIHT [SP96] est une amélioration de EZW. Du fait de sa faible complexité et de ses performances en terme de compression, il est devenu une référence. Il présente notamment les caractéristiques suivantes :

- Le protocole de descente des coefficients d'ondelettes est différent : les coefficients des sous-bandes de résolution basse sont regroupés par quatre. Un des quatre coefficients n'admet pas de descendant tandis que les trois autres possèdent chacun quatre descendants (voir figure 4.12).
- Le partitionnement en liste est modifié également. Trois listes sont considérées :
  - LIP = liste de pixels insignifiants,
  - LSP = liste de pixels significatifs,
  - LIS = liste d'ensembles insignifiants.

SPIHT consiste alors en un partitionnement récursif des ensembles de la liste LIS, pour localiser individuellement les pixels significatifs, les pixels insignifiants et les ensembles plus petits de pixels insignifiants. Puis ceux-ci sont déplacés dans les listes

respectivement appropriées: LSP, LIP et LIS. Après chaque étape de tri, des bits sont transmis, afin de raffiner les amplitudes de coefficients déjà positionnés dans LSP pour des pas de quantification supérieurs. Ce processus continue en divisant successivement le seuil de significativité par deux, jusqu'à ce que le budget de débit disponible soit atteint. L'algorithme est détaillé dans [SP96]. Cet algorithme, dépassant les performances de EZW, est devenu une référence en transmission progressive d'images fixes.

4.5.3 Approche Intra sous-bandes : EBCOT

L'idée ici consiste à considérer que les sous-bandes sont décorrélées et peuvent donc être codées de manière indépendante. Les seules dépendances que le codeur utilisera sont les corrélations présentes à l'intérieur même d'une sous-bande donnée. Deux principaux schémas de codage exploitant l'hypothèse de corrélations intra sous-bande ont été proposés EBCOT(*Embedded Block Coding with Optimized Truncation*)[Tau00] et SPECK (*Set Partition Embeddeded bloCK coder*)[IP99, PINS02]. Nous ne détaillerons ici que l'algorithme EBCOT qui offre des performances et une flexibilité supérieures à celles obtenues avec SPECK.

L'algorithme EBCOT, évolution de l'algorithme LZC (*Layered Zero Coding*)[TZ94], est l'algorithme de codage progressif adopté par le nouveau standard de codage d'images fixes JPEG-2000[ec00]. Il permet un codage emboîté, basé sur l'utilisation d'un codeur arithmétique contextuel, d'une décomposition en ondelettes discrète d'une image.

4.5.3.1 Vue générale

L'utilisation d'EBCOT est réalisée après une phase de quantification des coefficients d'ondelettes. Celle utilisée dans [Tau00] est une quantification scalaire uniforme avec zone morte (*dead-zone*).

Tout d'abord, chacune des sous-bandes est partitionnée en blocs d'échantillons B_i de taille 64×64 par exemple. Ensuite, pour chacun de ces blocs un train binaire indépendant est généré. Chaque train binaire ainsi formé est hautement scalable et peut être tronqué aux points R_i^n, correspondant respectivement à une distorsion D_i^n. La propriété intéressante de cet ensemble de points de troncature (R_i^n, D_i^n) est que la plupart de ces points sont situés sur la courbe débit-distorsion du bloc considéré.

Afin de générer un train binaire progressif pour représenter l'image entière, EBCOT organise le train binaire final en couches de qualité $\mathcal{Q}_q$. Étant donné un débit alloué à une couche donnée, le train binaire de chacun des blocs est tronqué de façon à minimiser la distorsion globale associée au décodage de la couche considérée. L'algorithme d'optimisation débit-distorsion, qui détermine la contribution de chaque bloc aux différentes couches, est appelé PCRD (*Post Compression Rate-Distortion*). L'organisation des couches de qualité est illustrée figure 4.13. En plus des portions des trains binaires de blocs concaténés, les couches de qualité $\mathcal{Q}_q$ contiennent des informations auxiliaires indiquant le point de troncature n_i associé à chaque bloc contribuant à la couche, ainsi que la longueur de train binaire $R_i^{n_i}$ correspondant. Une deuxième brique est introduite dans EBCOT, destinée à compresser ces données auxiliaires. L'ensemble

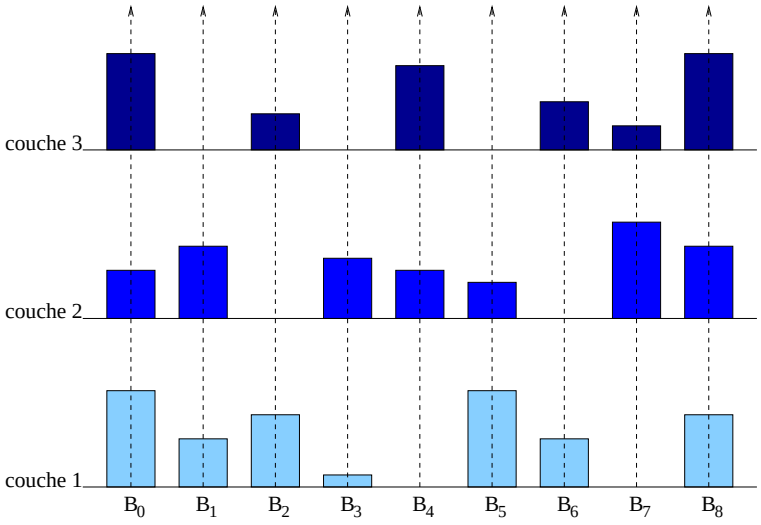

FIG. 4.13 – *Organisation des couches de qualité dans EBCOT.*

de l'algorithme EBCOT est schématisé figure 4.14.

4.5.3.2 Algorithme PCRD

L'algorithme PCRD (*Post Compression Rate-Distortion*) tente de trouver, pour chaque bloc B_i les points de troncature n_i de façon à minimiser la distorsion (additive) :

$$D = \sum_i D_i^{n_i} \qquad \text{sous la contrainte} \qquad R = \sum_i R_i^{n_i} \;\leq\; R^{max}. \tag{4.7}$$

La mesure de distorsion $D_i^{n_i}$ correspond à l'erreur quadratique moyenne pondérée par le carré de la norme L_2 des fonctions d'ondelettes de base pour la sous-bande b_i, à laquelle appartient le bloc B_i. Une approche lagrangienne est adoptée, consistant à minimiser la grandeur suivante :

$$(D(\lambda) + \lambda R(\lambda)) \;=\; \sum_i \left(D_i^{n_i^\lambda} + \lambda R_i^{n_i^\lambda} \right) \tag{4.8}$$

Pour ce faire, étant donné l'ensemble de tous les points de troncature possibles pour le bloc B_i : $j_1 < j_2 < j_3 < \ldots$, définissons les pentes distorsion-débit suivantes :

$$S_i^{j_k} = \frac{\Delta D_i^{j_k}}{\Delta R_i^{j_k}} = \frac{D_i^{j_{k-1}} - D_i^{j_k}}{R_i^{j_k} - R_i^{j_{k-1}}} \tag{4.9}$$

Un ensemble de points de troncature $j_1 < j_2 < j_3 < \ldots$ est dit admissible si la suite des pentes correspondantes $S_i^{j_1} < S_i^{j_2} < \ldots$ est strictement décroissante. L'ensemble $\mathcal{N}_i$ de points de troncature admissible le plus grand pour un bloc B_i est formé après le

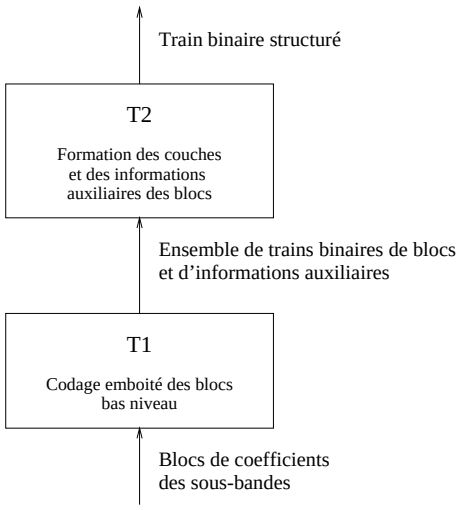

FIG. 4.14 – *Briques principales de l'algorithme EBCOT.*

codage emboîté de chaque bloc (brique T_1 de EBCOT). Ainsi, minimiser le lagrangien de l'équation 4.8 revient simplement à trouver :

$$n_i^\lambda = \max\left\{ j_k \in \mathcal{N}_i | S_i^{j_k} > \lambda \right\} \tag{4.10}$$

4.5.3.3 Codage emboîté des blocs

Nous adoptons dans la suite les notations suivantes :

- $s_i[\mathbf{k}] = s[k_1, k_2]$: séquence d'échantillons de sous-bandes appartenant au bloc B_i (k_1 et k_2 sont les positions horizontale et verticale)
- $\chi_i[\mathbf{k}] \in \{-1, 1\}$: signe de $s_i[\mathbf{k}]$
- $\nu_i[\mathbf{k}]$: amplitude des échantillons quantifiés, représentée sur M_i bits.
- $\nu_i^p[\mathbf{k}]$: $p^{\text{ème}}$ bit de la représentation binaire de $\nu_i[\mathbf{k}]$.

Significativité des sous-blocs Ce point n'apparaît plus dans le standard JPEG-2000, mais fait partie de l'algorithme EBCOT original. L'idée consiste à découper les blocs en sous blocs, par exemple de taille 16×16 et d'utiliser un codage par *quadtree* (i.e. arbre quaternaire) pour coder efficacement les sous blocs qui ne contiennent que des 0.

Primitives de codage de plans de bits La significativité $\sigma_i[\mathbf{k}]$ d'un échantillon $\mathbf{k}$ est définie comme suit : $\sigma_i[\mathbf{k}] = 0$ tant que le premier bit $\nu_i^p[\mathbf{k}]$ non nul de $\nu_i[\mathbf{k}]$ n'a pas été codé. De plus, 4 primitives sont alors utilisées pour coder les plans de bits :

- si $\sigma_i[\mathbf{k}] = 0$: utilisation d'une combinaison des primitives *ZC* (*Zero Coding*) et *RLC* (*Run-Length Coding*) pour coder si on a $\nu_i^p[\mathbf{k}] = \mathbf{1}$ ou non;

- si $\sigma_i[\mathbf{k}] = \mathbf{0}$ et $\nu_i^p[\mathbf{k}] = \mathbf{1}$: utilisation de la primitive *SC* (*Sign Coding*) pour coder le signe de $\nu_i[\mathbf{k}]$;
- si $\sigma_i[\mathbf{k}] = \mathbf{1}$: utilisation de la primitive *MR* (*Magnitude Refinement*) pour coder la valeur du nouveau bit $\nu_i^p[\mathbf{k}]$ (où p est le numéro du plan de bits courant).

Contextes associés au codage des primitives Les dépendances statistiques entre les significativités des échantillons sont capturées via la formation de trois types de contextes différents (processus illustré figure 4.15) :

- h : nombre de voisins horizontaux significatifs
- v : nombre de voisins verticaux significatifs
- d : nombre de voisins diagonaux significatifs

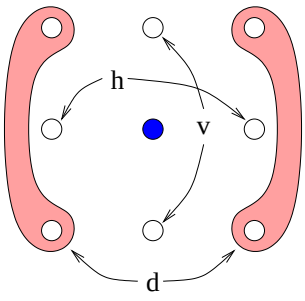

FIG. 4.15 – *Contextes formés pour le codage arithmétique des primitives.*

Plans de bits partiels et Ordre de parcours La figure 4.16(a) représente les points du plan débit-distorsion obtenus lorsqu'un codage par plans de bits est effectué. La courbe d'interpolation montre que tronquer un plan de bit donné avant son décodage complet conduit à des performances débit-distorsion sous-optimales. Une technique de *plans de bits partiels* est utilisée dans EBCOT, afin d'obtenir un train binaire finement granulaire, fournissant une multitude de points de troncature situés sur la courbe débit-distorsion de la source.

L'idée des plans de bits partiels consiste à séparer les échantillons du bloc traité en sous-ensembles statistiquement distincts. Il est alors possible de coder les échantillons par sous-ensembles, en transmettant d'abord les sous-ensembles conduisant à la baisse de distorsion la plus importante. Dans le cas de EBCOT, 3 sous-ensembles sont construits et correspondent à 3 passes de codage différentes, notées $\mathcal{P}_1^p, \ldots, \mathcal{P}_3^p$, où la fin de $\mathcal{P}_3^p$ marque le point auquel tous les échantillons ont été mis à jour dans le plan de bits p.

Brièvement, les rôles joués respectivement par chaque passe sont les suivants :

- "*Forward Significance Pass*", $\mathcal{P}_1^p$:
  - Suivi d'un parcours prédéfini : pour chaque échantillon $s_i[\mathbf{k}]$ insignifiant qui a un voisin significatif, application de la primitive ZC, accompagnée de la primitive SC si $\nu_i^p[\mathbf{k}] = \mathbf{1}$.

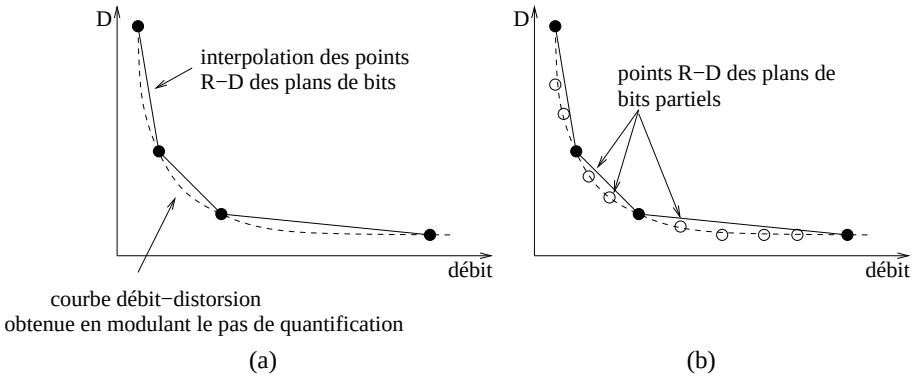

FIG. 4.16 – *Propriétés débit-distorsion (a) du codage par plans de bits réguliers et (b) du codage par plans de bits partiels.*

  - Les autres échantillons sont ignorés.
- "*Magnitude Refinement Pass*", $\mathcal{P}_2^p$:
  - Suivi du même parcours que dans la passe $\mathcal{P}_i^{p,1}$
  - Pour chaque coefficient significatif pour lequel aucune information n'a été codée dans le plan de bits p courant, utilisation de la primitive MR.
- "*Normalization Pass*", $\mathcal{P}_3^p$:
  - Suivi du même parcours
  - Tous les échantillons sont ignorés sauf les insignifiants ($\sigma_i[\mathbf{k}] = \mathbf{0}$) pour lesquels aucune information n'a encore été codée.
  - Traitement des échantillons considérés avec les primitives ZC, RLC et SC si nécessaire.

L'ordre d'apparition des différentes passes dans l'organisation du train binaire pour un bloc donné est illustré figure 4.17. On remarque que le train binaire commence par une passe de normalisation. En effet, les deux autres passes ont besoin d'une initialisation du processus pour fonctionner.

$\mathcal{P}_3^{p_{max_i}}$	$\mathcal{P}_1^{p_{max_i}-1}$	$\mathcal{P}_2^{p_{max_i}-1}$	$\mathcal{P}_3^{p_{max_i}-1}$	...	$\mathcal{P}_1^0$	$\mathcal{P}_2^0$	$\mathcal{P}_3^0$

FIG. 4.17 – *Ordre des passes dans le train binaire EBCOT correspondant à un bloc. B_i.*

4.5.3.4 Formation du train binaire final

Après la phase de formation des couches de qualité, nous avons Λ train binaires (i.e. un par couche), noté $\lambda = 1, 2, ..., \Lambda$. Chaque couche est scalable en résolution et en composantes (e.g. couleur: Y, U et V). En effet, pour chaque niveau de résolution,

$l = 0, 1, ..., L$, chaque composante c, et chaque couche, un *paquet* séparé est associé noté $K_\lambda^{l,c}$. Chaque paquet est composé d'une entête founissant un résumé des informations concernant les blocs contenus dans un paquet et d'un corps contenant les données effectives des blocs. La formation du train binaire final se fait alors par un agencement de ces paquets entre eux. Suivant l'ordre dans lequel ils sont placés dans le train binaire, on peut obtenir une scalabilité en résolution (spatiale), SNR (qualité) et en composantes.

4.5.3.5 Performances

L'algorithme de codage progressif EBCOT présente des performances au moins égales aux autres algorithmes progressifs proposés dans la littérature. Il offre par ailleurs plus de flexibilité. En effet, on peut obtenir un train binaire scalable SNR, en débit ,en résolution et en composantes. Il offre enfin des fonctionnalités d'accès aléatoire aux données dans le domaine compressé.

4.5.4 Approche hybride intra/inter sous-bande : EZBC

Un algorithme présenté récemment pour le codage d'image fixe est l'algorithme EZBC (*Embedded ZeroBlocks of wavelet coding based on Context modeling*) [HW00a]. L'idée ici est d'essayer d'exploiter les corrélations résiduelles intra et inter sous-bandes. Les auteurs ont voulu combiner dans ce codeur les avantages des deux types de codeurs présentés dans les sections précédentes : une faible complexité dûe au *Set partionning* (EZW, SPIHT) et une bonne efficacité de compression basée sur l'utilisation de contextes (EBCOT).

Une représentation des coefficients de la DWT sous forme d'un *quadtree* est, tout d'abord, réalisée pour chacune des sous-bandes indépendamment. La racine de l'arbre contient l'amplitude maximale des coefficients de la sous-bande alors que les feuilles représentent les coefficients eux-mêmes. Le but est ensuite de coder des cartes de significativité des noeuds du *quadtree* à un niveau donné, et de réitérer le processus sur le niveau inférieur de l'arbre. Le processus de construction de ces cartes de significativité se base sur l'utilisation de deux listes :

- $LIN_k[l]$: liste des noeuds insignifiants au niveau l de la sous-bande k,
- LSP_k : liste des pixels significatifs de la sous-bande k.

Il faut noter que dans cet algorithme les listes sont établies séparément pour chaque niveau de *quadtree* et chaque sous-bande.

Arrive ensuite la phase de codage des cartes ainsi que des bits de signes et de raffinement. Pour cela, les auteurs proposent non seulement d'exploiter les corrélations entre coefficients DWT mais également entre noeuds se trouvant au même niveau dans le *quadtree*. C'est dans cette phase qu'intervient le codage arithmétique contextuel. Un contexte formé des 8 noeuds appartenant au voisinage direct du noeud courant est utilisé. Notons que ces contextes sont les mêmes que ceux utilisés dans EBCOT (cf figure 4.18). Un 9ème contexte, permettant l'exploitation d'information inter-échelle, est également utilisé. Le noeud choisi est le noeud de la sous-bande parent au niveau

inférieur dans le *quadtree* (cf figure 4.18). La notion de dépendances inter sous-bandes parents-enfants est la même que celle classiquement utilisée dans EZW ou SPIHT. Le modèle est composé alors de 9 bits indiquant la significativité ou non des noeuds correspondant. Afin de réduire le nombre d'états de contextes possibles, une méthode de réduction de contexte similaire à EBCOT est utilisée. Le codage du signe est également réalisé comme celui d'EBCOT. La production de la carte de significativité utilise quant à elle l'algorithme proposé dans SPECK [IP99, PINS02].

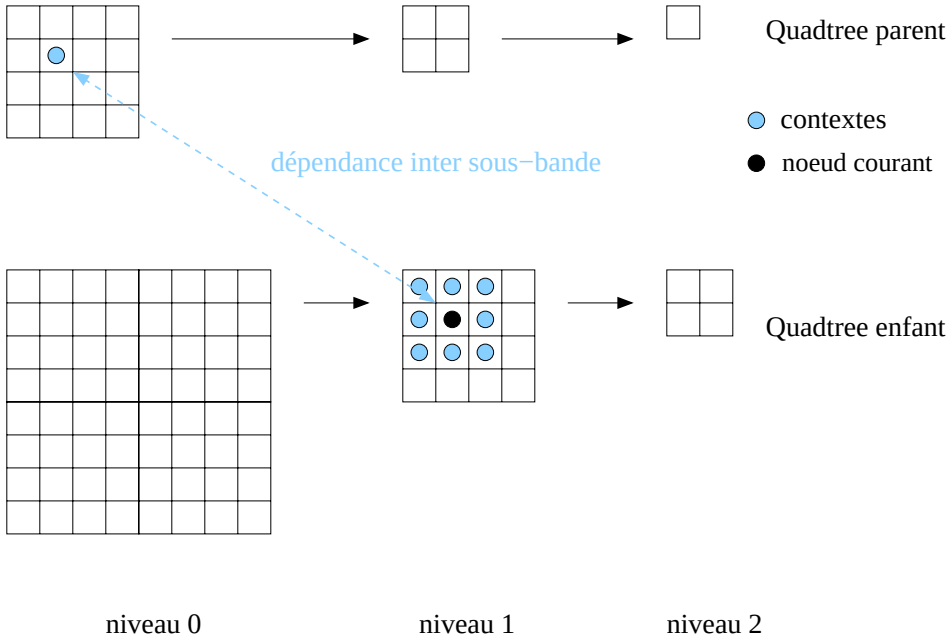

FIG. 4.18 – *Contextes utilisés pour le codage de la significativité des noeuds du* quadtree.

Les performances de l'algorithme se situent au même niveau que celles de EBCOT. Il faut toutefois noter que seule la scalabilité en résolution est permise par cet algorithme.

Remarque:
On pourrait penser que l'exploitation des dépendances à la fois intra et inter sous-bandes donne des performances nettement supérieures aux codeurs n'exploitant qu'un seul type de dépendance. Toutefois, une étude théorique proposée dans [LM01] sur les codeurs basés ondelettes montrent que le gain que l'on peut espérer en utilisant un mécanisme hybride reste peu élevé.

4.6 Transformation en ondelettes 2D+t

Les ondelettes ont fait leurs preuves dans le codage d'images fixes. L'extension de l'analyse multirésolution 2D à une suite d'images conduit à la formation de sous-bandes

spatio-temporelles 2D+t. Dans le cas de figure le plus fréquent, une telle décomposition est réalisée à l'aide d'un banc de filtres 3D séparables, construit par produits séparables de bancs de filtres 1D à deux bandes, lesquels sont réappliqués récursivement sur les sous-bandes passe-bas. Généralisant le principe de la décomposition dyadique multirésolution sur une image fixe, les sous-bandes spatio-temporelles sont chacune caractérisées par un niveau de résolution spatiale et un niveau de résolution temporelle. Leur organisation constitue un moyen naturel d'obtenir une scalabilité spatio-temporelle.

4.6.1 Principe

De manière générale, un filtrage est d'abord appliqué dans la dimension temporelle. Il transforme ainsi par exemple, à un premier niveau de résolution, un groupe de 2^p images en deux suites de 2^{p-1} images (i.e. sous-bandes temporelles), obtenues par sous-échantillonnage régulier d'une image sur deux. Le même traitement peut ensuite être repris de manière récursive sur chacune des 2^{p-1} images de la sous-bande passe-bas temporelle. Puis, chacune de ces sous-bandes peut être décorrélée spatialement selon une décomposition en ondelettes 2D. Dans la suite, le groupe d'images sur lequel est appliqué le filtrage temporel sera appelé **GOF** (*Group of Frame*). L'utilisation d'une transformée en ondelettes temporelle sur les images de la séquence est justifiée par la volonté d'exploiter les redondances temporelles entre images successives. C'est pourquoi, La mise en oeuvre du filtrage dans la direction temporelle repose le plus souvent sur une estimation/compensation de mouvement entre images du GOF.

La figure 4.19 illustre une décomposition en ondelettes 2D+t sur trois niveaux temporels et spatiaux, en considérant que la décorrélation temporelle intervient avant la décorrélation spatiale. L'ordre peut être inversé. Toutefois, lorsqu'intervient le mouvement, la décomposition en fréquences temporelles précède la décomposition spatiale. On a, en effet, remarqué dans les schémas de codage hybride prédictif classiques que la compensation en mouvement est plus efficace lorsqu'elle est appliquée dans le domaine spatial que dans le domaine transformé.

Notons, enfin, que la décomposition temporelle peut également être réalisée en utilisant un schéma de lifting. Nous verrons dans la section suivante en quoi ce type d'implémentation exhibe des propriétés intéressantes en vidéo.

4.6.2 Avantages et inconvénients

Les approches de codage en sous-bandes 2D+t présentent les avantages d'être naturellement hiérarchiques, permettant l'obtention d'un schéma de codage finement scalable et combinant scalabilités temporelle et spatiale.

De plus, l'allocation de débit entre sous-bandes spatio-temporelles orthogonales (ou quasi-orthogonales) d'un groupe d'images est relativement aisée, par rapport aux techniques mises en oeuvre dans des schémas de codage de type IPB [Ram93].

Cependant, un inconvénient majeur du filtrage temporel est lié à la latence introduite, égale à la longueur du filtre temporel. En effet, si la longueur du filtre temporel

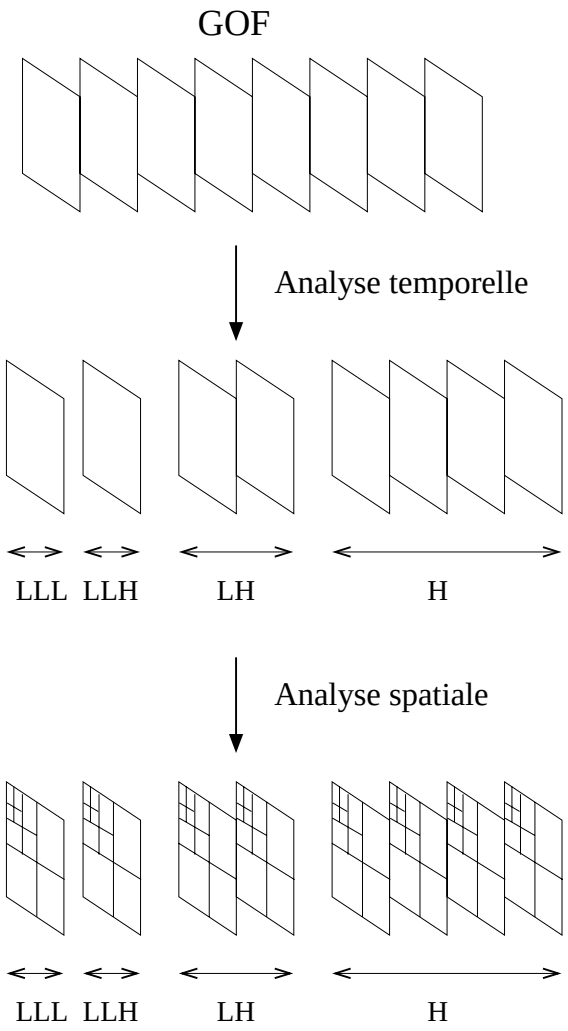

FIG. 4.19 – *Ondelettes 2D+t sur 3 niveaux (temporelles et spatiales) pour un GOF de taille 8.*

L est supérieure à 2, alors la reconstruction d'une image nécessite la réception de $\frac{L}{2}$ images successives dans le GOF traité.

4.7 Codeurs vidéo basés ondelettes 2D+t

Les techniques de compression vidéo basées sur des décompositions spatio-temporelles 2D+t sont des solutions privilégiées pour répondre à des objectifs de scalabilité fine. Les performances obtenues dans le cadre du codage d'images fixes rendent, en effet, ces solutions très attrayantes. Différentes techniques de codage vidéo en sous-bandes spatio-temporelles se sont distinguées dans la manière de prendre en compte deux aspects fondamentaux. Le premier problème à traiter, plus délicat, concerne la phase de décorrélation temporelle et notamment le choix de la transformation temporelle et de son couplage avec les éventuels modèles de mouvement. Les critères de choix sont conditionnés par un compromis entre une faible énergie résiduelle (bonne exploitation de la redondance temporelle), la fiabilité du modèle de mouvement et son coût de codage. Le second problème de ces schémas de compression concerne les techniques de codage des sous-bandes retenues. En règle générale, les techniques choisies sont des extensions directes ou non des schémas de codage finement scalable d'images fixes.

Dans ce chapitre, nous tentons tout d'abord de dresser une classification des différentes approches de décorrélation spatio-temporelle proposées dans la littérature. Cette classification est étroitement liée aux modèles de mouvement retenus ainsi qu'aux types de filtrage utilisés. Dans un second temps, nous présentons les principaux algorithmes de codage des sous-bandes spatio-temporelles proposés dans la littérature.

4.7.1 Décorrélation spatio-temporelle

Les différentes stratégies de décorrélation spatio-temporelle proposées, vont de simples techniques de filtrage direct dans l'axe temporel à des techniques plus élaborées permettant de filtrer le long des lignes de mouvement.

4.7.1.1 Filtrage direct selon l'axe temporel

Les premiers travaux réalisés en terme de codage en sous-bandes spatio-temporelles ont consisté en un filtrage temporel direct sans exploiter aucune information de mouvement. Ce type d'approche est utilisé dans [KV88, TPN94, PJF95, CP96, KP97, KXP00].

La méthode de décomposition temporelle préconisée dans [PJF95] consiste à réaliser une décomposition temporelle sur un niveau de résolution (à l'aide d'un banc de filtres QMF de longueur 10), puis une décomposition dyadique spatiale utilisant des niveaux différents selon le type de sous-bandes est réalisée. Une attention particulière est portée ici au codage des hautes fréquences temporelles contenant les informations de mouvement.

Plus récemment, Pearlman et al. [CP96, KP97, KXP00] se placent dans une optique

de codage bas débit. La décomposition spatio-temporelle est réalisée sur 16 images. Dans [CP96, KP97], le même filtre est appliqué pour la 3D-DWT dans les directions spatiales et la direction temporelle. Dans [CP96], c'est un filtre QMF à 9 bandes et dans [KP97] le filtre 9-7 biorthogonal de Daubechies [ABD92] est utilisé avec le même nombre de niveau (i.e. trois niveaux) dans les trois dimensions. Dans [KXP00] le filtre S+P [SP96] est utilisé dans la dimension temporelle (deux niveaux) alors que le filtre 9-7 de Daubechies est utilisé spatialement. L'algorithme proposé dans [KXP00] permet l'utilisation de compensation de mouvement. Toutefois aux vues de la complexité ajoutée, du coût de codage du mouvement et du faible gain obtenu pour les applications envisagées de codage bas débit, les auteurs préconisent de ne pas utiliser d'information de mouvement.

De telles approches s'avèrent efficaces sur des régions immobiles ou faiblement mobiles de la séquence. Cependant, en cas de fort mouvement, ce filtrage n'opère plus dans la direction de forte corrélation temporelle, correspondant aux lignes des mouvements présents dans la scène. L'image de moyenne temporelle devient alors floue sur les régions concernées, et l'essentiel de l'information relative à ces régions se trouve dans les sous-bandes de résolutions temporelles supérieures. De même, les contours en mouvement se retrouvent dans les sous-bandes de hautes résolutions spatiales et temporelles. Par conséquent, le pouvoir décorrélateur de la décomposition 2D+t sur le signal original décroît fortement avec l'activité de la séquence traitée. C'est pour résoudre ces problèmes que les techniques des sections suivantes ont été proposées.

4.7.1.2 Compensation de mouvement globale

Dans [TZ94] et [WXCM99] les auteurs opèrent tout d'abord une compensation de mouvement globale afin d'aligner spatialement les images sur une grille de plus forte résolution avant d'appliquer le filtrage temporel. Ce réalignement est réalisé en fonction du mouvement dominant estimé dans la scène. Ce mouvement est typiquement une translation, résultant du mouvement de caméra par rapport à la scène. Une décomposition spatiale de chaque image est alors effectuée, suivie d'un filtrage temporel appliqué aux sous-bandes spatiales. Les inconvénients de cette méthode sont la prise en compte que d'un seul type de mouvement et l'apparition de problèmes d'échantillonnage dans le cas de mouvements sous-pixeliques. Même si une telle approche s'avère efficace dans le cas de faibles mouvements, en présence de forts mouvements ou d'objets ayant des mouvements indépendants, ces performances deviennent très faibles. Enfin, il n'y a pas reconstruction parfaite.

4.7.1.3 Compensation des mouvements locaux

Afin de réduire encore les hautes fréquences temporelles, Tham [TRK97] propose de travailler sur des GOF de 4 images, et d'effectuer une compensation en mouvement basée-blocs (*block-matching*) des images du GOF par rapport à une image de référence. Ce schéma considère négligeable le problème de la réversibilité de la décomposition

en sous-bandes 2D+t sur un GOF de 4 images. En cas de forts mouvements, cette hypothèse de réversibilité n'étant plus valable, le codage d'une image d'erreur de prédiction par compensation de mouvement est proposé, au prix d'un surcoût de codage et d'une complexité accrue. Ce dernier point rend cette méthode complexe et peu attrayante. En outre, le problème inhérent à un tel type d'approche concerne les ruptures de modèles de mouvement estimés entre régions mobiles de l'image. Ceci se traduit par des recouvrements et des découvrements de régions, rendant difficile d'assurer la réversibilité de la transformation des images par compensation en mouvement. Celle-ci est en effet nécessaire à la reconstruction parfaite de la séquence.

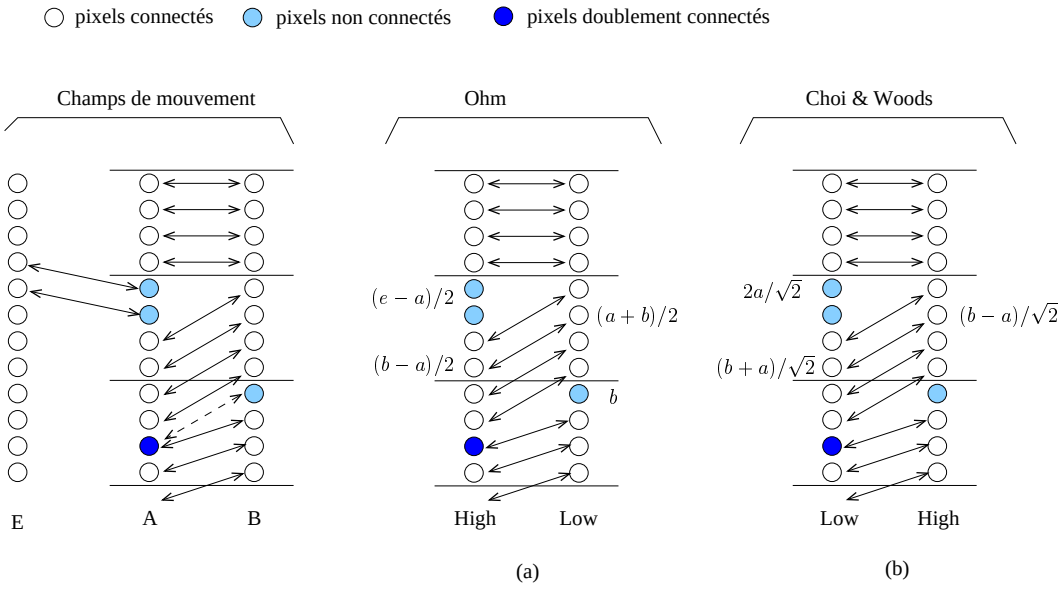

FIG. 4.20 – *Traitement des zones recouvertes et découvertes : a) Ohm b)Choi & Woods*

En se basant sur l'utilisation d'une estimation de mouvement basée-blocs [Ohm92], puis en généralisant à l'utilisation de modèles de mouvement aribitraires [Ohm94], Ohm résout le problème de la réversibilité. Pour ce faire, il décide de traiter différemment les régions couvertes, découvertes, et parfaitement mises en correspondance par la compensation en mouvement. Dans un premier temps, ce schéma a été développé avec des filtres de Haar, appliqués à des GOF de taille 2. Dans ces GOF, composés d'une image A suivie d'une image B, l'auteur distingue les régions suivantes : les régions de A et de B en parfaite correspondance à travers le mouvement estimé (**connectées**), les régions de A recouvertes dans B (**non connectées**) et les régions de B découvertes (**non connectées**). Lorsque deux pixels de l'image B sont connectés à un unique pixel de l'image A, on parle ici de pixel **doublement connecté** (cf figure 4.20), l' auteur choisit de connecter le pixel de A avec le premier pixel de B dans l'ordre lexicographique (i.e. parcours de haut en bas et de la gauche vers la droite). L'idée ensuite est de filtrer différemment les zones connectées et non connectées. Pour le cas connecté,

chaque coefficient de la sous-bande passe-haut résultante H est localisé à la position correspondante dans A et chaque coefficient de la sous-bande passe-bas L est positionné aux coordonnées spatiales pointées par le pixel correspondant dans B. Notons (d_m, d_n) le vecteur de mouvement associé au pixel situé à la localisation (m, n) dans l'image B. Ce qui donne pour les *zones connectées*:

$$\begin{array}{ll} L[m,n] = & \frac{1}{2}(B[m,n] + \bar{A}[m+d_m, n+d_n]) \\ H[m+\bar{d}_m, n+\bar{d}_n] = & \frac{1}{2}(\bar{B}[m+\bar{d}_m - d_m, n+\bar{d}_n - d_n] - A[m+\bar{d}_m, n+\bar{d}_n]) \end{array}$$

avec $\bar{d}_m$ et $\bar{d}_n$ les entiers les plus proches de d_m et d_n. $\bar{A}$ et $\bar{B}$ représentent les valeurs de pixels interpolés lorsque le mouvement est sous-pixelique. Dans le cas de pixels appartenant à une *zone découverte* de B, une basse fréquence temporelle est produite:

$$L[m,n] = \quad B[m,n].$$

Enfin, pour les pixels appartenant aux *zones recouvertes* de A une haute fréquence est produite:

$$H[m,n] = \quad \frac{1}{2}(\bar{E}[m+d_m e, n+d_n e] - A[m,n]$$

avec (m, n) les coordonnées du pixel traité dans A et $(d_m e, d_n e)$ le vecteur de mouvement entre A et E associé à ce pixel. $\bar{E}$ représente la valeur du pixel interpolé lorsque le mouvement est sous-pixelique. Ce processus de décomposition temporelle est illustré figure 4.20. La reconstruction parfaite est assurée uniquement lorsque la précision du mouvement est entière. Ces traitements peuvent être reproduits de manières récursives, sur plusieurs niveaux de résolution temporelle. Ohm a également étendu ce traitement à des filtres temporels de tailles 4 ou 10. Cependant, la méthode devient alors très complexe.

Une amélioration de ce schéma est proposée par Choi et Woods dans [CW99]. Les auteurs choisissent ici de faire correspondre les directions de l'estimation et de la compensation de mouvement. Les basses fréquences sont donc positionnées dans l'image A alors que les hautes fréquences sont placées dans B. Cette version permet ainsi de réduire les hautes fréquences lors de pixels non connectés dans B, car elle utilise le champ de mouvement A vers B. On obtient alors, pour les pixels connectés, en adoptant les mêmes notations:

$$\begin{array}{ll} L[m+\bar{d}_m, n+\bar{d}_n] = & \frac{1}{\sqrt{2}}(\bar{B}[m+\bar{d}_m - d_m, n+\bar{d}_n - d_n] + A[m+\bar{d}_m, n+\bar{d}_n]) \\ H[m,n] = & \frac{1}{\sqrt{2}}(B[m,n] - \bar{A}[m+d_m, n+d_n]) \end{array}$$

Pour les pixels non connectés :

$$\begin{array}{ll} L[m,n] = & \frac{2}{\sqrt{2}} A[m,n] \\ H[m,n] = & \frac{1}{\sqrt{2}} \left(B[m,n] - \bar{A}[m+d_m, n+d_n]\right) \end{array}$$

Le filtre temporel utilisé ici ainsi que le traitement des zones recouvertes et découvertes (cf figure 4.20) sont donc différents de celui de Ohm [Ohm94]. Le filtrage est réappliqué

récursivement sur des GOF de taille 8 et le mouvement est estimé de manière hiérarchique par blocs. Il faut noter également que, comme dans [Ohm94], la reconstruction parfaite n'est assurée que pour une précision pixelique du mouvement. Dans ce schéma, chacune des sous-bandes temporelles est ensuite décomposée dans la dimension spatiale à l'aide du banc de filtres 9-7 de Daubechies.

Afin de pallier cette limitation, un système d'analyse/synthèse spatio-temporelle à reconstruction parfaite, étendant les travaux de Choi et al., est proposé dans [HW99]. Ce nouveau schéma est réversible dans le cas d'une précision du mouvement au demi-pixel. L'idée repose sur le fait que le signal vidéo progressif peut être vu comme une sous-classe des signaux vidéo entrelacés. Cette remarque couplée à une notion de modèle global aboutit à l'utilisation d'images ou de blocs composites. Après compensation en mouvement, les blocs composant les images courante et de référence sont *replaqués* les uns sur les autres. Lorsque le mouvement est sous-pixelique le plaquage est décalée (verticalement, horizontalement ou diagonalement) et on obtient une sorte de bloc composite entrelacé de résolution supérieure. L'idée ensuite consiste à filtrer ce bloc composite dans une des dimensions spatiales avec, notamment, le banc de filtre 9-7 de Daubechies, et à former, après sous-échantilonnage par deux, une sous-bande passe-bas et une passe-haut. Ces sous-bandes ont alors la dimension des blocs originaux et sont positionnées spatialement à leur place initiale dans les images courante et de référence. Suivant l'orientation et le type des vecteurs mouvement entre les blocs, quatre classes sont définies: horizontale, verticale, diagonale et entier. L'entrelacement et le filtrage sont dépendants de la classe d'appartenance. Il est intéressant de noter que dans le cas dit "entier", cas pour lequel le mouvement a une précision entière, le filtrage temporel est un filtrage de Haar comme celui décrit plus haut [CW99]. Dans cette méthode, les pixels non connectés sont traités séparément de la même façon que dans [CW99]. L'algorithme repose sur l'utilisation de GOF de taille 16. Après décorrélation temporelle, une transformation en ondelettes 2D est appliquée utilisant là encore banc de filtre 9-7 de Daubechies. Cette technique de filtrage temporel est celle retenue dans la première version de l'algorithme MC-EZBC [HW00b].

Une extension de ces travaux est proposée dans [CW02a] pour la compression de séquences au format cinéma digital. L'idée ici consiste à appliquer d'abord un niveau de filtrage spatial sur chacune des images du GOF, puis intervient ensuite la phase de filtrage temporelle, décrite plus haut, uniquement sur les basses fréquences spatiales. Enfin, chacune des sous-bandes passe-bas temporelles est redécomposée spatialement. L'utilisation de GOF de taille adaptative est également proposée. Lorsque le pourcentage de pixels non connectés devient trop important, on décide d'écourter le GOF courant.

4.7.1.4 Filtrage basé-lifting

Les approches présentées jusqu'ici appliquent les filtres de manière convolutive. Récemment l'utilisation de filtrage temporel basé sur des schémas de *Lifting* (cf. section

4.4.3) a été proposée dans la littérature. L'implémentation lifting permet d'obtenir un mécanisme simple, rapide et flexible. Cette flexibilité offre d'ailleurs des possibilités intéressantes exploitées par certains auteurs.

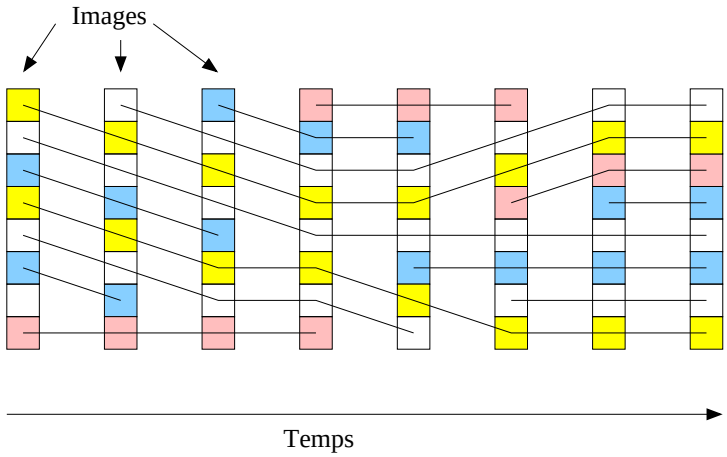

FIG. 4.21 – *Exemple de lignes de mouvement.*

Xu et al. [XLXZ00, XXLZ02] font partie des premiers auteurs à proposer l'utilisation d'un filtrage basé lifting dans la dimension temporelle. Le schéma mis en oeuvre applique le filtre 9-7 de Daubechies dans les 3 dimensions. Dans ce schéma, la notion de GOF n'est plus considérée. En effet, le filtrage d'un GOF de taille finie suppose l'extension de signal sur les bords de ce GOF. Cependant, ces extensions rendent la transformation temporelle non orthogonale et induisent des artefacts visuels non souhaitables pour la vidéo. C'est pourquoi, les auteurs proposent ici un filtrage temporel glissant ou à la volée permettant l'évitement de ces effets de bords. Il est nécessaire pour cela d'avoir un buffer permettant de garder en mémoire plusieurs images. Pour appliquer 3 niveaux de décompositions, 29 images doivent être mises dans ce buffer. Ce mécanisme de filtrage est également utilisé dans [XXLZ01], dans lequel la prise en compte des zones d'occlusions sont considérées. L'idée ici est de considérer ce que l'on appelle des *lignes de mouvement*. On connecte entre eux les pixels d'images adjacentes (via l'information de mouvement) pour former ces lignes de mouvements. Ces lignes sont en quelque sorte une extension directe des lignes de mouvement de taille deux, utilisées dans [Ohm94, CW99], à des supports potentiellement plus longs. Elles sont formées via les notions de zones connectées et non connectées exhibées par les champs de mouvement (cf figure 4.21). Une estimation de mouvement basée-blocs a été retenue ici. Une fois les lignes construites le filtrage temporel décrit plus haut est appliqué sur chacune d'elles séparément. Il faut remarquer que ces lignes ont des longueurs très hétérogènes dépendant de la mobilité des zones concernées. Une extension de ce mécanisme au codage basé-objet est également proposée.

Une formulation lifting de la décomposition temporelle proposée par Choi et Woods [CW99] est présentée dans [PPB01]. Les auteurs soulignent le fait que l'intégration d'estimation et de compensation de mouvement lors de la transformation temporelle mène naturellement à des opérateurs de *prédiction* (P) et de *mise à jour* (U) non linéaires. Ils montrent, par exemple, que l'opérateur P correspond à un opérateur de Compensation de mouvement ($\mathcal{C}$) suivi le cas échéant (mouvement sous-pixelique) d'un opérateur d'Interpolation ($\mathcal{I}$). L'opérateur U correspond également à une compensation de mouvement ($\bar{\mathcal{C}}$), avec les mêmes vecteurs que pour P mais avec un signe négatif, suivie d'une interpolation ($\mathcal{I}$). On adoptera ici la même notation que celle utilisée dans la description faite plus haut pour Choi et Woods. On notera de plus (p, q) la position $(m + \bar{d}_m, n + \bar{d}_n)$. En adoptant ces notations, l'analyse temporelle version lifting des pixels connectés devient :

$$\begin{aligned} H[m,n] = &\quad \tfrac{1}{\sqrt{2}}(B[m,n] - \mathcal{I}\{\mathcal{C}A[m,n]\}) \\ L[p,q] = &\quad \mathcal{I}\{\bar{\mathcal{C}}\{H[p,q]\}\} + \sqrt{2}A[p,q]. \end{aligned}$$

Une nouvelle méthode de gestion des pixels doublement connectés (cf figure 4.20), permettant l'optimisation du choix des pixels à connecter est ensuite présentée. Le choix se base alors sur deux critères : 1) minimisation de la DFD entre $A[p, q]$ et son correspondant dans B, 2) plus petit déplacement (i.e. vecteur mouvement). Aucune information supplémentaire n'est nécessaire pour la phase de décodage et de reconstruction. Un mécanisme de recouvrement de blocs intégré dans l'étape de filtrage permettant une réduction des effets blocs est également proposé. Basé sur la formulation lifting de la décomposition temporelle fournie, les auteurs proposent, plus récemment, plusieurs améliorations fondées sur la modification de l'opérateur de mise à jour (U). Ainsi, dans [ZPPPH02] une extension de ce schéma de filtrage temporel basé-lifting à des filtres plus longs est présentée. Dans ce schéma proposé pour un filtre 5-3, il est nécessaire de pouvoir prédire l'image X_{2t+1} à partir de X_{2t} (estimation de mouvement avant) et de X_{2t+2} (estimation de mouvement arrière). L'algorithme repose sur l'utilisation d'un opérateur de mise à jour U *adaptatif* permettant la gestion des pixels multiplement connectés.

Une approche différente est celle proposée par Secker et al. dans [ST01]. L'idée consiste notamment via les propriétés permises par les implémentations lifting à contourner le problème des pixels connectés et non connectés. Pour cela, deux champs de mouvement, avant et arrière, entre deux images consécutives à filtrer sont nécessaires. Dans les exemples de filtres décrits, seule la compensation en mouvement des images voisines de l'image courante sont nécessaires au moment du filtrage. L'utilisation de ce mécanisme est proposé pour les filtres de Haar et le filtre 5-3 biorthogonal classique; il peut-être étendu pour l'utilisation d'autres filtres. Nous donnons, tout d'abord, l'exemple dans le cas des filtres de Haar. Rappelons que la version lifting des filtres de Haar est donnée par:

$$\begin{aligned} H[m,n] = &\quad \tfrac{1}{2}(X_2[m,n] - X_1[m,n]) \\ L[m,n] = &\quad X_1[m,n] + H[m,n] \end{aligned}$$

avec L et H respectivement les filtres passe-bas et passe-haut, X_1 et X_2 deux images successives. La transformée temporelle compensée en mouvement proposée est alors:

$$\begin{aligned} H[m,n] = &\ \tfrac{1}{2}(X_2[m,n] - W_{1\to 2}(X_1)[m,n]) \\ L[m,n] = &\ X_1[m,n] + W_{2\to 1}(H)[m,n] \end{aligned}$$

avec $W_{i\to j}$ la compensation en mouvement de i vers j. On peut noter que le passe-haut n'est rien d'autre que le résidu de la compensation en mouvement, il ne dépend que de la validité du modèle de mouvement. Le résidu est d'autant plus faible que le modèle de mouvement est bon. Au dire des auteurs, la technique est meilleure avec des filtres plus longs. Des tests ont d'ailleurs été effectués avec un filtre 5/3 biorthogonal. Dans ce cas, la transformation temporelle est alors de la forme:

$$\begin{aligned} H_k[m,n] = &\ X_{2k+1}[m,n] - \tfrac{1}{2}(W_{2k\to 2k+1}(X_{2k})[m,n] + W_{2k+2\to 2k+1}(X_{2k+2})[m,n]) \\ L_k[m,n] = &\ X_{2k}[m,n] + \tfrac{1}{4}(W_{2k-1\to 2k}(H_{k-1})[m,n] + W_{2k+1\to 2k}(H_k)[m,n]) \end{aligned} \tag{4.11}$$

Les basses et hautes fréquences sont calculées sur les grilles d'échantillonnage de chaque image. En effet, à chaque étape, on compense par rapport à l'image qui est en cours de calcul. A l'étape k, le passe-haut calcule ses coefficients sur la grille de l'image $2k+1$, le passe-bas sur la grille de $2k$. A la reconstruction, l'inversibilité du schéma lifting permet de reconstruire les images sur leur propre grille d'échantillonnage. Ceci permet d'éliminer les problèmes d'échantillonnage qui apparaissaient dans les autres méthodes, tout en bénéficiant des avantages de la transformée lifting (inversibilité simple du schéma, gestion de l'erreur de reconstruction). L'inconvénient majeur est le coût de codage élevé du mouvement qui augmente avec la longueur du filtre. De plus, les performances sont dépendantes de la qualité de précision du modèle de mouvement. Enfin, la complexité opératoire du schéma n'est pas négligeable.

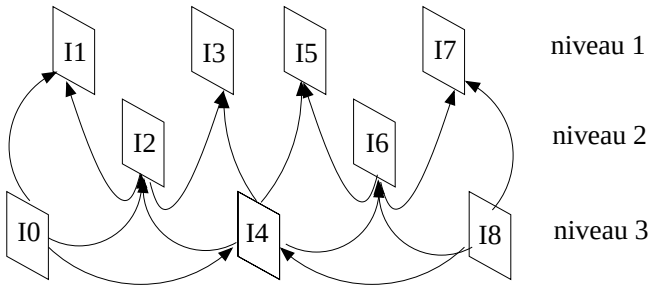

FIG. 4.22 – *Filtrage temporel dans l'approche MCLIFT.*

Une autre méthode utilisant moins d'information de mouvement et effectuant une transformée temporelle version lifting est présentée dans [LLL+01]. Dans cette méthode, appelée **MCLIFT**, il est en effet nécessaire, comme dans l'approche [PPB01], de pouvoir prédire l'image X_{2t+1} à partir de X_{2t} (estimation de mouvement avant) et de

X_{2t+2} (estimation de mouvement arrière). La transformation peut-être vue comme une décomposition en ondelettes à trois niveaux réalisée avec un filtre 5-3 *tronqué*. Dans le filtrage lifting 5-3 tronqué seule l'étape de production de hautes fréquences est réalisée (cf. équation (4.11)), le deuxième étage d'opération lifting (passe-bas) n'est pas appliqué. L'article précise que ce filtre donne de meilleurs résultats que des filtres comme le 5-3 ou le 9-7 classiques. La figure illustre le principe de la transformation sur un GOF de 9 images $(I_0, .., I_8)$. Les images I_0 et I_8 ne sont pas décomposées temporellement. Elles servent de références aux images internes qui, elles, subissent les étapes de lifting. Notons que ce sont les images originales qui servent de références plutôt que les images codées-décodées. Les auteurs montrent que la structure de codage des images obtenues est proche d'une structure IBBBPBBB d'un codeur MPEG. L'avantage de cette transformée en ondelettes version lifting sur le codeur MPEG est qu'elle effectue des prédictions entre images plus proches que ne le fait MPEG. Cette prédiction à partir d'images plus proches permet une meilleure décomposition temporelle. Classiquement chacune des sous-bandes temporelles obtenues est décomposée spatialement à l'aide d'un filtre 9-7 de Daubechies.

Différentes extensions des travaux [HW99, HW00b] ont été proposées notamment au sein de groupes de travail MPEG. Ainsi, [WC02] et [OHR02] proposent simultanément l'utilisation d'un nouveau schéma de filtrage lifting permettant notamment la reconstruction parfaite en présence de mouvement ayant une précision sous-pixelique inférieure au demi-pixel. Ohm [OHR02] fournit également une méthode de réduction des distorsions géométriques, induites par l'utilisation de modèles de mouvement basés-blocs, dans les basses fréquences. Pour ce faire, un traitement particulier des *zones de transition* entre pixels connectés et non-connectés est présenté. Cette méthode s'apparente aux algorithmes de *deblocking filter* mais est intégrée dans la phase de filtrage. La possibilité d'intégrer des sortes de "macroblocs" Intra est également permise par l'approche présentée. Enfin, une extension à l'utilisation de filtres longs implémentés en lifting est également proposée. Ces différents mécanismes ont été intégrés dans les versions suivantes de l'algorithme MC-EZBC présentées dans [WCH02, CW02b] par l'intermédiaire notamment des mécanismes dîts de *mode switching*.

4.7.2 Codage des sous-bandes spatio-temporelles

Parmi les différentes approches de codage des sous-bandes spatio-temporelles proposées, on distingue les approches basées sur des mécanismes de prédiction utilisés dans les schémas prédictifs classiques, des approches exploitant pleinement les propriétés de scalabilité naturelle permises par les transformations en ondelettes 2D+t.

4.7.3 Codage par prédiction temporelle

Les premières approches de codage des sous-bandes spatio-temporelles se fondent sur un codage DPCM temporel[PJF95, TPN94, CW99]. Dans ce cas, les sous-bandes passe-bas temporelles sont prédites à partir des sous-bandes correspondantes dans le

GOF précèdent. Les sous-bandes passe-haut peuvent être codées par quantification vectorielle [PJF95, TPN94]. L'algorithme décrit dans [CW99] met en oeuvre, quant à lui, une allocation de débit optimisée au sens débit-distorsion partageant le budget de bits disponible entre données de mouvement et codage des échantillons des sous-bandes spatio-temporelles.

Ces approches peuvent s'avérer complexes et peu efficaces. Une alternative, plus simple et très communément adoptée dans la littérature, consiste à étendre les algorithmes de compression d'images fixes basés sur une décomposition en ondelettes 2D au cas 2D+temps. Dans la section suivante nous présentons brièvement les principales approches proposées dans la littérature.

4.7.3.1 Extension des algorithmes 2D à la dimension temporelle

La première approche de codage scalable en débit basée sur l'utilisation de sous-bande 2D+t est celle proposée dans [TZ94]. Le codeur présenté, appelé **LZC** (*Layered Zero Coding*), exploite de manière efficace les corrélations existantes entre les coefficients d'une même sous-bande. Ces dernières sont codées par plans de bits. Ce codeur présenté alors pour le codage 2D+t offre alors les meilleures performances en terme de compression de toute la littérature (i.e. supérieure également aux codeurs d'images fixes).

L'algorithme bidimensionnel EZW a été étendu dans [CP96] à une version appelée **3D-IEZW** (I pour *Improved*) prenant en compte la troisième dimension. Une autre approche modifiant EZW est également proposée dans [LWCP96] pour la compression d'images volumétriques. Dans **Tri-Zerotree** [TRK97] Tham étend également EZW à une troisième dimension dans le cadre d'applications de compression bas débit.

L'algorithme SPIHT est également étendu à une décomposition multirésolution spatio-temporelle d'une séquence dans **3D-SPIHT** [KP97]. Le codage est maintenant effectué le long d'arbres d'orientations spatio-temporels. Ces structures d'arbres hiérarchiques 3D sont utilisées afin de permettre une représentation et un encodage efficaces des coefficients insignifiants. Le train binaire ainsi formé est scalable en débit et totalement emboîté. Plusieurs politiques d'agencement des composantes couleurs dans le train binaire sont également proposées. Un algorithme permettant une exploitation des corrélations entre composantes de couleurs Y,U et V, avec une version modifiée de 3D-SPIHT, est proposé dans [PPBB00]. Plus récemment, dans [BBFPP01], le codeur 3D-SPIHT a également été modifié dans le but d'obtenir un codeur pleinement scalable (spatialement, temporellement, SNR) nommé FSZ (*Fully Scalable Zerotree coder*).

Dans [XLZ00, XXLZ01], une version modifiée de EBCOT étendue à la dimension temporelle est proposée. Cet algorithme est appelé **3D-ESCOT** (*Three-Dimensional Embedded Subband Coding with Optimal Truncation*). Une des principales différences par rapport à EBCOT réside dans le fait que, comme son nom l'indique, chaque sous-bande est codée indépendamment. Un train binaire est donc formé pour une sous-bande

donnée et non pour un sous-bloc de la sous-bande. La troisième dimension est, quant à elle, intégrée dans le schéma de codage par l'ajout de contextes temporels pour le codage arithmétique des sous-bandes spatio-temporelles. L'ensemble de nouveaux contextes représentent tous les coefficients voisins distant de un pixel avec le pixel courant. A cet ensemble, il faut toutefois retirer les voisins dans les dimensions diagonales des images précédente et suivante.

L'algorithme EZBC a également été étendu pour le codage de sous-bandes issues d'une décomposition multirésolution spatio-temporelle. Ce nouvel algorithme appelé **3D-EZBC** est présenté dans [HW00b]. Dans cet algorithme, chacune des images obtenues est considérée comme "une" sous-bande temporelle qui elle même sera décomposée spatialement. L'extension à la troisième dimension n'est pas véritablement réalisée; c'est-à-dire qu'aucun contexte temporel n'a été rajouté. La formation du train binaire est ensuite réalisée en parcourant dans un ordre pré-établi chacune des sous-bandes (spatio-)temporelles résultantes. Dans une telle approche l'ordre de parcours dans la dimension temporelle a une forte importance sur le résultat final. C'est pourquoi dans [CW02a] les auteurs proposent un ré-ordonnancement des sous-bandes avant codage de manière à obtenir une qualité plus constante sur le GOF.

4.8 conclusion

Dans ce chapitre, nous avons proposé un état de l'art portant sur les schémas de codage scalable proposés dans la littérature. Après avoir souligné les limites des approches scalables présentes dans les standards vidéo actuels (H.263+ et MPEG-4), nous avons orienté notre étude vers les schémas se basant sur une représentation multirésolution de l'information basée ondelettes. Ce type d'approche étendue à la dimension temporelle fournit une voie naturelle vers l'obtention de trains binaires finement et pleinement scalables (spatialement, temporellement et en qualité). Les résultats obtenus avec les approches de compression vidéo scalable basée sur des décompositions spatio-temporelles sont très prometteurs. Un groupe de travail MPEG étudie, d'ailleurs, actuellement et activement l'intégration de ce type de schéma dans la norme. Dans le chapitre suivant, nous proposons une étude ainsi qu'un algorithme de codage finement scalable nouvelle génération basé sur une décomposition en ondelettes 2D+t.

Chapitre 5

Codage vidéo scalable à grain fin

5.1 Itroduction

Le domaine de la compression vidéo a connu, ces dernières années, de fortes évolutions menant à l'émergence d'un nombre important de standards internationaux (H.26X, MPEG-X). Malgré le nombre important de solutions, la compression reste un domaine de recherche ouvert notamment dans le cadre de transmision audiovisuelle sur différents types de canaux filaires ou non. L'arrivée de ce type d'infrastructures a également été à l'origine de nombreux travaux visant à optimiser la qualité de service de bout-en-bout. Ces travaux concernent la compression bas débit et, plus généralement, la flexibilité d'adaptation des flux compressés aux caractéristiques non stationnaires du réseau.

Face aux limites en terme de granularité des codeurs scalables standards, le concept de scalabilité à grain fin (FGS) a été introduit afin de permettre, notamment, l'adaptation de flux pré-encodés dans des scénarios de streaming. Les techniques de compression vidéo basées sur des décompositions spatio-temporelles sont des solutions privilégiées pour répondre à cet objectif. L'un des problèmes posé dans la conception de telles approches est le choix de la transformation temporelle et de son couplage avec les modèles de mouvement. Les critères de choix sont conditionnés par un compromis entre une faible énergie résiduelle (bonne exploitation de la redondance temporelle), une bonne localisation de l'énergie, la fiabilité du modèle de mouvement et son coût de codage. Le second problème posé alors est le codage des sous-bandes spatio-temporelles générées qui doit pouvoir conduire à la formation d'un train binaire finement scalable et assez flexible.

Différentes propositions de solutions reposant sur des ondelettes 2D+t, faisant suite notamment aux travaux [CW99] et [HW00b], sont actuellement à l'étude au sein d'un groupe de travail MPEG. Il faut noter, toutefois, que l'essentiel des approches proposées dans la littérature font l'hypothèse de codage haut voire très haut débit.

Dans ce chapitre, nous proposons l'architecture complète d'un nouvel algorithme de compression vidéo bas débit finement scalable, basée sur une décomposition en ondelettes 2D+t. Une étude est ensuite menée concernant la problématique de l'analyse spatio-temporelle compensée en mouvement. Différents schémas de filtrage basés sur

des filtres courts ou longs et utilisant divers modèles de mouvement sont utilisés dans cette étude. A la suite de l'étude, nous proposons deux nouveaux schémas de codage finement scalables bas débit. Les résultats sont comparés aux codeurs MPEG-4 part 2 et H.264 JM 2.1.

5.2 Cahier des charges

La conception d'un schéma de compression vidéo nécessite d'établir une sorte de cahier des charges permettant de lister les caractéristiques que doit présenter le codeur que nous cherchons à mettre au point. Celles-ci sont, dans notre cas, étroitement liées avec les notions de transmission et d'adaptation au réseau. Les caractéristiques suivantes ont été retenues :

- Une **scalabilité fine** permettant une adaptation fine du débit de la source aux contraintes du réseau, notamment dans les applications de vidéo à la demande (ou de streaming) et de visioconférence. Le codeur doit être scalable et capable d'ajuster avec finesse le débit des couches vidéo à la valeur désirée.
- Une **flexibilité** du codeur permettant de disposer des scalabilités temporelle, spatiale et SNR, et générer potentiellement un nombre élevé de couches. Ces propriétés sont notamment très intéressantes dans le cas d'application de streaming vers des clients hétérogènes.
- L'**emboîtement** du codeur assure une équité de qualité de service dans un contexte de diffusion multicast sur un réseau hétérogène. La variation de bande passante dans des couches de scalabilité données ne doit pas faire fluctuer la qualité visuelle obtenue par un récepteur abonné à un sur-ensemble de ces couches, si le débit total qu'il reçoit est inchangé. Cette propriété fait défaut aux standards en place.
- Les **performances** de compression en mode scalable doivent être supérieures à celles des codeurs standards. Le codeur produit doit également exhiber de bonnes performances face aux standards en mode non scalable.
- Les applications envisagées sont orientées **bas débit** (voire moyen). Les flux sont destinés à un panel hétérogène de clients ayant potentiellement des capacités modestes.
- La **complexité** du codeur vidéo obtenu doit être acceptable et se situer dans la même gamme que les codeurs classiques (MPEG-4 et H.263+).

Dans la section suivante, nous exposons la structure que prendra le codeur envisagé.

5.3 Description du codeur proposé

L'architecture du codeur retenue est donnée sur la figure 5.1. Dans ce schéma, nous décomposons la séquence à coder en groupe d'images ou GOF (*Group Of Frame*). Les traitements seront alors appliqués sur ces GOFs. Une phase d'estimation de mouvement est tout d'abord réalisée sur les images du GOF. L'algorithme d'estimation

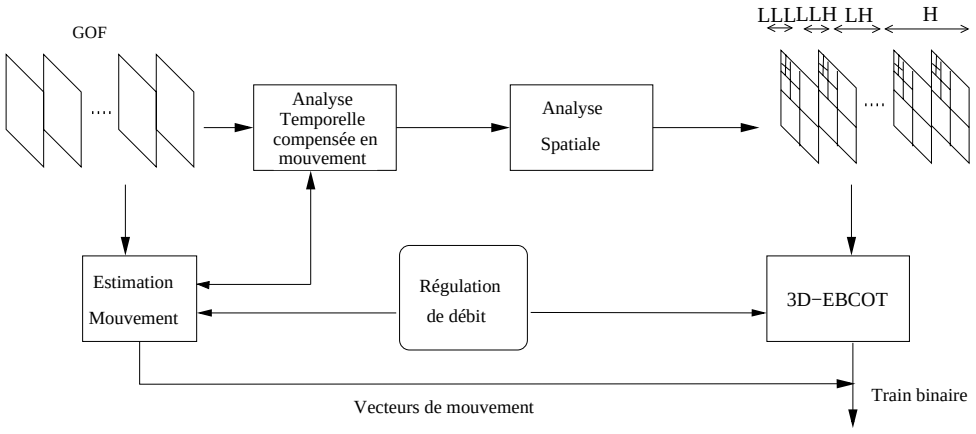

FIG. 5.1 – *Structure générale du schéma de codage proposé.*

retenu se base sur l'utilisation d'un *quadtree*[1], de vecteurs mouvement, contraint en débit représentant une pyramide de blocs de taille différente. Ce type de représentation permet de contrôler finement le débit alloué aux informations de mouvement. Un mécanisme de régulation de débit gérant la répartition de débit entre mouvement et texture est d'ailleurs présent dans le codeur. Après l'estimation de mouvement, la phase de transformation temporelle compensée en mouvement intervient. Puis, chacune des sous-bandes temporelles est ensuite décomposée avec le banc de filtres bi-orthogonal 9-7 de Daubechies. Cette transformation est réalisée à l'aide d'un schéma de lifting. On applique ensuite une quantification scalaire uniforme avec zone morte (*dead-zone*) sur l'ensemble des coefficients issus des sous-bandes spatio-temporelles. Enfin, nous avons retenu l'algorithme EBCOT pour le codage de ces sous-bandes permettant l'obtention d'un train binaire scalable à grain fin.

Dans les sections suivantes nous justifions et décrivons les choix algorithmiques réalisés concernant les différentes briques du schéma de codage.

5.3.1 Mouvement : estimation et codage

Dans la conception d'un schéma de codage vidéo la phase d'estimation de mouvement est un point primordial. De la pertinence du choix du modèle et de la qualité du mouvement obtenue dépend la qualité de la compression.

Plusieurs méthodes d'estimation de mouvement ont été testées. Celle que nous avons retenue est une **estimation hiérarchique par blocs de taille variable**. Il y a plusieurs intérêts à ce type de méthode. Tout d'abord, l'avantage d'une estimation hiérarchique réside dans sa faible complexité et sa rapidité d'exécution. Les vecteurs mouvement obtenus dans une première étape (à faible résolution) sont ensuite raffinés

1. arbre quaternaire

pour les résolutions supérieures. L'utilité des blocs de taille variable est de pouvoir adapter cette taille aux caractéristiques des mouvements locaux au sens débit-distorsion et ainsi réguler finement le débit des champs de mouvement. Cette régulation n'est généralement pas permise avec des techniques de mise en correspondances de blocs de taille fixe.

Pour parvenir à un bon compromis entre coût de transmission et qualité de compensation de mouvement, nous représentons une pyramide de blocs de taille différente dans une structure de *quadtree* contraint en débit. Ces techniques basées sur l'utilisation de *quadtree* permettent notamment d'affiner l'estimation dans les zones de l'image où le mouvement nécessite une précision spatiale accrue.

5.3.1.1 Estimation de mouvement

Le but est de construire un *quadtree* de vecteurs mouvement associés aux blocs de l'image pleine résolution. On produit tout d'abord une pyramide multirésolution spatiale pour chacune des deux images entre lesquelles le mouvement doit être estimé. On combine ensuite des opérations dites de *raffinement* des vecteurs mouvement entre niveaux de résolution spatiale et des opérations de *découpe* des blocs courants à un niveau de résolution donnée. Les premières opérations sont liées à l'estimation de mouvement hiérarchique [Bie88] alors que les suivantes sont liées à la construction des blocs de taille variable. Le processus d'estimation est illustré pour un bloc sur la figure 5.2. Nous en donnons ici l'algorithme:

1. Génération des pyramides multirésolution pour chacune des images I_t et I_{t-1}. On obtient alors pour chacune des images les différentes résolution I_t^l avec $l = 0, 1, ..., L$. La résolution L représente l'image de plus faible taille.

2. Estimation du mouvement à la résolution la plus grossière. L'estimation est réalisée ici par bloc de manière classique dans une fenêtre de recherche de dimensions (W_x^{L-1}, W_y^{L-1}) dépendantes du niveau de résolution courant. On obtient alors les vecteurs mouvement (dx^0, dy^0).

3. On découpe chaque bloc en quatre sous-blocs et on estime alors leur mouvement (dx^1, dy^1) avec $i = 0, 1, 2, 3$. Pour cela, on restreint la zone de recherche en forçant comme base de recherche le vecteur obtenu pour le bloc parent. De plus, une fenêtre de taille réduite est utilisée autour de cette position. On compare ensuite, l'erreur d'estimation moyenne des fils avec celle du bloc père. Si cette dernière est supérieure, le bloc père est réellement *découpé*.

4. On change alors de niveau de résolution. A cette étape, tous les noeuds (internes ou feuilles) de la résolution précédente sont *raffinés*. On raffine alors les vecteurs $(dx^0, dy^0), .., (dx^n, dy^n)$ en mulipliant tout d'abord par deux chacun de ceux-ci. Ce raffinement est complété par une recherche autour de ce vecteur également réalisée dans une fenêtre de taille très réduite (i.e. 1 ou 2 pixels autour).

5. On découpe chaque bloc en quatre sous-blocs et on estime alors leur mouvement (dx^{n+1}, dy^{n+1}) avec $i = 0, 1, 2, 3$. On compare ensuite, l'erreur d'estimation

moyenne des fils avec celle du bloc père. Si cette dernière est supérieure, le bloc père est réellement *découpé*.

6. Tant que la résolution courante n'est pas la pleine résolution aller à l'étape 4.
7. Le quadtree initial est ensuite formé à partir des différentes découpes réalisées sur chaque bloc de l'image à pleine résolution.

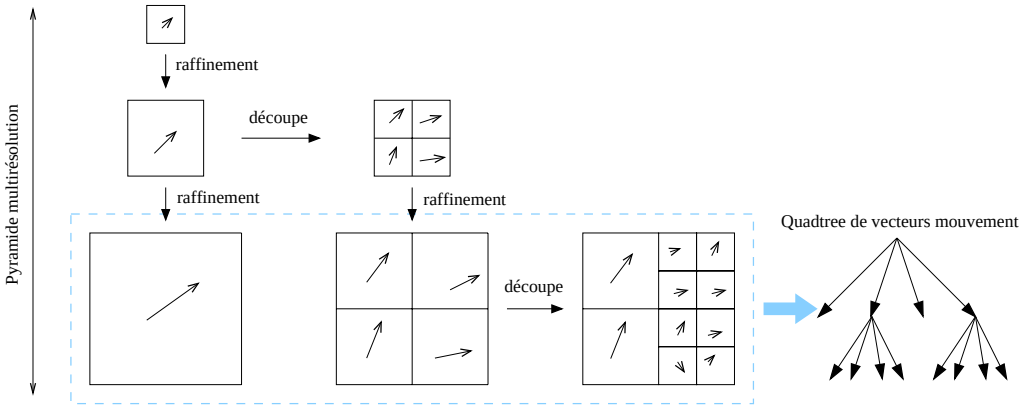

FIG. 5.2 – *Construction d'un quadtree de vecteurs mouvement pour un bloc seulement.*

L'estimation de mouvement est réalisée ici avec une précision pixelique et en prenant pour critère de mise en correspondance entre blocs la somme des différences absolues SAD. De plus, afin de favoriser le mouvement nul nous diminuons artificiellement via une simple soustraction la distorsion associée au déplacement nul. Cette technique est couramment utilisée dans les standards vidéo.

En pratique nous fixons une borne minimale et maximale concernant la taille des blocs. Celles-ci varient entre 8x8 et 64x64.

Remarque:
Afin de lisser le champ de mouvement nous utilisons une technique de recouvrement de blocs à l'estimation (OBME : *Overlapped Block Motion Estimation*). Ce recouvrement n'est cependant pas réalisé lors des phases de compensation.

5.3.1.2 Quadtree contraint en débit

Le but de la formation d'un quadtree de vecteurs de mouvement est de pouvoir adapter son débit d'encodage en fonction du besoin. Pour cela, il est nécessaire de définir une stratégie d'élagage des branches de l'arbre permettant d'obtenir le compromis débit-distorsion optimal. Il est également nécessaire d'associer à chaque noeud de l'arbre une fonction de coût. C'est en fonction de ce coût que l'algorithme d'élagage prendra ces décisions.

Classiquement, dans une approche débit-distorsion, on utilise une technique à base de multiplicateurs de Lagrange, ce qui revient à minimiser un critère du type $D + \lambda R$ pour un λ donné, avec R le débit (coût de codage) et D la distorsion associée. Une recherche sur λ est ensuite effectuée jusqu'à atteindre le débit ou la distorsion désirés. La méthode utilisée n'est pas directement basée sur les multiplicateurs de Lagrange, et évite cette phase de recherche. Nous mettons en oeuvre ici l'algorithme proposé dans [CLG89]. Cette méthode est itérative et effectue à chaque étape la division minimisant le critère $\frac{\Delta D}{\Delta R}$. Dans notre cas ΔD représente la variation de distorsion entre le bloc père et les blocs fils (ici cette variation sera toujours positive). De même ΔR est toujours positive et représente la différence de coût de codage entre le bloc père et les blocs fils. Nous décrivons dans la suite l'**algorithme d'élagage**.

A chaque noeud de l'arbre est associé un ensemble d'information servant dans le déroulement de l'algorithme. Ainsi au noeud i est associé $\{D_i, R_i, \Delta D_i, \Delta R_i, \lambda_i, \lambda min_i\}$ avec D_i la distorsion du noeud i, R_i son coût de codage, ΔD_i et ΔR_i définis plus haut, $\lambda_i = \Delta D_i / \Delta R_i$ et λmin_i la valeur de λ_j minimale parmi tous les descendants du noeud i. L'algorithme d'élagage est le suivant :

- **Initialisation**
  - Initialisation de chaque noeud avec les valeurs de débit et distorsion associée.
  - Pour chaque feuille, on a $\Delta D_i = 0$, $\Delta R_i = 0$ et $\lambda_i = \infty$.
  - Pour chaque noeud interne, on calcule en partant des feuilles les ΔD_i, ΔR_i et λ_i. Le paramètre λmin_i prend la valeur minimum entre le λ_i courant les λmin_j de ces fils.
  - Calcul du coût R_{glob} et de la distorsion D_{glob} de l'arbre entier.
- **Elagage**
  1. Si le débit R_{targ} et/ou la distorsion D_{targ} cibles ne sont pas atteints, on recherche le noeud de l'arbre qui offre un λ_i minimum. Cette recherche est facilitée par la valeur λmin_i contenue dans chaque noeud.
  2. Une fois trouvé ce noeud on supprime sa descendance.
  3. Mise à jour des paramètres ΔD_i, ΔR_i, λ_i et λmin_i du noeud courant et de tous les parents de celui-ci.
  4. Nouveau calcul de R_{glob} et D_{glob}. Si $R_{glob} > R_{targ}$ ou $D_{glob} < D_{targ}$ retourner à l'étape 1.

La distorsion D_i retenue est la SAD calculée dans la phase d'estimation de mouvement. Le coût de codage R_i est estimé en se basant sur la table de codes VLC utilisés pour le codage du mouvement dans H.263+. C'est le coût de codage (i.e. la taille du code VLC) du différentiel entre le vecteur du noeud courant et le vecteur du père qui sert pour estimer R_i. Ce coût est raffiné avec celui de la structure de l'arbre à coder. Nous détaillons dans la section suivante le codage de la structure.

5.3.1.3 Codage des informations de mouvement

Une fois l'arbre élagué, il est nécessaire de coder les informations de mouvement et de former un train binaire en conséquence. La structure du train binaire est composée de deux parties : la structure de l'arbre et les vecteurs mouvement.

Codage de la structure de l'arbre

L'idée est de coder, pour chaque noeud de l'arbre, l'information de paternité ou non. Ainsi lorsqu'un noeud est parent (i.e. le bloc est subdivisé) on code 1 alors que si c'est une feuille on code 0. Le train binaire correspondant à la structure est alors codé en parcourant le quadtree en largeur d'abord et en insérant les 1 et 0 en conséquence. En pratique, le coût peut encore être réduit par l'exploitation des informations de taille de bloc minimale et maximale. Dans ce cas, on ne commence à coder la structure qu'au niveau des blocs de taille maximale. De plus, aucun bit n'est codé pour les noeuds correspondant à des blocs de taille minimale, car ce sont obligatoirement des feuilles. La figure 5.3 montre un exemple de codage d'une structure.

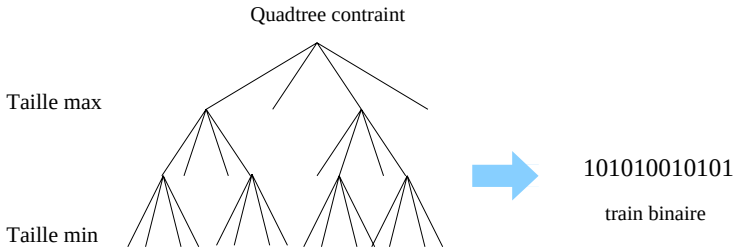

FIG. 5.3 – *Exemple de structure codée.*

Codage des vecteurs mouvement

La structure de codage admise dans la phase de construction et d'élagage du *quadtree* n'est pas exactement celle que nous utilisons. Nous utilisons une technique donnant de meilleures performances. Ceci implique que la phase d'élagage n'est plus tout à fait optimale au sens débit-distorsion. Toutefois les résultats montrent une dérive faible.

Le codage de la structure de l'arbre nous fournit une carte spatiale des blocs et non plus une structure arborescente (i.e. la structure est mise à plat nous permettant d'avoir les tailles respectives de chacun des blocs). Nous codons ensuite les vecteurs mouvements associés à ces blocs de manière prédictive. Le prédicteur utilisé est la valeur médiane des vecteurs associés aux blocs voisins (cf. figure 5.4). L'erreur de prédiction est alors codée via les codes VLC du codeur H.263+.

5.3.2 Analyse spatio-temporelle compensée en mouvement

Dans notre schéma, les phases d'analyses temporelles et spatiales sont séparées. Elles ne sont toutefois pas totalement indépendantes comme nous le verrons dans la suite.

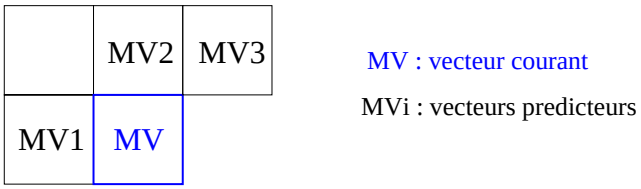

FIG. 5.4 – *Prédiction des vecteurs mouvement.*

La phase d'analyse temporelle est fortement couplée avec le modèle de mouvement utilisé. Les critères de choix de la transformation temporelle sont conditionnés par un compromis entre une faible énergie résiduelle, une bonne localisation de l'énergie, la fiabilité et le coût de codage du modèle de mouvement. Dans la section 5.4, nous proposons une étude de différentes solutions envisagées portant sur divers critères de choix comme les modèles de mouvement, les longueurs de filtres, le type de filtres, la gestion des zones connectées et non connectées et enfin l'importance de l'orthogonalité pour ce type de transformée.

Concernant la transformation en ondelettes spatiale, une implémentation lifting d'un filtre de Daubechies 9-7 est utilisée. Par défaut, 3 niveaux de décomposition sont appliqués sur chacune des sous-bandes temporelles. Nous verrons dans la suite qu'il peut être intéressant d'appliquer des niveaux de décomposition différents selon le type de sous-bandes temporelles (voire selon la composante de couleur (Y, U ou V)).

5.3.3 Codage des sous-bandes spatio-temporelles : EBCOT-3D

On a vu dans la section 4.7.3.1 du chapitre précédent que plusieurs travaux ont été menés afin d'étendre divers algorithmes de progressivité 2D à la dimension temporelle. Nous avons choisi ici d'utiliser l'algorithme EBCOT également dans la dimension temporelle plutôt que des algorithmes comme EZW ou SPIHT. Plusieurs raisons ont motivé notre choix.

5.3.3.1 Motivations

La première raison, et non la moindre, concerne les performances exhibées par l'algorithme EBCOT qui surpasse ou fait au moins aussi bien que toutes les solutions scalables proposées dans la littérature. De plus, dans la mesure où des algorithmes comme 3D-EZW ou 3D-SPIHT sont basés sur la notion de relation père-fils entre coefficients de sous-bandes de fréquences différentes, ceux-ci semblent naturellement très sensibles aux pertes de paquets. Par ailleurs, l'algorithme EBCOT ne prend pas en compte d'arbre de dépendances entre coefficients d'ondelettes de sous-bandes de fréquences distinctes. Les données compressées représentant un bloc d'une sous-bande donnée sont indépendament décodables. Cette propriété est intéressante et ouvre une voie vers une encapsulation robuste des données compressées dans des paquets pour la

transmission sur l'Internet. Enfin, la flexibilité permise par EBCOT permettant l'obtention des diverses scalabilités spatiale, SNR, en composantes, voire temporelle dans le cas de l'extension au cas 2D+t. Pour ces raisons, nous avons choisi l'algorithme EBCOT pour coder les sous-bandes 2D+t issues de l'étape de transformation.

5.3.3.2 EBCOT-3D

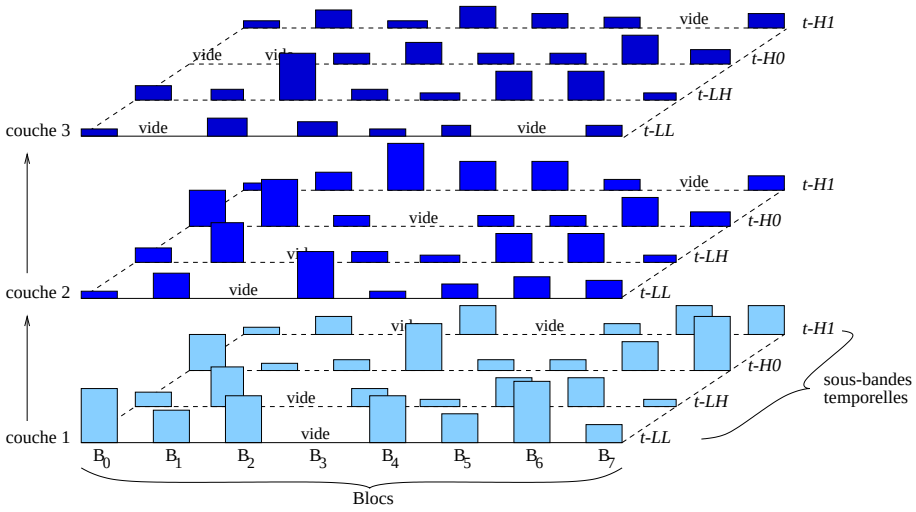

FIG. 5.5 – *Couches de qualité EBCOT-3D.*

Il n'y a pas à proprement parler d'extension directe des mécanismes du schéma EBCOT 2D à la dimension temporelle. En effet, aucun contexte intra sous-bandes temporelles n'est exploité dans la phase de codage arithmétique. A la manière de l'algorithme EZBC [HW00a], nous considérons ici chaque image comme *une* sous-bande temporelle décomposée spatialement. C'est ce groupe de sous-bandes que nous donnons à coder à EBCOT. Ainsi, la première étape de codage des blocs est identique. La seconde étape de formation des couches de qualité est étendue, afin de prendre en compte l'ensemble de tous les blocs de toutes les sous-bandes spatio-temporelles du groupe d'image traité comme illustré sur la figure 5.5. L'agencement du train binaire est ensuite dépendant des modes de scalabilité désirés.

Au sein de chacune des couches de qualité, les informations relatives à la basse fréquence temporelle viennent en premier, puis viennent les images de hautes fréquences dans l'ordre décroissant (i.e. LLL, LLH, LH, H). Il faut noter également que les composantes de couleurs sont entrelacées. Un exemple d'ordre d'apparition des sous-bandes et des composantes de couleurs est illustré sur la figure 5.6. Chacune des couches est alors scalable en résolution, en composantes et temporellement.

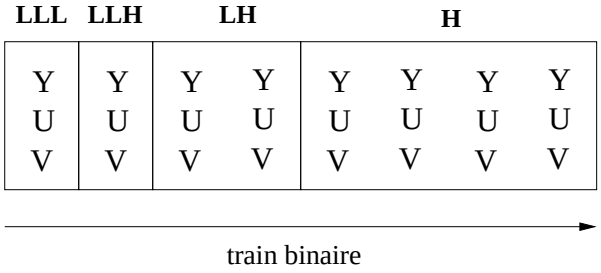

FIG. 5.6 – *Exemple d'ordonnancement des sous-bandes temporelles et des composantes de couleurs (Y,U,V) dans le train binaire (pour un GOF de taille 8).*

5.3.4 Régulation de débit

A l'initialisation, nous attribuons un budget de débit pour chaque GOF en fonction de la contrainte de débit cible globale. La contrainte par GOF R_{GOF} est alors donnée par :

$$R_{GOF} = G_{size} \times R_{target}/Fr$$

avec G_{size} le nombre d'images composant le GOF, R_{target} le débit cible en bits/s et Fr la fréquence temporelle de la source exprimée en nombre d'images/s. Le débit obtenu R_{GOF} est donc exprimé en bits.

Il s'agit ensuite de répartir ce budget entre informations de mouvement et informations de texture. Pour cela un pourcentage de débit p_{mv} est alloué au mouvement. Ce débit est donné par

$$R_{mv} = p_{mv} \times R_{GOF}.$$

Ce budget est ensuite partagé en un débit par champ de mouvement. Ce débit évolue au cours du temps en fonction du débit utilisé par les champs précédents. Une meilleure politique non utilisée ici serait de répartir le débit parmi les n champs de mouvement à coder en fonction de leur coût initial avant élagage. Ceci requiert toutefois que tous les champs de mouvement soient disponibles à $t = 0$, ce qui n'est généralement pas possible.

En pratique, le budget utilisé réellement pour le mouvement est différent de R_{mv} et est noté $\bar{R}_{mv}$. Le budget alloué à la texture est donc :

$$R_{texture} = R_{GOF} - \bar{R}_{mv}.$$

5.4 Analyse 2D+t compensée en mouvement : une étude

Dans cette section, nous réalisons une étude se basant sur trois schémas d'analyse temporelle compensée en mouvement. Le premier schéma consiste en un filtrage de Haar itéré sur trois niveaux dans lequel les zones couvertes et découvertes sont traitées

comme dans [CW99]. Le second schéma met en oeuvre, quant à lui une transformation temporelle basée lifting utilisant une compensation de mouvement avant et arrière (*forward/backward*) [ST01]. Pour ce schéma deux bancs de filtres bi-orthogonaux (5-3 et 9-7) sont étudiés et appliqués sur trois niveaux. Le dernier schéma utilise quant à lui un filtrage 5-3 tronqué compensé en mouvement sur trois niveaux [LLL+01].

Plusieurs aspects sont étudiés ici. Nous nous intéressons, tout d'abord, au compromis entre coût de codage de mouvement et énergie résiduelle dans les hautes fréquences temporelles. Nous considérons ensuite la problématique liée à l'utilisation de filtres temporels longs. Nous abordons, également, l'impact de la gestion des zones d'occlusion sur les performances de la phase d'analyse temporelle. Enfin, nous évoquons l'intérêt d'utilisation de niveaux de décomposition spatiale adaptés aux types de sous-bandes.

5.4.1 Trois modèles de mouvement

Nous pouvons classifier les différentes approches d'analyse temporelle compensée en mouvement proposées dans la littérature suivant les modèles de mouvement qu'elles utilisent. Les traitements et problématiques associés diffèrent. Nous distinguons trois types d'estimation : estimation arrière, estimation/compensation avant et arrière, et estimation avant ou arrière.

5.4.1.1 Estimation de mouvement arrière :

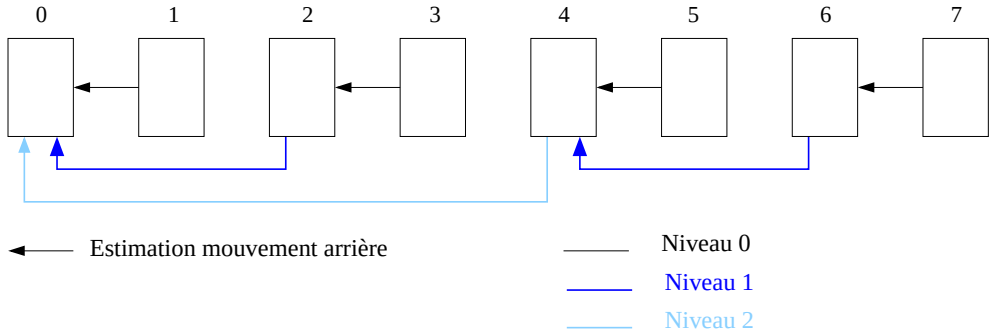

FIG. 5.7 – *Différents champs de mouvement utilisés au sein d'un GOF pour le filtre de Haar itéré sur 3 niveaux.*

L'utilisation d'estimation de mouvement dans un seul sens mène de manière inhérente aux problèmes de gestions de pixels connectés ou non connectés correspondant aux zones d'occlusion. Il est alors impératif de coupler étroitement les phases de compensation de mouvement et de filtrage si l'on veut pouvoir garantir une reconstruction parfaite. De plus, en règle générale la gestion de ce type de zones induit des restrictions quant à la taille du support du filtre et implique également l'utilisation de filtre court. La solution étudiée pour ce type de modèle de mouvement se base sur un filtre de Haar

appliqué itérativement sur des paires d'images. La figure 5.7 illustre les champs de mouvement utilisés pour l'application d'un tel schéma sur un GOF de taille 8 avec 3 niveaux de décomposition. Aux niveau 1 et 2, l'estimation et la compensation sont réalisées sur les sous-bandes basses fréquences temporelles du niveau précédent. Dans ce schéma, le nombre de champs de mouvement nécessaires à la phase de filtrage temporelle est identique aux approches prédictives classiques, c'est-à-dire 7 champs de mouvement pour un GOF de taille 8.

5.4.1.2 Estimation de mouvement avant et arrière :

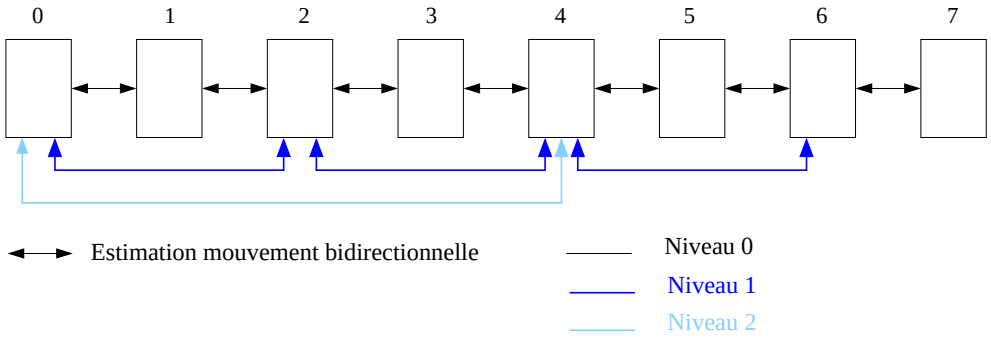

FIG. 5.8 – *Différents champs de mouvement utilisés au sein d'un GOF pour les filtres 5-3/9-7 lifting sur 3 niveaux.*

L'utilisation de paires de champs de mouvement avant et arrière, entre couple d'images adjacentes à un niveau donné, a été introduit afin de contourner les problèmes liés aux zones d'occlusion. Dans [ST01], cette façon de gérer le problème est facilitée par l'implémentation lifting du filtrage temporel. Dans ce schéma, les filtrages longs sont simplifiés car le schéma lifting réduit les traitements à de simples traitements entre images adjacentes. Deux bancs de filtres bi-orthogonaux, 5-3 et 9-7, sont étudiés ici. La figure 5.8 illustre les différents champs de mouvement nécessaires à la mise en oeuvre d' un tel schéma sur un GOF de taille 8 comme proposé dans [ST01]. Une des principales limites de ce type d'approche est le nombre de champs de mouvement impliqués dans le processus qui s'élève ici à 22 pour un GOF de taille 8.

5.4.1.3 Estimation de mouvement avant ou arrière :

Nous utilisons ici les termes d'estimation *avant ou arrière* pour exprimer le fait qu'il n'y a entre chaque couple d'images qu'un seul champ de mouvement qui est soit avant, soit arrière. Ce type de solution est intéressant puisqu'il permet de réduire de manière significative le nombre de champs de mouvement utilisés. En fait, pour chaque image d'indice impair un champ allant vers l'image précédente et un autre allant vers la suivante sont calculés. Ceci permet de capturer des redondances provenant des deux directions temporelles. Une solution utilisant ce type de compensation est étudiée via

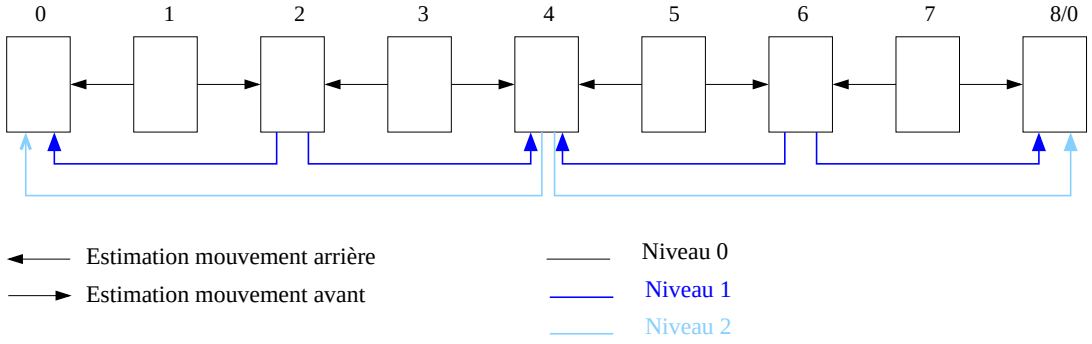

FIG. 5.9 – *Différents champs de mouvement utilisés au sein d'un GOF pour le filtre 5-3 tronqué sur 3 niveaux.*

le schéma proposé dans [LLL+01] s'appuyant sur l'utilisation d'un filtre lifting 5-3 tronqué. Dans un tel schéma, aucune basse fréquence n'est produite seules le sont les hautes fréquences. La figure illustre les champs de mouvement nécessaires à la mise en oeuvre de ce schéma sur 3 niveaux de décomposition. Le GOF utilisé ici est de taille 9 mais la dernière image sert de première image au GOF suivant, ce qui revient en pratique à coder 8 images à chaque fois. Dans ce schéma, les images extrémités sont codées en Intra. Enfin, 14 champs de mouvement sont utilisés pour une décomposition sur 3 niveaux.

5.4.2 Concentration de l'énergie / Coût de codage du mouvement

Un des points clés en terme de concentration de l'énergie, et par conséquent, de minimisation de l'énergie résiduelle dans les hautes fréquences, est le choix du modèle de mouvement retenu dans la transformation temporelle. Les trois schémas étudiés ici offrent des modèles d'estimation de mouvement différents. Nous examinons dans cette section l'impact du modèle de mouvement sur le compromis entre énergie résiduelle et le coût de codage du mouvement.

Les figures 5.10 et 5.11 illustrent les sous-bandes temporelles obtenues après filtrage en considérant les 3 approches étudiées. Nous montrons trois types de sous-bandes hautes fréquences obtenues : une sous-bande obtenue au niveau 0, une au niveau 1 et enfin celle obtenue au niveau 2. L'estimation de mouvement est réalisée par blocs en utilisant l'algorithme hiérarchique décrit dans la section 5.3.1 avec des blocs allant de la taille 8x8 à 64x64.

La figure 5.10 montre l'énergie résiduelle obtenue avec les trois méthodes dans le cas où le coût de mouvement est non contraint. La figure 5.11 illustre le cas où le mouvement est contraint à 35% du débit total fixé à 140 kbits/s.

Dans le cas non contraint on peut voir que l'approche 5-3 et 9-7 avec les champs de mouvement avant et arrière permet la meilleure réduction de l'énergie résiduelle sur les sous-bandes de niveaux 0 et 1. Les approches 5-3 tronqué et Haar viennent après.

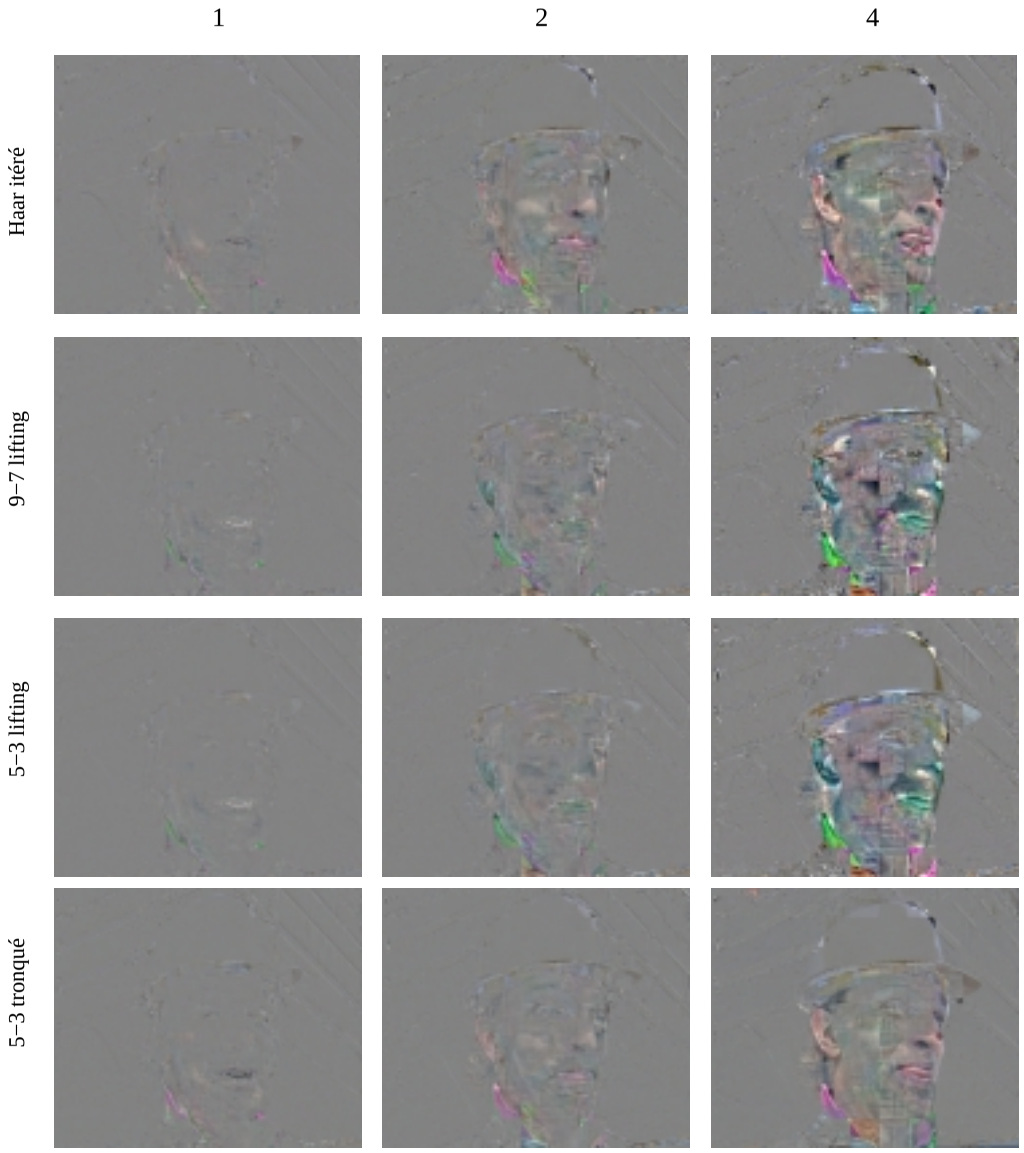

FIG. 5.10 – *Energie résiduelle dans les sous-bandes hautes fréquences temporelles situées à différents niveaux de la décomposition (niveau 0 : sous-bande 1, niveau 1: sous-bande 2 et niveau 2: sous-bande 4) lorsque le débit du mouvement est non contraint pour les approches (a) Haar itéré (b) 9-7 lifting (c) 5-3 lifting et (d) 5-3 tronqué.*

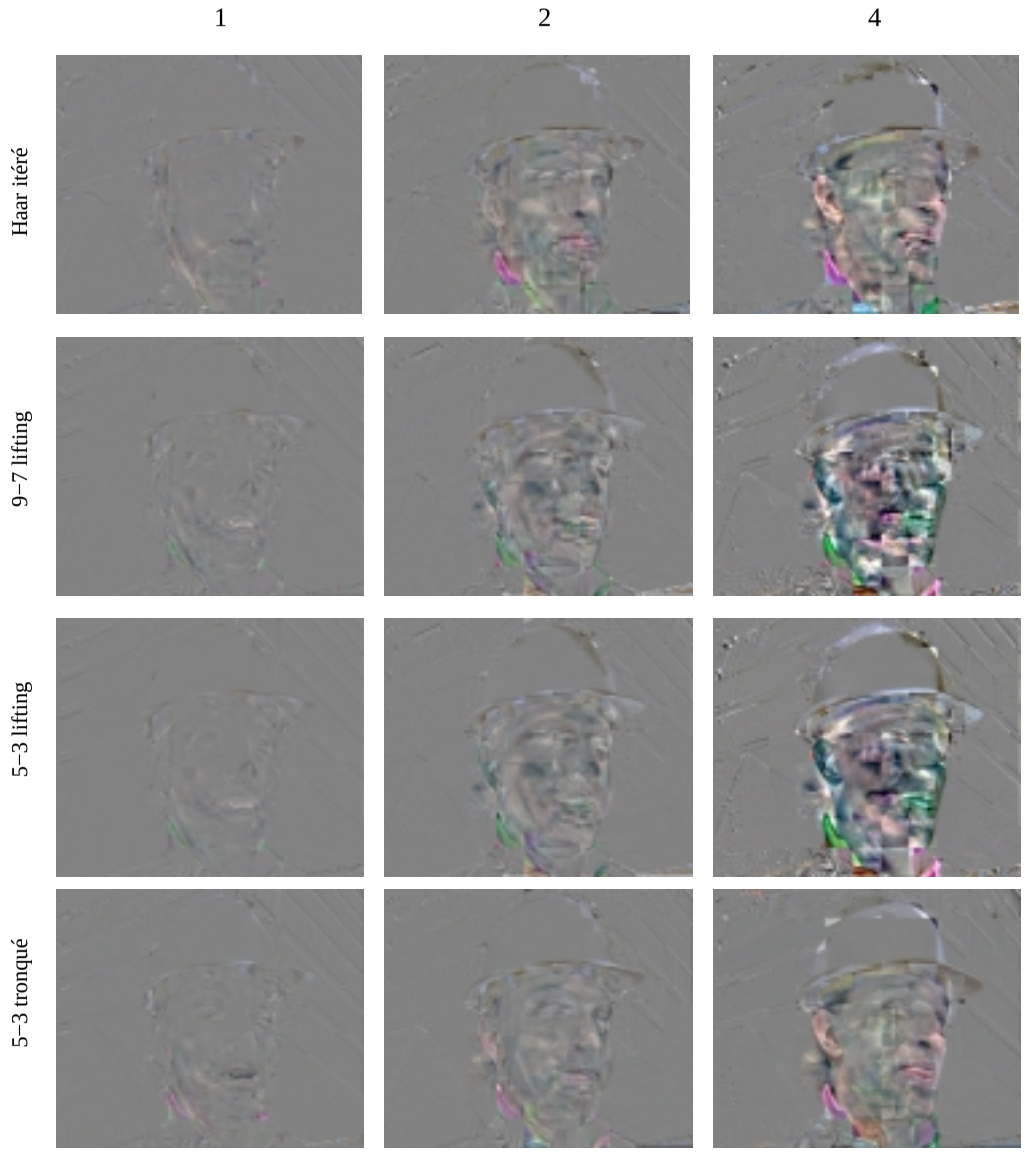

FIG. 5.11 – *Même chose que pour la figure 5.10 avec un coût de mouvement contraint à* 35% *du débit total fixé à 140 kbits/s.*

Toutefois, en ce qui concerne la sous-bande de niveau 2 les approches 5-3 et 9-7 sont déjà plus limitées. Lorsque la contrainte de débit sur le mouvement est activée, les approches 5-3 et 9-7 lifting offrent un compromis qui s'avère non satisfaisant à bas débit. Il faut noter que dans les deux cas la sous-bande haute fréquence de niveau 2 est de très mauvaise qualité . On remarque, de plus, que les performances sont moins bonnes avec le banc de filtres 9-7. Les deux autres approches offrent des performances en terme de minimisation de l'énergie résiduelle relativement comparables et fournissent le meilleur compromis. Ces performances sont en accord direct avec les résultats obtenus et donnés dans la section suivante.

5.4.3 Performances

Nous avons expérimenté ces 3 approches, dans notre schéma de codage vidéo présenté plus haut, sur diverses séquences au format CIF PAL (352×288) avec une fréquence de 15 Hz. Nous avons retenu les séquences suivantes : *Hall*, *Foreman*, *Mother and daughter*, *Coastguard*, *Mobile and calendar*, *Flower garden* et *Tempete*. Toutes les séquences décodées possèdent 85 images. Les expérimentations ont été réalisées pour 5 débits cibles : 100, 140, 200, 256 et 500 kbits/s.

Les tableaux 5.1, 5.2, 5.3 et 5.4 donnent les résultats obtenus (i.e. PSNR moyen) pour les 7 séquences et les 5 débits avec les 3 méthodes. Nous donnons ici quelques courbes afin d'illustrer les PSNR obtenus au cours du temps pour certaines séquences avec les 4 approches. Nous illustrons ici les séquences *Coastguard* à 100 kbits/s (figure 5.12), *Tempete* à 140 kbits/s (figure 5.13), *Foreman* à 200 kbits/s (figure 5.14), *Flower garden* à 256 kbits/s 5.15 et *Mobile and calendar* à 500 kbits/s (figure 5.16).

On voit clairement que les approches 5-3 et 9-7, basées sur l'utilisation d'un champ avant et arrière entre chaque paire d'images adjacentes, ont des performances très faibles. En plus d'être faibles les PSNR obtenus sont très instables. On observe de fortes chutes de plusieurs dB sur chaque GOF. Ces chutes peuvent s'expliquer à la fois par la mauvaise réduction de l'énergie résiduelle observée dans les sous-bandes de la section précédentes. Plusieurs phénomènes influent sur ces variations de qualité. Dans la suite, nous proposons une discussion permettant d'expliquer les résultats obtenus.

Les résultats montrent des performances relativement comparables pour les approches Haar et 5-3 tronqué. On observe toutefois certaines différences de performances plus ou moins importantes suivant les séquences comme le montrent les courbes obtenues. Ces deux approches surpassent très nettement les approches basées sur les bancs de filtres 5-3 et 9-7 lifting qui ne sont manifestement pas réalistes dans un scénario bas débit.

5.4.4 Discussions

Dans cette section, nous examinons les différents points critiques liés à l'analyse temporelle compensée en mouvement. La discussion menée ici a pour but d'expliquer les résultats, mais aussi et surtout, de servir de base de travail dans la conception de futurs schémas de codage 2D+t. Nous soulignons les faiblesses et les avantages des

	Haar				
Débit (kbits/s)	100	140	200	256	500
Hall	32.13	34.15	36.16	37.53	39.81
Foreman	29.72	30.99	31.97	32.79	35.45
Mother	35.90	37.25	38.64	39.67	42.12
Coastguard	26.04	26.86	27.77	28.52	30.35
Mobile	20.33	21.20	22.16	22.80	24.71
Flower	21.65	22.62	23.55	24.42	26.45
Tempete	24.61	25.50	26.50	27.23	29.11

TAB. 5.1 – *PSNR moyen obtenu pour les 7 séquences aux débits de 100, 140, 200, 256 et 500 kbits/s avec la méthode de Haar (mouvement arrière).*

	5-3 tronqué				
Débit (kbits/s)	100	140	200	256	500
Hall	28.83	30.80	33.16	34.59	38.27
Foreman	29.15	30.55	32.01	32.97	35.70
Mother	34.52	35.90	37.49	38.77	41.77
Coastguard	25.52	26.37	27.35	28.09	30.17
Mobile	19.80	20.61	21.70	22.54	25.21
Flower	20.85	21.67	22.83	23.54	26.11
Tempete	23.85	24.80	26.02	26.80	29.43

TAB. 5.2 – *PSNR moyen obtenu pour les 7 séquences aux débits de 100, 140, 200, 256 et 500 kbits/s avec la méthode 5-3 tronqué (mouvement avant ou arrière).*

	5-3 lifting				
Débit (kbits/s)	100	140	200	256	500
Hall	29.86	31.81	33.76	35.06	37.76
Foreman	27.10	28.54	29.84	30.63	33.28
Mother	34.28	35.47	36.89	37.85	40.37
Coastguard	24.14	24.70	25.45	25.92	27.70
Mobile	18.26	18.99	20.07	21.01	23.51
Flower	19.24	20.14	21.20	21.94	24.39
Tempete	22.42	23.28	24.37	25.21	27.66

TAB. 5.3 – *PSNR moyen obtenu pour les 7 séquences aux débits de 100, 140, 200, 256 et 500 kbits/s avec la méthode 5-3 lifting (mouvement avant et arrière).*

	9-7 lifting				
Débit (kbits/s)	100	140	200	256	500
Hall	28.61	30.32	32.08	33.19	35.90
Foreman	23.50	24.79	25.88	26.54	28.88
Mother	32.18	33.36	34.60	35.45	37.60
Coastguard	22.01	22.22	22.56	22.88	23.80
Mobile	16.02	16.79	17.83	18.54	20.81
Flower	17.33	18.31	19.12	19.63	21.88
Tempete	19.80	20.57	21.73	22.43	24.88

TAB. 5.4 – *PSNR moyen obtenu pour les 7 séquences aux débits de 100, 140, 200, 256 et 500 kbits/s avec la méthode 9-7 lifting (mouvement avant et arrière).*

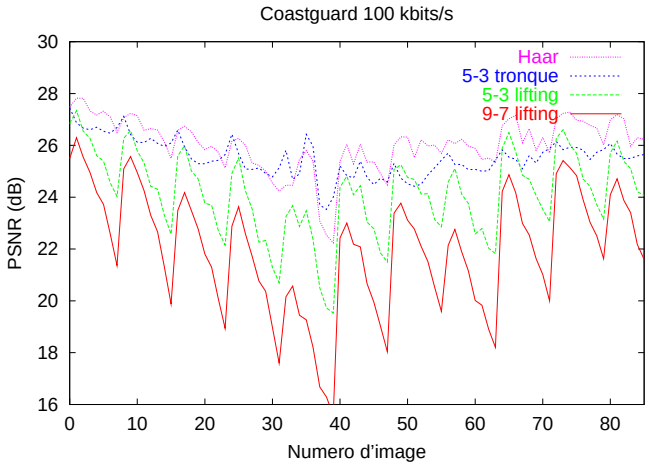

FIG. 5.12 – *Séquence* Coastguard *à 100 kbits/s avec les 4 approches : Haar itéré, 5-3 tronqué, 5-3 lifting et 9-7 lifting.*

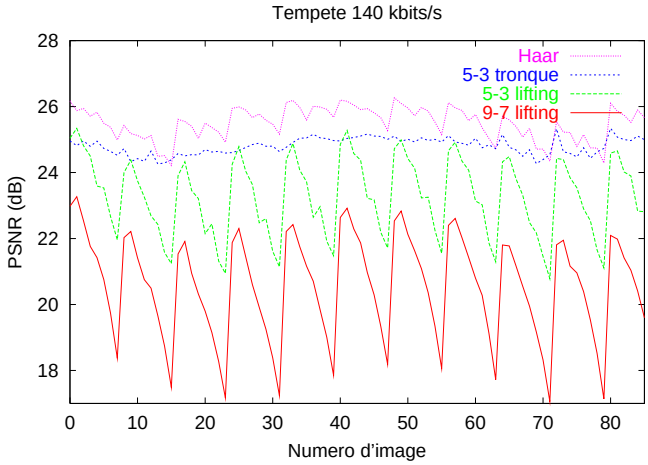

FIG. 5.13 – *Séquence* Tempete *à 140 kbits/s avec les 4 approches : Haar itéré, 5-3 tronqué, 5-3 lifting et 9-7 lifting.*

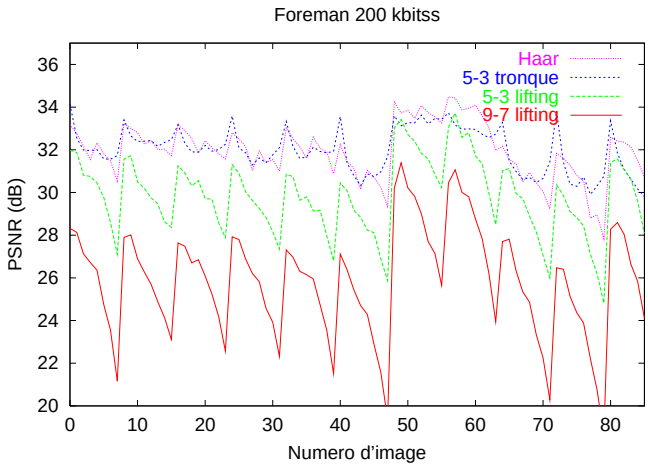

FIG. 5.14 – *Séquence* Foreman *à 200 kbits/s avec les 4 approches : Haar itéré, 5-3 tronqué, 5-3 lifting et 9-7 lifting.*

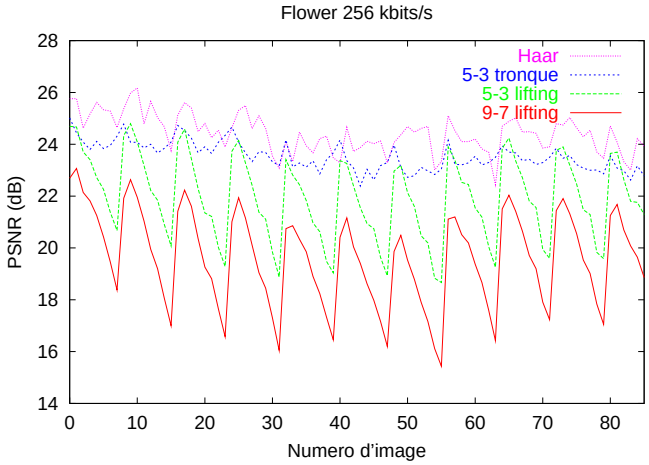

FIG. 5.15 – *Séquence* Flower garden *à 256 kbits/s avec les 4 approches : Haar itéré, 5-3 tronqué, 5-3 lifting et 9-7 lifting.*

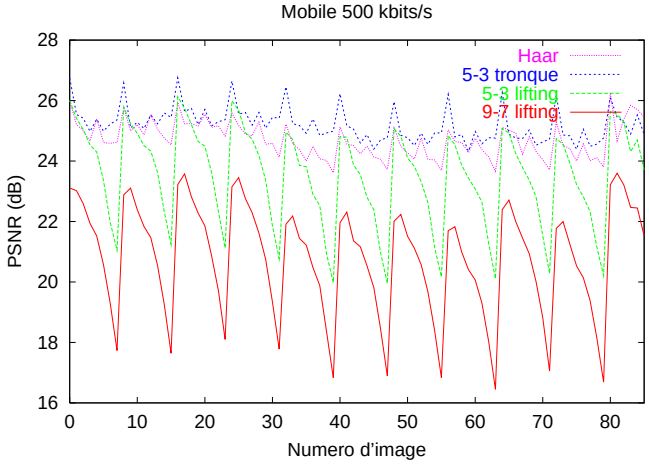

FIG. 5.16 – *Séquence* Mobile and calendar *à 500 kbits/s avec les 4 approches : Haar itéré, 5-3 tronqué, 5-3 lifting et 9-7 lifting.*

différents types de schéma.

5.4.4.1 Analyse temporelle et taille de filtres

La longueur des filtres peut contribuer à améliorer l'exploitation de la redondance temporelle, induisant ainsi une diminution de l'énergie résiduelle présente dans les hautes fréquences. Toutefois, la mise en oeuvre de ce type de filtre dans la dimension temporelle pose différents problèmes.

Le premier problème est lié à la différence entre la taille du support et celle du filtre utilisé. Dans le cas monodimensionnel, lorsque le filtre dépasse le support, on opère généralement une extension de signal de manière symétrique ou anti-symétrique sur les bords du support. Toutefois, ces extensions supposent, en règle générale, un support assez conséquent et n'influent que sur une faible quantité de coefficients. Dans la dimension temporelle, le cas de figure est tout autre. En effet, les GOF sont de taille assez réduite, 8 ou 16, et les phénomènes de repliement de spectre engendrés par les mécanismes d'extension de signal influent sur un nombre important de coefficients. De plus, le problème s'amplifie dès que plusieurs niveaux d'ondelettes sont appliqués, ce qui est le cas général. On se retrouve alors à appliquer un filtre long sur un support de plus en plus court. Filtrer 2 ou 3 coefficients (au niveau le plus haut) avec un banc de filtres 9-7 n'a plus aucun sens et amplifie les problèmes de non orthogonalité de la transformée.

Une solution à ce problème est de ne pas utiliser d'extension sur les bords et d'utiliser les images des GOF suivants et précédents comme dans [XLXZ00]. Cependant, cette solution nécessite une taille de buffer assez importante et entraine par conséquent une latence plus élevé. Enfin, lorsque la scène est faiblement mobile ce type de schéma est intéressant. Lorsqu'elle est plus mouvementée l'estimation de mouvement entre image porte sur des images de plus en plus éloignées temporellement et qui sont potentiellement peu corrélées.

Dans les expérimentations décrites ci-dessus, une extension du signal a été réalisée sur les bords pour les approches 5-3 et 9-7. On peut clairement voir, dans les tableaux et sur les courbes précédentes que ces approches sont clairement pénalisées par les repliements de spectres. Naturellement l'approche 9-7 souffre encore plus de ce phénomène.

Les chutes de PSNR et les moins bonnes performances des approches 9-7 et 5-3 sont liées également à la (non) continuité du mouvement. Les longueurs de filtres, en effet, sont en étroite corrélation avec la continuité du mouvement dans la dimension temporelle. Afin de pleinement bénéficier de la longueur des filtres, un pixel doit se trouver sur une unique trajectoire de mouvement. En pratique, et particulièrement lorsque l'estimation de mouvement est basée bloc, c'est très rarement le cas. Aucune continuité de mouvement entre les images successives ne peut, en effet, être garantie. On obtient alors des lignes de mouvement se recoupant. Dans le cas présent, les champs de mouvement avant et arrière entre images successives n'étant pas bijectifs, la phase de filtrage n'opère plus le long des lignes de mouvement. La figure 5.17 illustre ce problème. Ce

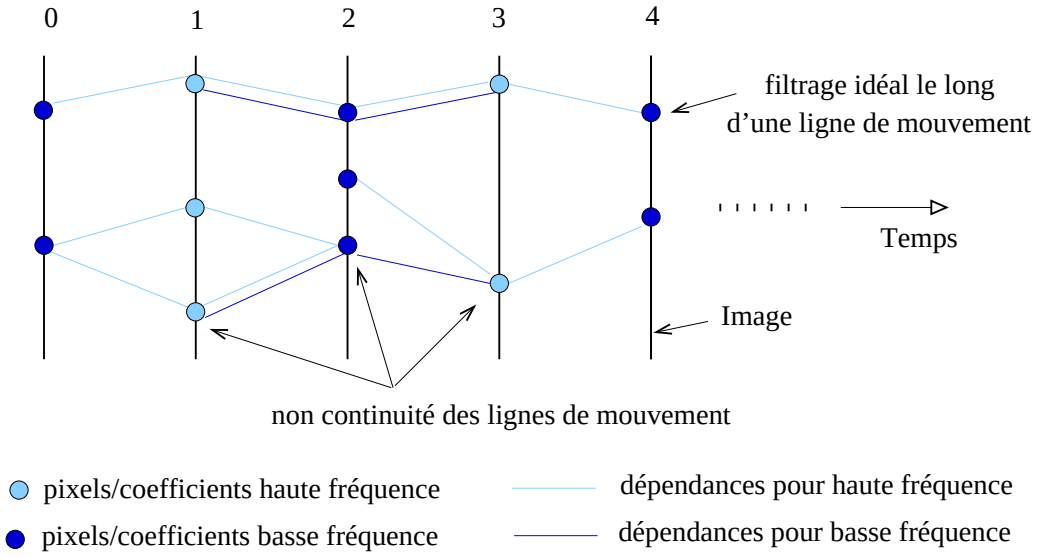

FIG. 5.17 – *Problème de filtrage lifting le long des lignes de mouvement.*

phénomène est amplifié par l'utilisation de schémas de lifting qui permettent de ramener un filtrage long à une suite de filtrages courts portant sur les images adjacentes. Dans un tel cas, une haute fréquence H_t peut être produite à partir de pixels correspondants dans les images précédente I_{t-1} et suivante I_{t+1}, mais au moment du filtrage basse fréquence dans l'image I_{t+1}, la haute fréquence correspondante (i.e. H_t) peut ne pas être utilisée.Il faut noter ici que celle-ci est très rarement utilisée dans le cas d'un double champ de mouvement. Dans un tel schéma, on peut s'interroger sur la réelle signification de la transformation opérée puisque le schéma de lifting n'est pas respecté. Ce phénomène entraîne donc une dérive des trajectoires et ni la décorrélation ni la concentration d'énergie permise par la transformée en ondelettes classique ne sont pleinement atteintes. Donc, deux problèmes se rajoutent aux effets de bords : la continuité du mouvement et le filtrage le long des lignes de mouvement. Il faut remarquer, également, que ces problèmes de filtrage le long des lignes de mouvement sont également accentués dans les zones connectées et non connectées.

5.4.4.2 De la gestion des zones d'occlusion

Nous avons souligné dans la section précédente les problèmes liés à l'utilisation de filtres longs ainsi qu'à une dérive permise par les implémentations lifting conduisant à une non orthogonalité de la transformation. Le filtrage temporel doit, comme on a pu le voir dans le chapitre précédent, faire face à une gestion particulières des zones d'occlusions. Dans les approches 5-3 et 9-7 décrites ici ce problème est contourné par l'utilisation de champs de mouvement avant et arrière entre deux images. Cependant, on a pu voir dans la section précédente que d'autres problèmes induits par ces champs

de mouvement apparaissaient. Dans l'approche 5-3 tronqué, les zones multiplement mises en correspondance engendrées par la non bijectivité des champs de mouvement influent également sur la non orthogonalité de la transformation.

Dans les deux types d'approches précédentes, aucun traitement particulier n'est réservé aux pixels non-connectés par rapport aux pixels connectés. Dans l'approche basée sur le filtrage de Haar et sur l'utilisation de champs de mouvement arrière uniquement, le même traitement que celui proposé dans [CW99] est appliqué à ces pixels. Plusieurs problèmes sont liés à la méthode de gestion proposée. Tout d'abord, avec cette technique un pixel multiplement connecté de la première image est mis en corrélation avec le premier pixel candidat de la seconde image dans l'ordre lexicographique. Il pourrait être plus judicieux de déterminer le meilleur des candidats plutôt que le premier [PPB01]. Après détermination du meilleur candidat, il peut être intéressant de s'assurer de la pertinence de filtrer le pixel courant avec ce candidat. En effet, il se peut que ce soit le meilleur mais que ce ne soit pas un bon candidat. Dans un tel cas, la basse fréquence produite peut ne pas avoir de sens et conduire à des performances de compression et de reconstruction très réduites. Enfin, on peut remarquer que lorsque l'on applique la technique de filtrage de Haar sur un GOF d'une taille donnée, la qualité de reconstruction est décroissante au sein du GOF. Ce phénomène est illustré sur la figure 5.18 pour les 5 premiers GOF de la séquence *Foreman* codée à 200 kbits/s.

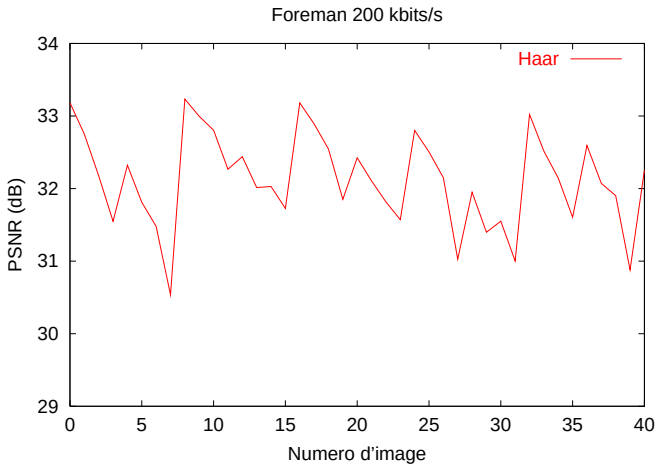

FIG. 5.18 – *Illustration du problème de variation de qualité de reconstruction au sein d'un GOF avec la méthode de Haar, avec les 5 premiers GOF de la séquence* Foreman *codée à 200 kbits/s.*

Nous avons particulièrement étudié ce phénomène et déterminé une raison à cette chute de qualité au sein du GOF. Nous avons tout d'abord remarqué que l'image ayant la meilleure qualité est l'image située temporellement à l'endroit où la basse fréquence temporelle de plus haut niveau est située. En fait, il s'avère que les images les mieux reconstruites sont les images contenant les basses fréquences à un niveau donné. Plus

le niveau auquel se situe la basse fréquence est élevé, meilleure sera la reconstruction de l'image correspondante (cf figure 5.18). Cette hiérarchie est d'autant plus respectée que le mouvement est important. Outre la non orthogonalité de la transfornation, l'explication de ce phénomène est notamment liée à la méthode de gestion des pixels isolés (non connectés) qui sont utilisés directement comme basse fréquence (après mise à l'échelle). Ces pixels détectés comme non connectés à un niveau donné, le restent, en règle générale, au niveau suivant et ceci jusqu'en haut de la pyramide. De plus, comme nous le verrons dans la section suivante le nombre de pixels non-connectés augmente avec le niveau de décomposition temporelle. Ainsi, au plus haut niveau le nombre de pixels (originaux) appartenant à l'image originale est très important et ceux-ci seront choisis prioritairement lors de l'optimisation débit-distorsion du processus de codage des sous-bandes temporelles. Dans ce schéma, à chaque étape il n'y a pas vraiment de bonne concentration de l'énergie dans les basses fréquences et chacune des images basses-fréquences sera mieux reconstruite que l'image contenant les hautes fréquences associées. Par conséquent, sur un GOF de taille 8 il y a un déséquilibre de qualité entre la première et la seconde partie du GOF. L'image 0 est bien reconstruite, puis une chute de qualité opère jusqu'à l'image 4 qui révèle un pic de qualité (cf figure 5.18). La figure 5.19 illustre la formation des différentes sous-bandes avec le filtrage de Haar ainsi que le phénomène de cheminement des pixels non connectés, de l'image 0, entre les niveaux de décomposition.

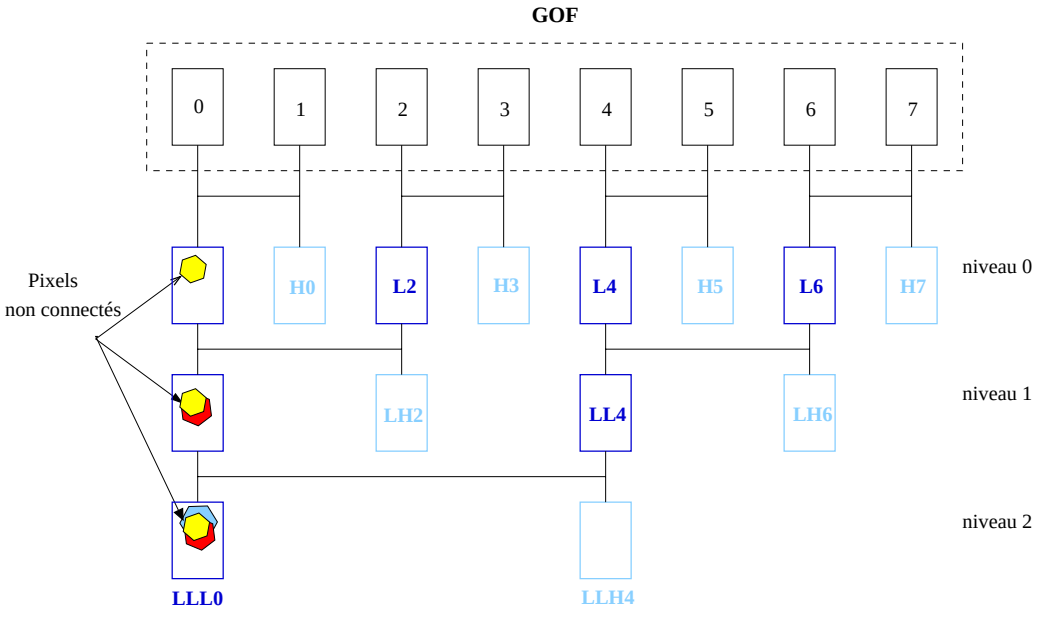

FIG. 5.19 – *Illustration du filtrage de Haar et de la gestion des pixels non-connectés dans les basses fréquences.*

5.4.4.3 Estimation de mouvement dans le domaine transformé

Un autre point faible des méthodes de filtrage temporel compensé en mouvement est justement la compensation de mouvement. Le problème intervient lorsque plusieurs niveaux d'ondelettes sont appliqués. Dans ce cas, une estimation de mouvement est opérée dans le domaine transformé entre les images basses fréquences du niveau précédent. A ce stade intervient deux obstacles influant sur la qualité d'estimation de mouvement entre ces différentes sous-bandes. Le premier est relatif à la distance temporelle croissante entre images à compenser. Ce problème n'en serait pas véritablement un si le second problème, les pixels non-connectés, n'intervenait pas. En effet, à un niveau donné l'estimation de mouvement est d'autant plus difficile que les images basses fréquences entre lesquelles est estimé le mouvement sont de médiocre qualité. On a vu dans la section suivante que chacune de celles-ci contient les pixels non-connectés du niveau précédent. Le filtrage intervenant dans le sens du mouvement et, du fait des recouvrements de blocs, ces mêmes pixels ont alors très peu de chance de se connecter au niveau suivant. Les expérimentations montrent que le nombre de pixels déconnectés augmentent d'au moins 20% entre deux niveaux de décomposition. Ce phénomène couplé avec les distances temporelles croissantes entre les images mènent à une faible qualité de la compensation de mouvement, notamment au niveau le plus haut. Enfin, dans un tel cadre, l'estimation de mouvement basée-blocs ne contribue pas à l'obtention d'un champ de mouvement lissé souhaitable pour la phase de filtrage.

Remarque :

Il faut noter que dans l'approche utilisant le filtrage 5-3 tronqué, l'estimation de mouvement est toujours réalisée dans le domaine original et non dans le domaine transformé.

5.4.4.4 Effet de dérive : *drift*

Le codage basé ondelettes 2D+t est motivé par la scalabilité naturelle permise par ce type de schéma. Toutefois, être capable de supporter des flux scalables dépend à la fois de l'organisation du train binaire et de la capacité d' évitement de l'*effet de drift* intervenant lorsque l'information de reférence, utilisée dans la prédiction temporelle ou ici dans le filtrage temporel compensé en mouvement, diffère entre le codeur et le décodeur. Les schémas proposés dans ce domaine n'évitent pas le phénomène de *drift* intra GOF. Lors de la phase de décomposition temporelle, les signaux originaux sont utilisés pour estimer les champs de mouvement et à chaque étape les sous-bandes passe-bas pleine qualité qui sont utilisées. Si au moment de la transmission, le débit des sous-bandes supérieures est réduit suite aux contraintes du réseau, alors un effet de *drift* intervient sur les étapes suivantes d'analyse/synthèse. Ces effets peuvent contribuer à une forte diminution de la qualité et, sont des limites à considérer dans la conception d'un schéma pleinement scalable. Dans la section 5.6, des résultats portant sur la réduction de l'effet de *drift* sont donnés.

5.4.4.5 Niveaux de décomposition spatiale adaptatifs

Dans la littérature, la majorité des approches considère une décomposition en ondelettes spatiale sur 3 niveaux des sous-bandes temporelles obtenues. Aucune distinction n'est faite entre les sous-bandes passe bas et les sous-bandes passe haut. L'idée de la phase de décorrélation spatiale est de compacter au maximum l'énergie dans la basse fréquence spatiale afin de permettre une meilleure compression. Il faut remarquer cependant que la quantité d'information à décorréler dans les hautes fréquences temporelles est moins importante que celle contenue dans la sous-bande de basse fréquence. C'est pourquoi, afin d'améliorer les performances, nous pensons que des niveaux de décomposition spatiale différents selon les sous-bandes temporelles doivent être utilisés. Les résultats confirment l'intérêt de ces niveaux de décomposition spatiale adaptés.

5.5 Schémas proposés

Dans la section précédente, nous avons étudié trois types d'approches de décomposition spatio-temporelle et avons mené une étude portant sur différentes propriétés de ces schémas. Nous avons également souligné certains points intéressants à prendre en compte dans la conception d'un codeur vidéo basé ondelettes 2D+t. Dans cette section, nous donnons deux nouveaux schémas de codage inspirés des approches 5-3 tronqué et Haar étudiées plus haut. Il faut noter toutefois, que tous les points évoqués dans l'étude n'ont pas été élucidés mais servent de base de travail pour des développements futurs.

Nous avons décidé de nous baser sur ces deux approches au vu des résultats présentés dans la section précédente, mais également du fait des propriétés de ces méthodes. Tout d'abord, ces deux méthodes sont faiblement complexes par rapport aux approches nécessitant des filtres plus longs. De plus, par l'utilisation de ces méthodes nous évitons les problèmes liés aux bords de GOF et n'avons ni besoin d'extension de signal, ni de buffer de taille trop importante. Enfin, l'approche 5-3 tronqué est également intéressante par le fait qu'elle permet d'éviter les problèmes d'estimation de mouvement dans le domaine transformé. Dans la suite, nous décrivons en détails les deux approches proposées.

5.5.1 Une nouvelle approche basée Haar : NHC

La figure 5.20 illustre l'architecture globale du codeur basé sur le filtre de Haar que nous proposons ici. Dans la suite, ce codeur sera noté **NHC** (pour *New Haar Codec*).

5.5.1.1 Vue d'ensemble

Nous avons tout d'abord amélioré les performances du schéma original en appliquant différents niveaux de décomposition spatiale entre les différentes sous-bandes temporelles. Ainsi, 3 niveaux sont appliqués sur la sous-bande passe bas temporelle alors que 2 niveaux seulement sont utilisés pour les hautes fréquences. Nous avons également décidé de modifier le mécanisme de filtrage temporel afin de limiter les problèmes de

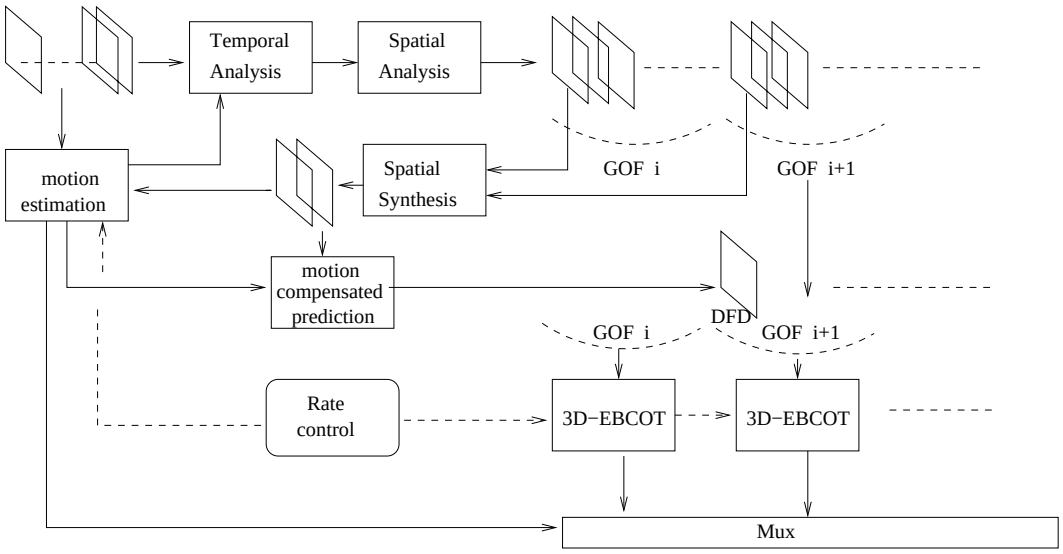

FIG. 5.20 – *Schéma de principe du codeur NHC.*

qualité liés à l'estimation de mouvement décrit dans la section précédente. Enfin, nous avons intégré la possibilité de faire de la prédiction Inter-Gof.

5.5.1.2 Une nouvelle architecture de filtrage

L'idée première de cette nouvelle architecture était d'obtenir une distance temporelle plus faible entre les sous-bandes à compenser à chaque niveau. On peut ainsi obtenir une meilleure estimation/compensation de mouvement tirant parti de la gestion des pixels non connectés dans les images basses fréquences. En effet, du fait que les pixels originaux passent au travers des niveaux de décomposition, les images, entre lesquelles est appliquée l'estimation de mouvement, deviennent plus proches. Pour ce faire, nous avons décidé de placer la basse fréquence temporelle finale au centre du GOF. La figure 5.21 illustre ce nouveau mécanisme. Ici les champs de mouvement sont des champs avant sur la première moitié du GOF et des champs arrière sur la seconde. Ce processus permet de limiter le nombre de pixels non-connectés à chaque niveau et fournit donc de meilleures basses fréquences. Enfin, la distance sur laquelle porte le filtrage est d'au plus quatre images dans une direction ou dans l'autre. Dans la section 5.6, nous montrons l'amélioration de qualité permise par cette nouvelle architecture vis-à-vis de l'architecture de filtrage classique.

Remarque :
Le placement de la basse fréquence au centre du GOF entraîne une augmentation de la qualité au centre du GOF avec des bords de moins bonne qualité. Afin de limiter cet effet nous avons pondéré les sous-bandes hautes fréquences situées aux extrémités pour favoriser leur contribution dans l'optimisation débit-distorsion d'EBCOT-3D.

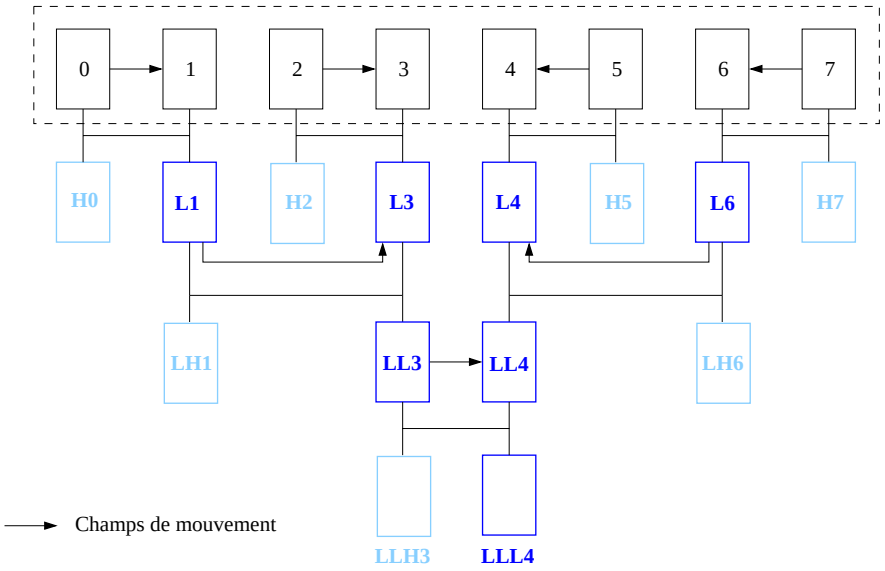

FIG. 5.21 – *Nouvelle structure de filtrage temporelle compensée en mouvement basée Haar.*

5.5.1.3 Prédiction Inter-Gof

Nous avons également rajouté la possibilité d'une prédiction Inter-GOF au système de codage. On distingue alors deux types de GOF : Intra et Inter. Le mécanisme de prédiction temporelle nécessite un champ de mouvement supplémentaire et est réalisé en boucle fermée afin de prévenir certains effets de dérive Inter-GOF. La prédiction en boucle fermée peut-être réalisée en prenant comme information de référence une image (sous-bande) décodée à un débit plus faible, comme dans les couches basses d'une représentation scalable classique.

Après le codage d'un GOF Intra, la basse fréquence temporelle est reconstruite à l'encodeur par une phase de décodage et de synthèse temporelle. Cette sous-bande reconstruite est ensuite utilisée comme information de référence dans la compensation de mouvement de la sous-bande basse fréquence temporelle du GOF suivant (Inter). La DFD (*Displaced Frame Difference*) issue de cette prédiction est ensuite substituée à la sous-bande basse fréquence du GOF Inter. Comme pour les hautes fréquences, la DFD subit 2 niveaux de décomposition spatiale. Au décodeur, le processus inverse est réalisé afin de reconstruire la sous-bande basse fréquence du GOF puis la transformation inverse est exécutée.

Remarque :
Afin de réduire les hautes fréquences dûs aux effets blocs, nous appliquons un filtre passe-bas sur l'image compensée avant de procéder à la différence avec la sous-bande basse fréquence du GOF précédent.

5.5.1.4 Structure du train binaire

Le train binaire relatif à un GOF est ensuite agencé de manière simple (cf figure 5.22). Les informations de mouvement précèdent le train binaire scalable fournit par EBCOT-3D. Lors de la formation des couches de qualité, chacune des sous-bandes temporelles apparaissent par ordre d'importance. Chacun des trains binaires de chaque GOF est ensuite finement scalable jusqu'à la zone contenant les champs de mouvement mis à la suite l'un de l'autre. L'idéal serait de pouvoir avoir un champ de mouvement scalable permettant de répartir de manière optimale les informations de mouvement et de texture dans le train binaire final. Le train binaire obtenu est scalable en débit, temporellement et spatialement.

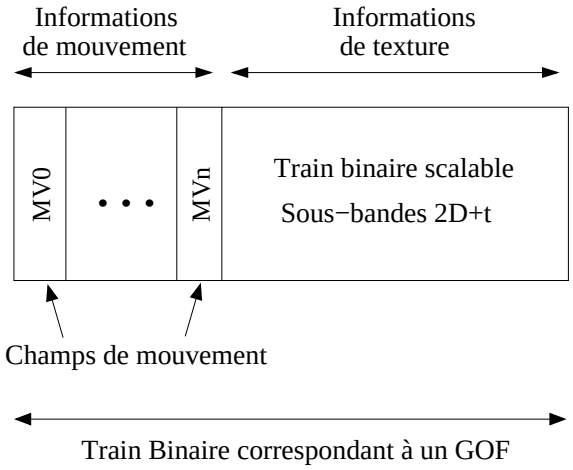

FIG. 5.22 – *Train binaire relatif à un GOF dans le codeur NHC.*

5.5.2 L'approche basée 5-3 tronqué : TLC

La figure 5.20 illustre l'architecture globale du codeur basé sur le filtre 5-3 tronqué que nous proposons ici. Dans la suite, ce codeur sera noté **TLC** (pour *Truncated Lifting Codec*).

5.5.2.1 Vue d'ensemble

De la même façon que pour le codeur NHC nous appliquons des niveaux de décomposition spatiale différents entre les sous-bandes hautes et basses fréquence temporelles : 3 pour la sous-bande passe-bas et 2 pour les sous-bandes passe-haut. Deux options additionnelles ont également été rajoutées. La première concerne le rajout d'une possibilité de prédiction temporelle entre la première et la dernière image du GOF. Afin d'améliorer la qualité, l'utilisation d'une boucle fermée et le codage d'un résidu est également intégré dans le système de codage.

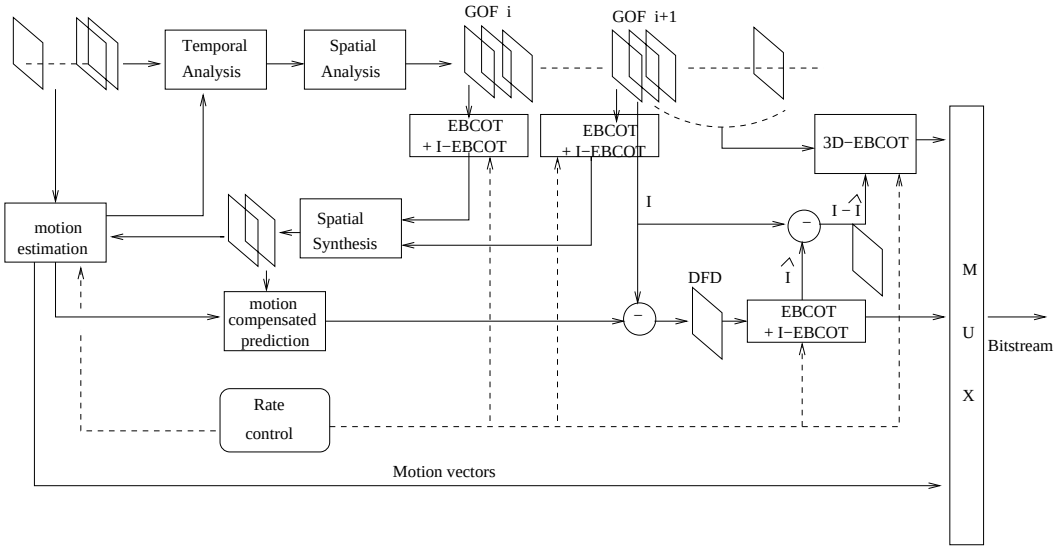

FIG. 5.23 – *Schéma de principe du codeur TLC.*

5.5.2.2 Prédiction temporelle en boucle fermée

Dans le schéma original, les images se trouvant aux extrémités du GOF sont codées séparément en mode Intra avec le codeur EBCOT, puis les sous-bandes hautes fréquences sont, quant à elles, codées simultanément avec EBCOT-3D. Remarquant l'impact de la qualité des images extrêmes sur la qualité de reconstruction du GOF entier, nous avons décidé d'intégrer la possibilité de prédire la dernière image du GOF à partir de la première image.

L'algorithme exact est le suivant. La première image de la séquence est codée et décodée séparément (en mode Intra). Au moment du codage du premier GOF, cette image codée/décodée sert de référence pour la prédiction de la dernière image du GOF. La DFD est ensuite codée/décodée et transmise. Cette image reconstruite ainsi que la première image du GOF servent ensuite pour la phase de filtrage temporel. Afin d'améliorer la qualité de la dernière image qui servira pour le filtrage du suivant, nous codons un *résidu* supplémentaire représentant la différence entre la dernière image du GOF originale et sa version reconstruite. Ce résidu ainsi que les sous-bandes hautes fréquences sont ensuite données à EBCOT-3D. Le processus est ensuite répété pour le prochain GOF en prenant la dernière image reconstruite du GOF précédent comme première image du nouveau GOF.

Avec cette architecture nous considérons un débit minimal pour la couche de base qui sera toujours compatible avec les contraintes de bande passante variables du réseau. Cette pseudo couche de base est codée/décodée du côté source et le signal reconstruit sert de reférence à l'estimation de mouvement et au filtrage temporel. Cette décompostion temporelle compensée en mouvement en boucle fermée contribue à réduire les effets de *drift*.

5.5.2.3 Structure du train binaire

La structure du train binaire correspondant à un GOF est différente ici (cf figure 5.24). Les informations relatives à la couche de base sont tout d'abord placées dans le train binaire. On y retrouve le champ de mouvement entre la première image et la dernière image du GOF, suivi du train binaire scalable de la DFD. Ensuite, viennent les informations de la couche d'amélioration qui elles se composent des champs de mouvement nécessaires au filtrage temporel, suivies du train binaire scalable des sous-bandes hautes fréquences et du résidu issu de EBCOT-3D. La structure finale choisie fournit naturellement une scalabilité temporelle. La scalabilité en résolution est obtenue en tronquant de manière appropriée dans les deux sous-trains binaires correspondant respectivement à la dernière image du GOF et aux sous-bandes hautes fréquences.

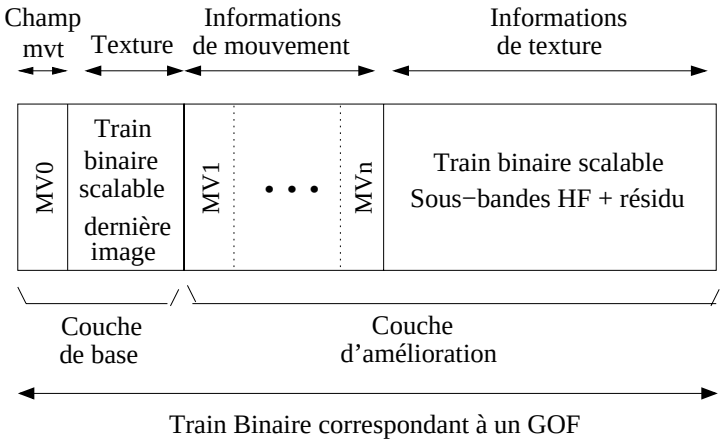

FIG. 5.24 – *Train binaire relatif à un GOF dans le codeur TLC.*

5.6 Résultats

5.6.1 Configurations

Nous donnons ici les résultats des expérimentations menées avec les deux codeurs NHC et TLC. Une version du codeur NHC utilisant la technique de filtrage originale est également proposée. Ces résultats servent à montrer l'intérêt des choix réalisés. Nous rappelons que l'estimation de mouvement est effectuée avec une précision pixelique et que la taille de la fenêtre de recherche, pour les résultats donnés ici, est fixée à 16.

Les résultats obtenus sont ensuite comparés aux codeurs MPEG-4 part 2 ainsi qu'au codeur H.264 (MPEG-4 part 10). Le codeur MPEG-4 part 2 utilisé est le codeur Momusys utilisant une estimation de mouvement au 1/2 pixel ainsi qu'une compensation avec recouvrement de blocs (OBMC: *Overlapped Block Motion Compensation*), la taille de la fenêtre de recherche est fixée à 32. La régulation de débit opère sur le long terme

et conduit, potentiellement, à de fortes variations de débit (i.e. des rafales). Le codeur H.264 intègre quant à lui une estimation de mouvement au 1/4 de pixels, une fenêtre de taille 32 et le mécanisme d'optimisation débit-distorsion a posteriori est activé. Une méthode de régulation a été activée pour les simulations.

Nous avons testé les mêmes séquences que dans la section 5.4.3, à savoir : *Hall*, *Foreman*, *Mother and daughter*, *Coastguard*, *Mobile and calendar*, *Flower garden* et *Tempete*. Ces séquences sont au format CIF PAL (352×288) avec une fréquence de 15 Hz. Toutes les séquences décodées possèdent 85 images. Les expérimentations ont été réalisées pour 5 débits cibles : 100, 140, 200, 256 et 500 kbits/s .

5.6.2 Performances

Les tableaux 5.5, 5.6, 5.7, 5.8, 5.9 and 5.10 illustrent respectivement les PNSR moyen obtenus avec:

- l'approche NHC basée sur la structure de filtrage de Haar originale,
- l'approche NHC sans mode Inter,
- l'approche NHC incluant le mode Inter avec une fréquence de 2 (i.e. un GOF Inter tous les deux GOFs),
- l'approche TLC,
- le codeur MPEG-4 part 2 Momusys,
- le codeur H.264 JM 2.1.

Il faut noter que certains débits n'ont pu être atteints pour certaines séquences avec le codeur MPEG-4. En effet, selon l'activité de la séquence la mise au maximum des paramètres de quantification ne suffisent pas à l'obtention de certaines gammes de débit.

	NHC (filtrage classique)				
Débit (kbits/s)	100	140	200	256	500
Hall	32.41	34.20	35.90	37.05	38.77
Foreman	30.02	31.28	32.52	33.45	35.67
Mother	36.19	37.48	38.83	39.84	42.05
Coastguard	26.20	27.03	27.97	28.71	30.51
Mobile	20.49	21.35	22.31	22.96	24.81
Flower	21.74	22.72	23.67	24.52	26.48
Tempete	24.77	25.71	26.69	27.40	29.29

TAB. 5.5 – *PSNR moyen obtenu pour les 7 séquences aux débits de 100, 140, 200, 256 et 500 kbits/s avec le codeur NHC avec la structure de filtrage classique (mode Intra).*

	NHC (Intra)				
Débit (kbits/s)	100	140	200	256	500
Hall	32.45	34.24	35.97	37.04	38.73
Foreman	30.41	31.63	32.83	33.72	35.84
Mother	36.34	37.64	38.99	39.96	42.02
Coastguard	26.48	27.28	28.11	28.84	30.58
Mobile	20.87	21.72	22.72	23.34	25.08
Flower	21.83	22.89	23.81	24.63	26.57
Tempete	25.04	25.96	26.91	27.61	29.37

TAB. 5.6 – *PSNR moyen obtenu pour les 7 séquences aux débits de 100, 140, 200, 256 et 500 kbits/s avec le codeur NHC incluant la nouvelle structure de filtrage (mode Intra).*

	NHC (Inter)				
Débit (kbits/s)	100	140	200	256	500
Hall	33.48	35.08	36.61	37.50	39.03
Foreman	30.54	31.70	32.93	33.80	35.90
Mother	36.89	38.14	39.43	40.34	42.32
Coastguard	26.34	27.16	28.03	28.72	30.52
Mobile	21.17	22.01	22.97	23.57	25.29
Flower	21.89	22.90	23.87	24.60	26.58
Tempete	25.33	26.25	27.16	27.84	29.56

TAB. 5.7 – *PSNR moyen obtenu pour les 7 séquences aux débits de 100, 140, 200, 256 et 500 kbits/s avec le codeur NHC incluant la nouvelle structure de filtrage (mode Inter).*

	TLC				
Débit (kbits/s)	100	140	200	256	500
Hall	33.10	34.61	36.15	37.12	39.34
Foreman	30.17	31.38	32.60	33.53	36.30
Mother	36.61	37.79	39.18	40.09	42.95
Coastguard	25.49	26.30	27.35	28.10	30.49
Mobile	20.96	21.87	22.78	23.48	26.02
Flower	21.20	22.05	23.23	24.01	26.85
Tempete	25.25	26.06	27.07	27.88	30.40

TAB. 5.8 – *PSNR moyen obtenu pour les 7 séquences aux débits de 100, 140, 200, 256 et 500 kbits/s avec le codeur TLC.*

	MPEG-4 part 2				
Débit (kbits/s)	100	140	200	256	500
Hall	31.80 [a]	33.11	34.45	35.26	37.45
Foreman	X	30.08	31.95	32.94	35.97
Mother	35.06	36.75	38.30	39.29	41.30
Coastguard	X	X	27.11	28.46	30.99
Mobile	X	X	X	23.55	26.90
Flower	X	X	23.57 [b]	24.42	27.08
Tempete	X	25.76 [c]	26.77	27.84	30.53

TAB. 5.9 – *PSNR moyen obtenu pour les 7 séquences aux débits de 100, 140, 200, 256 et 500 kbits/s avec le codeur vidéo MPEG-4.*

[a] débit réel: 110 kbits/s.
[b] débit réel: 210 kbits/s.
[c] débit réel: 165 kbits/s.

	H.264 JM 2.1				
Débit (kbits/s)	100	140	200	256	500
Hall	36.44	37.62	38.62	39.21	40.86
Foreman	32.78	34.30	35.83	36.86	39.76
Mother	39.36	40.72	42.08	42.93	45.08
Coastguard	27.39	28.37	29.45	30.25	32.69
Mobile	24.25	26.01	27.63	28.73	31.32
Flower	23.58	24.96	26.44	27.49	30.26
Tempete	28.08	29.41	30.81	31.70	34.22

TAB. 5.10 – *PSNR moyen obtenu pour les 7 séquences aux débits de 100, 140, 200, 256 et 500 kbits/s avec le codeur vidéo H264.*

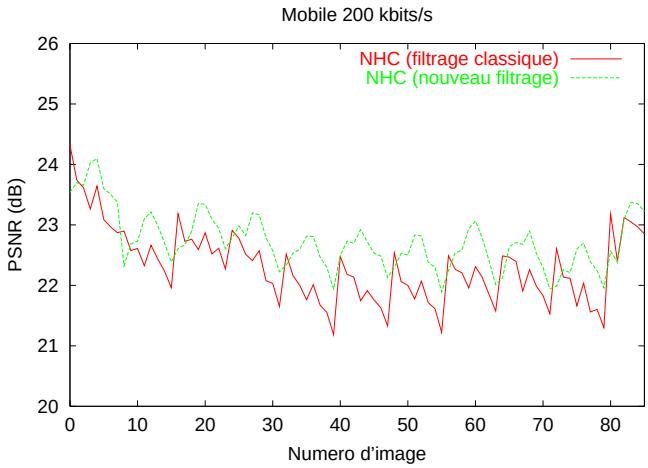

FIG. 5.25 – *Illustration de l'intérêt de la nouvelle architecture de filtrage dans le codeur NHC.*

5.6.3 Analyse

5.6.3.1 NHC : intérêt de la nouvelle structure de filtrage

Les résultats donnés dans les tables 5.5 et 5.7 montrent que l'utilisation de la nouvelle structure de filtrage temporel permet d'améliorer les résultats. Le gain varie d'une séquence à l'autre en fonction de la quantité de mouvement. Plus le mouvement est important, plus le gain est significatif. La figure 5.25 illustre les performances respectives des deux méthodes sur la séquence *Mobile and calendar* à 200 kbits/s.

5.6.3.2 NHC : intérêt de de la prédiction Inter-GOF

Les tables 5.6 et 5.7 représentant respectivement le codeur NHC sans et avec mode Inter, montrent le gain substantiel permis par l'utilisation de la prédiction Inter-GOF. La figure 5.26, donnant les performances du codeur NHC dans les deux modes de codage sur la séquence *Tempete* à 140 kbits/s, illustre l'amélioration obtenue avec le mode Inter.

5.6.3.3 TLC versus 5-3 tronqué

Les tables 5.2 et 5.8 donnent respectivement les PSNR moyens obtenus avec les approches 5-3 tronqué originale et le codeur TLC. On peut voir ici que l'approche TLC s'avère également significativement supérieure à l'approche 5-3 tronquée n'incluant pas la boucle fermée, ni le codage de résidu, ni la différenciation de niveaux de décompositions spatiales entre les sous-bandes hautes et basses fréquences. La figure 5.27 illustre les performances des deux codeurs sur la séquence *Foreman* à 140 kbits/s.

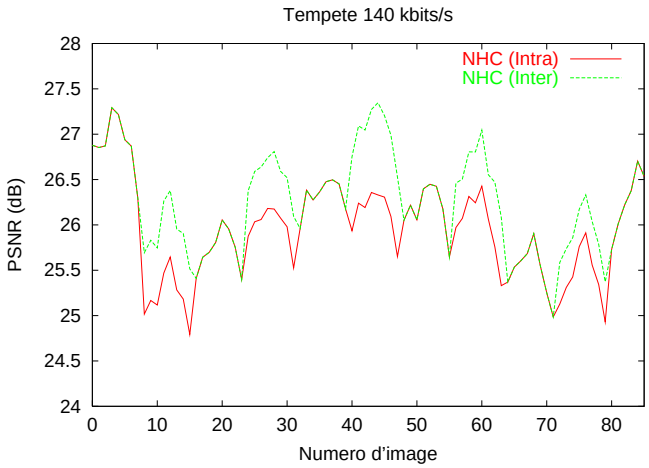

FIG. 5.26 – *Illustration de l'intérêt d'utilisation de GOF Inter dans le codeur NHC.*

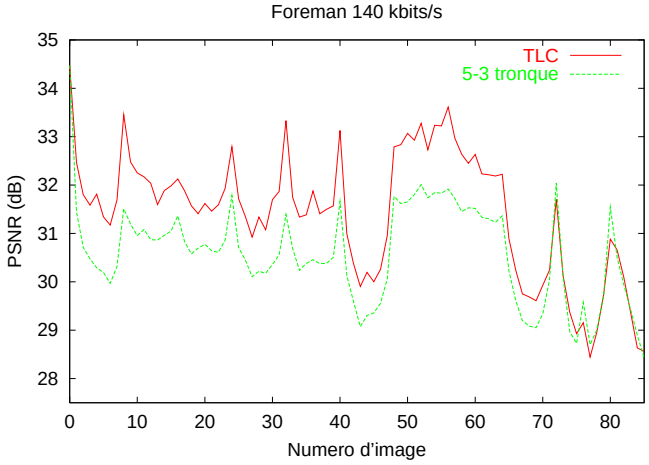

FIG. 5.27 – *Illustration du gain apporté par l'approche TLC par rapport à l'approche 5-3 tronqué originale.*

5.6.3.4 Comparaisons

Au vu des résultats donnés dans les tables 5.8, 5.7, 5.10 et 5.9, on peut observer que les codeurs NHC et TLC proposés offrent des performances comparables. Celles-ci surpassent très nettement le codeur MPEG-4 à bas débit. Les résultats obtenus restent cependant inférieurs à ceux de H.264. Il faut toutefois noter que la complexité des codeurs vidéos MPEG-4 et H.264 est très nettement supérieure à celle des deux codeurs proposés. Il faut remarquer également que H.264 code la première image en Intra avec un débit très élevé (cf figures 5.31et 5.32). Dans les séquences avec peu de mouvement utilisées ici, cette image Intra fait rapidement gagner plusieurs dB sur toute la séquence. Nous n'avons pas intégré ce type de mécanisme dans nos codeurs vidéo. Enfin, les solutions que nous proposons sont finement scalables contrairement aux approches MPEG-4 et H.264.

Nous donnons ici quelques courbes afin d'illustrer les PSNR obtenus au cours du temps pour certaines séquences avec les approches : NHC, TLC, MPEG-4 part2, H.264 JM2.1. Nous illustrons ici les séquences *Mother and daughter* à 100 kbits/s (figure 5.28), *Foreman* à 140 kbits/s (figure 5.29), *Coastguard* à 200 kbits/s (figure 5.30), *Flower garden* à 256 kbits/s 5.31 et *Hall* à 500 kbits/s (figure 5.32).

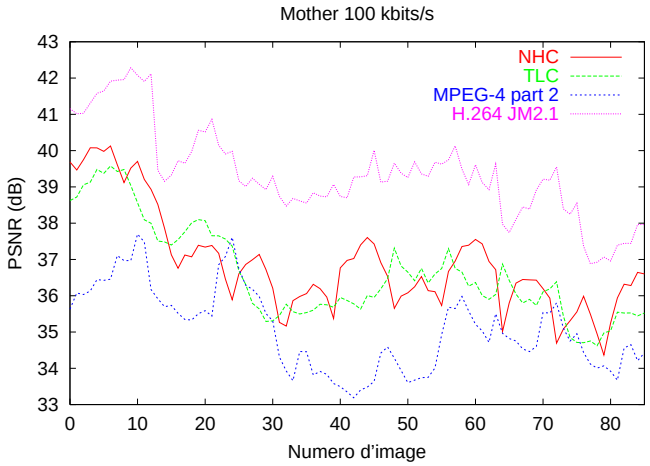

FIG. 5.28 – *Séquence* Mother and daughter *à 100 kbits/s avec les 4 approches : NHC, TLC ,MPEG-4 part 2 et H.264 JM 2.1.*

5.6.4 Scalabilité

Les deux codeurs proposés ici sont scalables en résolution temporelle, en résolution spatiale et en qualité. Les scalabilités spatiales et temporelles sont difficilement illustrables et les bases de comparaison pertinentes sont difficiles à déterminer. Concernant la scalabilité SNR, il faut noter que pour le codeur NHC les résultats fournis dans la table 5.6 ont été obtenus à partir d'un seul flux décodé aux 5 débits étudiés (i.e. 100,

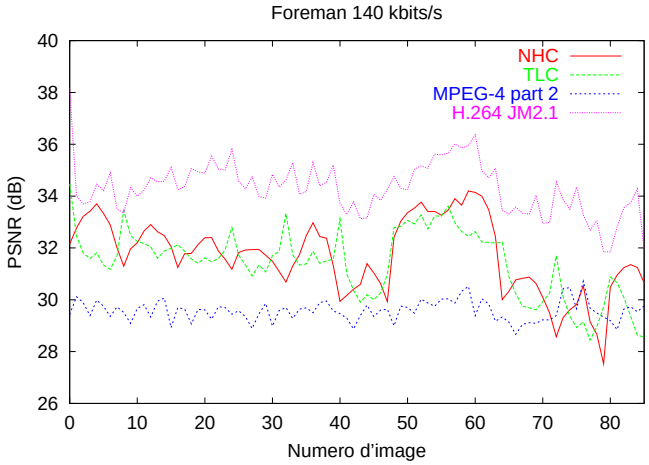

FIG. 5.29 – *Séquence* Foreman *à 140 kbits/s avec les 4 approches : NHC, TLC ,MPEG-4 part 2 et H.264 JM 2.1.*

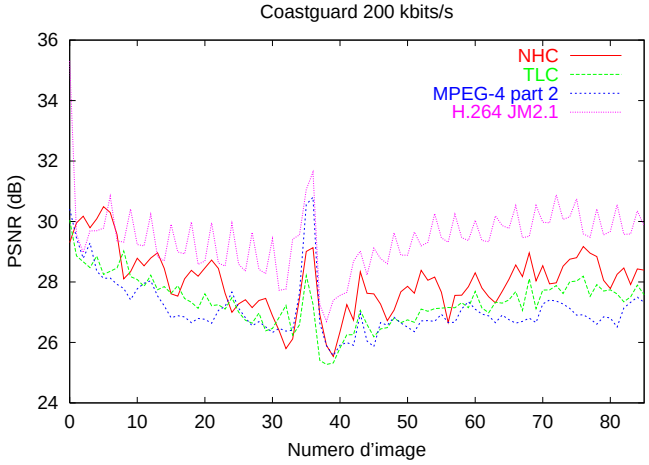

FIG. 5.30 – *Séquence* Coastguard *à 200 kbits/s avec les 4 approches : NHC, TLC ,MPEG-4 part 2 et H.264 JM 2.1.*

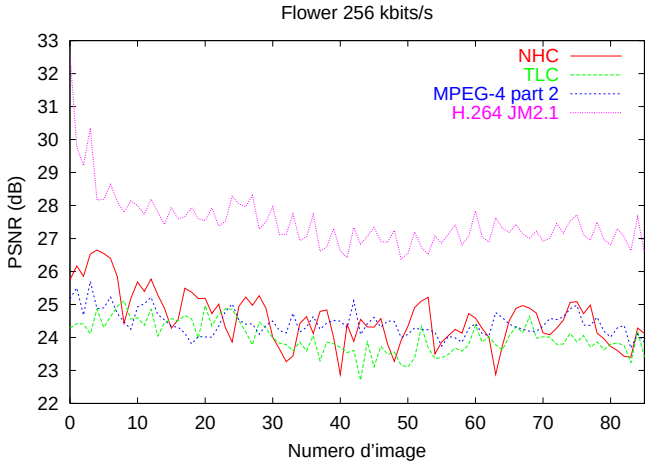

FIG. 5.31 – *Séquence* Flower garden *à 256 kbits/s avec les 4 approches : NHC, TLC ,MPEG-4 part 2 et H.264 JM 2.1.*

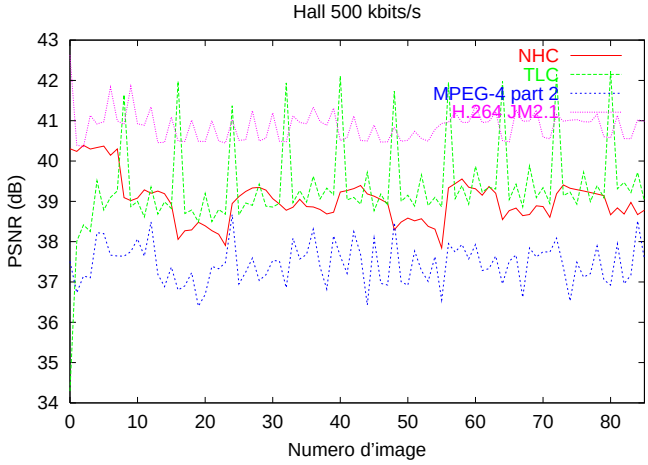

FIG. 5.32 – *Séquence* Hall *à 500 kbits/s avec les 4 approches : NHC, TLC ,MPEG-4 part 2 et H.264 JM 2.1.*

140, 200, 256 et 500). Les résultats obtenus pour la scalabilité SNR avec le codeur TLC sont sous-optimaux puisqu'aucune optimisation débit-distorsion globale n'est effectuée entre le débit alloué à la couche de base et celui alloué à la couche d'amélioration.

5.7 Perspectives

Les résultats précédents montrent que les systèmes de codage basés sur des sous-bandes 2D+t sont devenus des alternatives viables aux systèmes traditionnels basés sur des schémas de prédiction temporelle hybride. De nombreuses voies sont encore totalement ouvertes dans la définition de ces mécanismes de codage nouvelle génération. Nous présentons ici diverses voies intéressantes et envisagées à plus ou moins long terme concernant les différentes briques des algorithmes de compression présentés dans les sections précédentes.

5.7.1 Mouvement

Concernant la phase d'estimation/codage de mouvement, plusieurs points sont à l'étude. Une estimation de **mouvement sous-pixelique** permettant, de plus, l'obtention de champs de mouvement plus lissés semble intéressante. Le but ici est de réduire le nombre de pixels non connectés qui influent sur les performances de compression tant au niveau de l'orthogonalité mais également dans le processus même de codage basé EBCOT. En effet, on peut observer que ce type de pixels entraine un comportement non souhaitable de la part d'EBCOT qui tente de coder prioritairement ces pixels.

Une première étape de **compensation de mouvement globale** du GOF entier suivie d'une estimation de mouvement locale (basée-blocs) peut également contribuer à réduire l'énergie résiduelle dans les hautes fréquences temporelles et le coût de codage de l'information de mouvement.

Des gains significatifs peuvent également être espérés du côté de la **compression des champs de mouvement**. L'utilisation d'un algorithme de codage arithmétique adaptatif basé-contextes dans l'esprit de l'algorithme CABAC (*Context-based Adaptive Binary Arithmetic Coding*) [MBHW01] devrait permettre une diminution significative du coût du mouvement et donc une amélioration potentielle de la qualité de l'information de mouvement. Dans ce cadre, l'utilisation de contextes dans la dimension temporelle permettant d'exploiter les corrélations entre champs de mouvement peut également être envisagée.

Enfin, la définition de **champs de mouvement scalables** est une voie attrayante dans l'objectif d'un schéma de codage pleinement scalable. Une exploitation intelligente de cette information dans la phase de transformation spatio-temporelle devrait permettre de réduire substantiellement les problèmes de dérive intra-GOF.

5.7.1.1 Transformation spatio-temporelle

La phase de transformation spatio-temporelle compensée en mouvement est la phase qui semble la plus ouverte en terme de recherche. Différents points sont évoqués ici.

Cette phase est directement corrélée avec la phase d'estimation/codage du mouvement. Ainsi, en suite directe des perpectives liées au mouvement l'idée est d'étendre la transformation dans la dimension temporelle à la prise en compte de mouvement sous-pixelique. La **réversibilité de la transformation** est évidemment l'objectif dans cette extension.

Nous considérons qu'une des principales perspectives concernant la phase de transformation spatio-temporelle réside dans la mise en oeuvre de schéma de filtrage dynamique permettant d'adapter la taille des filtres utilisés à la celle du support. Cette notion de **taille de filtre adaptative** peut intervenir à différents niveaux.

Nous avons fait le choix ici d'utiliser des GOF de taille fixe et limitée. Nous envisageons l'utilisation de GOF de taille variable déterminée, par exemple, par le nombre de pixels non connectés. L'utilisation de filtres de taille adaptative facilite la mise en oeuvre de ce type mécanisme.

Il est également possible d'utiliser au sein d'un GOF des filtres de taille différente selon l'activité d'une zone spatiale donnée. Dans le cas de zones très mobiles, des filtres courts peuvent être préférables à des filtres plus longs convenant mieux aux zones immobiles.

A l'instar de ce qui est fait dans la dimension spatiale, la multiplication du nombre de niveaux dans la transformation temporelle doit tenir compte de la taille du support. En effet, l'utilisation d'un banc de filtre 9-7 sur 3 niveaux pour une petite image de taille 16x16 n'aurait aucun sens. L'idée ici est de réduire la taille des filtres en fonction du niveau et donc de la taille du support. Ces solutions peuvent permettre une réduction des effets de bords dûs au repliement de spectre sur les bords.

Lorsque plusieurs niveaux d'ondelettes temporelles sont appliqués, une phase d'estimation de mouvement dans le domaine transformé est réalisée. Toutefois, du fait des traitements associés aux pixels non connectés dans les niveaux précédents et de l'allongement des distances (temporelles)inter images, les champs de mouvement deviennent de plus en plus hétérogènes. L'hétérogénéité ainsi que leur discutable pertinence sont des limites aux performances de compression. L'approche simple retenue ici consistait à minimiser la distance entre les sous-bandes de basses fréquences à filtrer au plus haut niveau. Nous pensons que la définition d'un critère permettant de prendre la décision quant à l'**utilisation pertinente ou non d'une compensation en mouvement** entre deux images, pourrait conduire à une amélioration des performances. Le critère pourrait être un simple seuil portant sur le nombre de pixels non connectés.

Une approche différente, mais proche, pourrait ne prendre en compte le mouvement que dans les zones intéressantes. La décision pourrait être prise suivant un critère d'EQM ou de DFD. Par le biais de cette technique des zones assimilables à des “ma-

croblocs" Intra peuvent être intégrées.

Enfin, l'utilisation d'un schéma de filtrage temporel basé sur la notion de **lignes de mouvement** proposée dans [XXLZ01] nous parait attrayante. L'idée est toutefois différente puisqu'il n'est pas envisagé d'appliquer des filtres de manière glissante. Il n'est donc pas nécessaire d'avoir un buffer de taille importante et par conséquent aucune latence supplémentaire n'est ajoutée. La solution envisagée consiste, après formation des lignes de mouvement d'utiliser des **filtres de taille adaptative** selon la taille de ces lignes de mouvement. Ce schéma nous permet directement de filtrer différemment les zones en fonction de leur mobilité. De plus, le filtrage retrouve tout son sens puisque ce sont bien les éléments d'un même support qui serviront à la production des coefficients de hautes et basses fréquences. Enfin, une détermination des lignes de mouvement pertinentes doit être mis en oeuvre. L'intégration de zones Intra est parfaitement compatible avec cette approche.

5.7.1.2 Codage des sous-bandes spatio-temporelles

Dans la version actuelle du codeur TLC, le débit alloué à la couche de base est fixé a priori en proportion du débit total. Une amélioration envisagée est la définition d'un **mécanisme d'optimisation débit-distorsion globale** permettant de répartir de manière optimale le débit entre la pseudo-couche de base et la couche d'amélioration dans le codeur TLC. Ce mécanisme rendra plus aisée l'obtention d'une pleine scalabilité en débit pour ce codeur. Il pourrait être intégré dans le codeur EBCOT-3D.

Dans le schéma utilisé ici, chacun des coefficients des sous-bandes spatio-temporelles est quantifié à l'aide d'un quantificateur scalaire uniforme intégrant une zone morte. Il est envisagé d'étudier d'autres types de quantification comme par exemple une quantification basée treillis (TCQ) prenant en compte les modèles statistiques des sous-bandes sous-jacentes. L'étude d'une quantification adaptée aux différents types de coefficients, en fonction de leur appartenance ou non aux zones d'occlusion, est également envisagé. La prise en compte de cette information de quantification dans la définition des contextes définis dans EBCOT-3D devrait permettre une amélioration des performances. Enfin, l'ajout de **contextes dans la dimension temporelle** est également une piste envisagée.

Remarque:
Nous pensons que dans la conception de ce type de schéma de codage, il pourrait être intéressant de modéliser les biais introduits par la non orthogonalité de la transformation afin de permettre leur prise en compte dans la phase de codage entropique ou de quantification menant ainsi à une sorte d'*orthogonalisation a posteriori*.

5.8 Conclusion

Plusieurs schémas de filtrage compensé en mouvement ont été proposés dans la littérature à des fins de codage vidéo basé sur des ondelettes 2D+t. Dans ce chapitre, nous proposons la conception d'un nouveau schéma adapté à la compression bas débit. Dans ce schéma, le mouvement est estimé par un algorithme hiérarchique basé sur des blocs de taille variable. Le coût du mouvement est régulé par l'utilisation d'un quadtree contraint en débit. Nous avons proposé ensuite une étude portant sur différents schémas d'analyse temporelle. Dans cette étude, l'accent est porté sur les modèles de mouvement ainsi que sur les tailles de filtres temporels. L'étude menée a aussi pour but de souligner les limites des approches classiques et d'ouvrir d'éventuelles voies de recherches. Suite aux résultats de cette étude, nous avons proposé ici deux nouveaux schémas de codage scalable à grain fin. La scalabilité est obtenue par l'extension du codeur EBCOT à la dimension temporelle, pour le codage des sous-bandes spatio-temporelles. Les performances des codeurs proposés ont été comparées à celles obtenues avec les codeurs MPEG-4 et H.264. Les nouveaux codec proposés ont des performances significativement supérieures à celles de MPEG-4 mais restent inférieures à celles obtenues avec H.264. Il faut toutefois noter que la complexité des codeurs MPEG-4 et H.264 est très nettement supérieure à celle de nos codeurs. Enfin, les algorithmes proposés sont pleinement scalables ce qui n'est pas le cas des codeurs standards.

Conclusion

L'étude menée durant ces travaux de thèse s'incrit dans la problématique de la transmission de données vidéo temps-réels sur réseau de paquets de type Internet. Dans ce cadre, nous nous sommes, tout d'abord, attachés à la partie réseau en proposant des méthodes de contrôle de congestion et de régulation de débit dédiées aux transmissions multimédia en mode point-à-point et multipoint. Dans un second temps, afin d'approfondir le couplage entre la source et le réseau nous avons proposé la conception d'un algorithme de codage scalable à granularité fine dans le but de pouvoir adapter de manière plus précise le débit de la source vidéo. Nous faisons ici la synthèse des contributions apportées dans cette thèse, puis donnons quelques perspectives d'évolution de ces travaux.

5.9 Synthèse

Plusieurs points complémentaires ont été abordés dans cette thèse. Nous les partitionnons en trois classes : contrôle de congestion pour la transmission vidéo en mode point-à-point, contrôle de congestion pour la transmission vidéo multipoint en couches et codage vidéo scalable nouvelle génération.

Contrôle de congestion pour transmission vidéo point-à-point

- Nous avons proposé dans la première partie de la thèse un nouveau protocole de contrôle de congestion TCP-compatible basé sur RTP prenant en compte, à la différence des protocoles classiques, les caractéristiques des flux multimédia (paquets de tailles variables, délais,...) transportés.
- Un modèle global de régulation de débit considérant les modèles de délais de la source ainsi que les contraintes de délais de bout-en-bout de flux temps-réels, pour la transmission de vidéo sur l'Internet, a été couplé avec le nouveau protocole TCP-compatible proposé.
- Le modèle a été validé par un nombre élevé d'expérimentations réalisées sur l'Internet entre divers sites. On observe que ce modèle permet de réduire de manière significative le nombre de retards de paquets. Il permet ainsi d'améliorer la qualité du signal décodé, tout en maximisant l'utilisation de la bande passante disponible.
- Les performances ont encore pu être améliorées en introduisant une nouvelle stratégie d'adaptation de la fréquence temporelle de la source (réduction du

nombre d'images codées par seconde) permettant d'obtenir un meilleur compromis entre fréquence temporelle et PSNR, et ceci même en présence de contraintes de débits très variables. Les performances obtenues, en considérant de faibles délais de mise en *buffer*, sont comparables aux quelques approches proposées dans la littérature pour des délais plus de dix fois supérieurs.

- Malgré le fait que cet algorithme ait été validé avec un codec vidéo H.263+, l'approche a été conçue avec pour objectif l'utilisation de codecs vidéo scalables à grain fin (FGS). Les mécanismes peuvent donc s'étendre aisément à ce type de codeur vidéo.

Contrôle de congestion pour transmission vidéo multipoint en couches

- Un nouvel algorithme de contrôle de débit multicast hybride orienté émetteur-récepteur prenant en compte les caractéristiques débit-distorsion de la source, a été proposé dans un second temps.
- L'algorithme s'appuie sur un mécanisme d'aggrégation distribué des rapports des récepteurs dans les noeuds du réseau, permettant une classification des récepteurs en fonction du taux de pertes et de la bande passante estimés sur leur lien. Cet algorithme est, de plus, couplé à un système de transmission vidéo FGS multicouche permettant une adaptation plus aisée du nombre de couches et de leurs débits respectifs.
- Les informations collectées par la source sont injectées dans un mécanisme de sélection du nombre de couches à émettre, de leurs débits et de leurs éventuels niveaux de protection. La décision se caractérise par la prise en compte explicite de la qualité perçue par chaque récepteur. Elle cherche alors à maximiser la qualité globale perçue par l'ensemble des récepteurs.
- Plusieurs critères d'estimation de bande passante, par les récepteurs, ont été étudiés. Les résultats, obtenus avec le simulateur de réseau NS2, montrent que le couple (taux de pertes, estimation de bande passante TCP-compatible), utilisé comme variable discriminante dans le mécanisme de classification, conduit à la meilleure qualité globale.
- Le protocole ainsi obtenu se comporte de manière équitable avec TCP en terme de partage de bande passante.

Codage vidéo finement scalable nouvelle génération

- Dans cette partie de la thèse, nous avons proposé la conception d'un nouveau schéma de codage vidéo finement scalable adapté à la compression bas débit. Ce schéma de codage se base sur l'utilisation d'une décomposition en ondelettes dans les dimensions spatiale et temporelle (2D+t). Nous avons proposé une architecure de codage vidéo complète.

- Afin de mieux exploiter les corrélations temporelles l'information de mouvement est utilisée dans la phase de filtrage. Le mouvement est estimé par un algorithme hiérarchique basée sur des blocs de taille variable et stocké dans *quadtree* de vecteurs mouvement. Ces techniques basées sur l'utilisation de *quadtree* permettent notamment d'affiner l'estimation dans les zones de l'image où le mouvement nécessite une précision spatiale accrue. Elles permettent, de plus, de réguler le coût du mouvement de manière optimale au sens débit-distorsion.
- Une étude a ensuite été menée sur la problématique de l'analyse spatio-temporelle compensée en mouvement. Dans cette étude, l'accent est porté sur les modèles de mouvement ainsi que sur les tailles de filtres temporels. L'étude menée a aussi pour but de souligner les limites des approches classiques et d'ouvrir d'éventuelles voies de recherches. Les problèmes induits, par la gestion des zones d'occlusion, sur les performances de la phase de décorrélation temporelle ont également été abordés. Nous avons évoqué, ensuite, les problèmes de *drift* inhérent aux mécanismes de codage basés sur des décompositions en ondelettes 2D+t. L'intérêt d'un nombre de niveaux de décomposition spatiale adapté aux types de sous-bandes temporelles est également abordé.
- Cette étude nous a mené à proposer deux nouveaux schémas de codage finement scalables bas débit utilisant des schémas de filtrage temporel différents. Nous avons proposé une nouvelle architecture de filtrage basée sur le filtre de Haar permettant de limiter certains problèmes déterminés dans l'étude. L'intégration d'un mécanisme de prédiction Inter GOF a été proposé. Nous avons proposé également l'utilisation d'un boucle fermée permettant de limiter les effets de *drift*.
- Les sous-bandes spatio-temporelles sont alors codées avec le codeur EBCOT-3D prenant en compte la dimension temporelle dans sa phase d'optimisation débit-distorsion. Ce codeur entropique nous fournit alors un train binaire finement scalable temporellement, SNR et en résolution.
- Nos nouveaux codecs exhibent des performances significativement supérieures à celles de MPEG-4 mais restent inférieures à celles obtenues avec H.264. Il faut toutefois noter, toutefois, que la complexité des codeurs MPEG-4 et H.264 est très nettement supérieure à celle de notre schéma et que, de plus, les algorithmes proposés ici sont, quant à eux,finement scalables.

5.10 Perspectives

La première perspective de ces travaux est naturellement l'intégration du schéma de codage finement scalable nouvelle génération développé dans les chaînes de transmission unicast ou multicast proposées. De cette manière, la chaîne de communication vidéo complète sera développée.

Le mécanisme de contrôle de flux développé en mode point-à-point considère des applications fortement interactives. Un perspective pourrait être l'extension de ces travaux aux applications de diffusion vidéo de type *streaming*. Dans ce cadre, le couplage des algorithmes proposés avec le concept de "pré-chargement" (ou *prefetching*) semble

intéressant en vue de fournir une meilleure qualité de service. Le concept de "pré-chargement" s'inspire du fait qu'au cours d'une session, il existe des périodes brèves pendant lesquelles il peut y avoir des excédents de bande passante. Ainsi, en tirant partie des larges capacités de stockage des PCs ou stations de travail, le client peut alors "pré-charger" des portions de flux vidéo durant ces périodes. Il est également possible ici d'utiliser des mécanismes de demande de retransmission des paquets perdus. Enfin, le couplage de tous les mécanismes précédents avec l'idée de ralentissement et d'accélération de la diffusion aux clients, évoquée dans [SFG01], devrait permettre d'obtenir un système de diffusion vidéo sur l'Internet TCP-compatible et offrant une très bonne qualité de service.

Le protocole de contrôle de flux et de congestion pour la transmission multipoint de flux vidéo en couches que nous avons proposé, fait l'hypothèse d'un codage tandem. C'est à dire que l'on considère qu'il y a séparation entre le codage de source et le codage de canal. Par conséquent, la métrique utilisée, dans la détermination de la répartition de débit entre couches, afin de maximiser la qualité globale perçue par les clients n'est plus adaptée à l'utilisation de codage conjoint source-canal. Une approche envisagée dans ce domaine est l'utilisation de transformations sur des bases de fonctions redondantes (par exemple: bancs de filtres sur-échantilonnés), qui permettent un couplage direct de la redondance avec le signal. L'extension de notre algorithme d'optimisation globale à ce type de codage n'est pas directe et nécessite une reformulation du problème.

Les approches proposées ici font l'hypothèse d'un réseau Internet "*best-effort*". Si cet objectif est incontournable, il est néanmoins intéressant de considérer comme objectif secondaire la conception d'un algorithme qui s'accomode, voire exploite, à son avantage les avancées dans l'offre de qualité de service du réseau Internet, vers une différenciation de services.

Comme a pu le voir dans la section 5.7, les perpectives concernant l'évolution des schémas de codage vidéo finement scalable dits de nouvelle génération, sont très nombreuses. Les principaux points concernent l'utilisation de filtres de taille adaptative dans la dimension temporelle, et l'utilisation d'une précision de mouvement sous-pixelique. L'hypothèse d'une quantification adaptée des pixels appartenant à des zones d'occlusion est également intéressante. Enfin, les travaux réalisés ici se sont placés dans une optique de compression scalable bas débit. L'extension vers des gammes de débit nettement supérieure, de l'ordre de plusieurs Méga-bits, pour des applications de diffusion Haute Définition est une piste qui devrait également être explorée.

Annexe A

Le protocole TCP

Sur l'Internet, deux protocoles de transport dominent TCP (*Transmission Control Protocol*) [Pos81] pour la transmission de données fiable et UDP (*User Datagram Protocol*) [Pos80] pour le multiplexage démultiplexage des paquets sans aucune garantie sur la transmission et démuni contrôle de congestion. Nous décrivons ici brièvement le protocole TCP, principal protocole de transport utilisé sur l'Internet (i.e. pour les communications traditionnelles). Nous donnons ici les principaux points nécessaires à la compréhension de son fonctionnement général. Plus de détails peuvent être trouvés dans la littérature [Pos81, Jac88, Ste94, WS95, Ste97].

A.1 Présentation Générale

Le protocole TCP est orienté connexion. Ainsi, avant toute transmission de données entre deux entités, une connexion doit être établie. La connexion est dite *full-duplex*, c'est-à-dire que les données peuvent être échangées simultanément dans les deux sens entre les deux entités distantes. TCP permet une transmission de données fiables; toute donnée émise arrivera au récepteur (sans perte). Pour permettre cela, le récepteur acquitte tous les paquets qu'il reçoit. Les paquets non acquittés au bout d'un certain temps sont retransmis par l'émetteur. Dans le cas où deux paquets identiques arrivent du côté récepteur (i.e. retransmission inutile), un des deux paquets est jeté. Les paquets arrivant dans le désordre sont réordonnés. Un numéro de séquence, correspondant au décalage en octet depuis le premier octet du flux, est associé à chaque paquet. Les rapports d'acquittement (ACK) contiennent le numéro de paquets qu'il s'attend à recevoir (i.e. le numéro du dernier octet prêt à être consommé incrémenté de un). Lorsque les deux entités ont des données à transmettre, TCP peut faire du *piggybacking* d'acquittements (i.e. les acquittements sont transmis dans les paquets de données).

A.2 Contrôle de flux

Le contrôle de flux a pour but de prévenir le sur-remplissage du buffer de réception d'un récepteur lent par un émetteur rapide. L'émetteur n'est pas autorisé à envoyer plus de données que le récepteur peut en traiter. Pour cela une "fenêtre coulissante" (*sliding window*) à l'émetteur et au récepteur limite le nombre de paquets (non acquittés) circulant dans le réseau. L'envoi de paquets diminue la taille de la fenêtre, ainsi l'émetteur peut envoyer des paquets jusqu'à ce que la taille de la fenêtre soit nulle. Lorsqu'un paquet est acquitté la taille de la fenêtre augmente. On peut donc décider d'émettre à nouveau.

A.3 Slow-start

Les connexions entre machines distantes transitent au travers de différents équipements intermédiares (routeurs) ayant des ressources limitées. Envoyer des paquets en rafales à un débit élevé peut causer de fortes congestions et réduire le *goodput* (i.e. quantité de données reçues sans pertes) de la connexion.

Jacobson [Jac88] présente une solution à ce problème par l'utilisation d'un algorithme dit de *slow-start*. Ce mécanisme requiert l'utilisation d'une seconde fenêtre appelée "fenêtre de congestion" *cwnd*. La taille de cette fenêtre est initialisée à un segment (i.e. un paquet de données). L'émetteur utilisera le minimum entre la fenêtre de congestion et la fenêtre du récepteur comme borne supérieure pour le nombre de paquets pouvant transiter en même temps dans le réseau. Lorsqu'une nouvelle connexion s'établie, l'émetteur peut envoyer un segment et doit attendre ensuite l'acquittement correspondant. Chaque ACK augmente la fenêtre de congestion d'un paquet. L'émetteur peut donc envoyer deux paquets. Lorsque les acquittements correspondants arrivent la fenêtre peut être augmentée de deux paquets, donnant une fenêtre de taille quatre etc. L'augmentation de la fenêtre de congestion est exponentielle. Sa taille augmente à chaque RTT (*Round-Trip Time*: délai d'aller-retour entre l'émetteur et la source) jusqu'à l'observation de la première perte.

A.4 Détection de pertes

Dans un protocole basé sur des acquittements, l'émetteur est responsable de la détection des pertes de paquets. Les paquets perdus sont repérés par les trous subsistants dans les numéros de séquence des paquets acquittés. L'émetteur TCP repère les paquets perdus de deux façons différentes : détection de retards et trois ACK identiques.

A.4.1 Retards

TCP attend un certain intervalle de temps l'acquittement d'un paquet. Si l'ACK n'est pas reçu dans cet intervalle de temps, le paquet correspondant est considéré comme perdu et est retransmis. Ce délai de retransmission sur retard T_o est crucial aux performances du protocole. Si il est trop large, le protocole n'est pas très performant puisqu'il

attend longtemps avant de prendre la décision. Si à l'inverse, le délai est trop petit celui-ci ré-émettra des paquets non perdus gaspillant ainsi inutilement la bande passante. De manière évidente ce délai doit être relatif au RTT. On utilisera, pour estimer le RTT, une valeur lissée par une moyenne exponentielle pondérée des mesures de RTT réalisées :

$$S_{RTT} = \frac{1}{8} m_{RTT} + \frac{7}{8} S_{RTT} \tag{A.1}$$

avec m_{RTT} la dernière valeur mesurée sur le réseau, en faisant la différence entre l'instant de réception d'un ACK et l'instant d'émission du paquet de données correspondant.

A l'origine, le paramètre T_o était un multiple de RTT. Toutefois, les connexions observant une forte variance en terme de RTT étaient trop pénalisées. Les performances ont donc été améliorées en prenant compte cette variance var_{RTT} dans le calcul du paramètre T_o [Jac90].

$$var_{RTT} = \frac{1}{4} |S_{RTT} - m_{RTT}| + \frac{3}{4} var_{RTT} \tag{A.2}$$

et

$$T_o = S_{RTT} + 4 var_{RTT} \tag{A.3}$$

Le calcul est simplifié par l'utilisation de l'écart type des mesures de RTT comme estimation de la variance qui requiert une racine carrée.

Le paramètre T_o a un impact important sur le débit d'émission, notamment lorsque le taux de pertes sur le réseau est élevé. Dans le cas où la congestion causant les retards est persistante et où les paquets retransmis ne sont pas acquitté dans l'intervalle de temps, TCP double le délai T_o à chaque tentative de retransmission d'un paquet.

A.4.2 Trois acquittements identiques

Attendre l'expiration du *timer* de retransmission augmente considérablement le délai. Afin d'accélérer la retransmission des paquets perdus, un mécanisme appelé *fast retransmit/fast recovery* est ajouté à TCP. Lorsqu'un paquet est reçu "dans le désordre", un ACK est transmis avec le prochain numéro de séquence attendu. Cet ACK est alors un duplicata du précédent ACK. L'émetteur ne sait pas alors si ces deux ACK dupliqués sont causés par une perte de paquet ou par l'arrivée d'un paquet à réordonnancer. Cependant, à l'arrivée d'un troisième ACK, l'émetteur considère qu'il s'agit d'une perte et retransmet le paquet manquant sans attendre l'expiration du *timer*. Ce mécanisme est décrit dans [Ste94]

A.5 Contrôle de congestion

Afin de détecter l'éventuelle bande passante supplémentaire, TCP augmente son débit également après la phase de slow-start. Cette augmentation est toutefois très petite par rapport à l'augmentation exponentielle réalisée dans cette phase. A chaque réception d'un acquittement, la fenêtre est augmentée de $1/cwnd$. Puisque $cwnd$ paquets sont acquittés pendant un RTT, l'augmentation totale sera d'un paquet par RTT.

Lorsque TCP détecte une congestion, il décide de diminuer son débit d'émission, permettant ainsi au réseau de sortir de cette phase de congestion. Dans le cas de trois acquittements identiques, TCP réduit sa fenêtre de congestion, donc son débit d'émission, par deux. L'émetteur continue avec un contrôle de congestion normal en augmentant sa fenêtre de $1/cwnd$. Lorsque c'est un retard qui est détecté, cela indique qu'aucun paquet n'a été reçu par le récepteur ou que tous les acquittements ont été perdus. On est alors face à une congestion très importante. Dans ce cas, l'émetteur met sa fenêtre de congestion à un et utilise l'algorithme de slow-start pour déterminer un débit d'émission approprié. La plupart des implémentations TCP réduise leur débit d'émission en réponse à une perte de paquet seulement une seule fois par RTT. Les autres pertes intervenant dans la même fenêtre sont alors "ignorées".

Il a été montré dans [CJ89] que ce mécanisme dit AIMD (*Additive Increase Multiplicative Decrease*) a de bonnes propriétés de stabilité et permet un partage équitable de la bande passante entre les multiples flux.

Remarque:
Il existe plusieurs implémentations différentes de TCP à l'heure actuelle. Les plus importantes versions sont : *Tahoe*, *Reno*, *New Reno* et *Sack*. La plus utilisée sur l'Internet est sans contexte TCP-Reno.

Annexe B

Le protocole RTP/RTCP

B.1 Le protocole RTP (*Real-time Transport Protocol*)

Les protocoles de transport classiques utilisés sur l'Internet, comme TCP (*Transmission Control Protocol*), ne sont pas adaptés aux flux temps-réel. En effet, leur fiabilité se fait au dépend d'importantes variations de délais et de débits qui sont inacceptables en terme de qualité de service pour des applications temps-réel. C'est pourquoi, un grand nombre d'applications se sont mises à utiliser des protocoles, non fiables, comme UDP (*User Datagram Protocol*) plus adaptés à la manipulation de tels flux. Cependant, les services fournis par UDP ne répondent pas vraiment aux besoins des applications temps-réel. Ainsi est né RTP, en 1996 (*Request For Comments* 1889) [SCFJ96], un protocole dédié aux applications temps-réel et particulièrement bien adapté au multimédia. Il a été développé par le groupe de travail AVT (*Audio Video Transport*) de l'IETF (*Internet Engineering Task force*). Nous en sommes actuellement à la version 2 [SCFJ98] du protocole.

RTP est un protocole de niveau applicatif qui est indépendant des couches inférieures. En effet, il peut être utilisé soit directement au dessus de IP (*Internet Protocol*), soit au dessus du protocole UDP (*User Datagram Protocol*) (voir figure B.1). Malgré cela, les applications temps-réel comme la vidéo-conférence utilisent RTP au dessus de UDP afin de bénéficier des services qu'il fournit comme le multiplexage et les calculs de checksum. RTP est un protocole dit "non fiable" dans le sens où il ne garantit aucune qualité de service (réception, pertes, délai etc.). Il tend, toutefois, à satisfaire l'ensemble des services qu'une application multimédia temps-réel se doit d'attendre de sa pile de communication. Pour parvenir à assurer un minimum de contrôle, RTP est couplé avec le protocole de contrôle RTCP (*Real-time Transport Control Protocol*).

Nous décrirons, tout d'abord, les différents services fournis par le couple RTP/RTCP. Puis, il sera question des différents paquets et format de paquets RTP mis en oeuvre dans la réalisation de ces services.

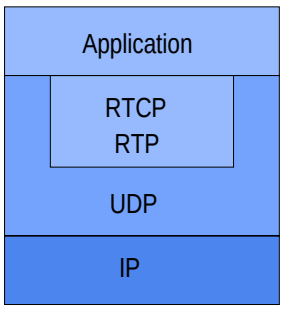

FIG. B.1 – *RTP/RTCP dans la pile de protocoles de l'Internet*

B.1.1 Les différents services offerts

Les services fournis par le couple de protocoles RTP/RTCP à l'application sont résumés ci-dessous:

- un service d'envoi de paquets en direction d'un ou plusieurs récepteurs (unicast ou multicast)
- un service d'estampillage temporel et séquentiel
- un service de réception de paquets d'un utilisateur donné parmi un groupe d'émetteurs
- un service d'information sur l'identité des participants
- un service de filtrage de certaines sources, parmi l'ensemble des utilisateurs
- un service d'information sur les conditions de réception de chacun des participants
- un service de statistiques concernant les caractéristiques des flots reçus et transmis

B.1.2 Fonctionnement et architecture du protocole

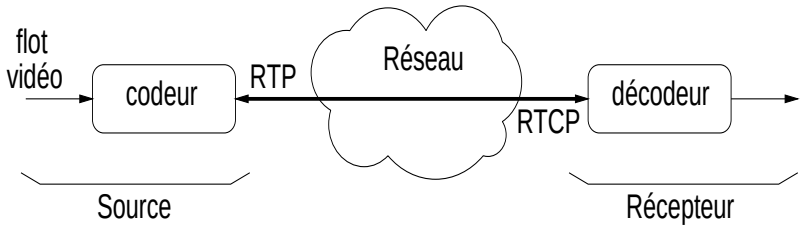

FIG. B.2 – *Transmission vidéo sur Internet*

Une session RTP est composée d'une double connexion. La première connexion supporte les échanges de données proprement dites et la seconde est la connexion supportant le contrôle. Ces connexions sont bidirectionnelles. Ainsi, chaque participant de la session peut être à la fois source et récepteur.

Le fonctionnement du couple RTP/RTCP repose sur les différents types de paquets mis en jeu dans une session RTP. On distingue parmi ces paquets, les paquets de données

RTP et les paquets de contrôle RTCP. Ce dernier est utilisé afin de fournir des rapports concernant les conditions de transport des paquets de la connexion RTP. Il est basé sur un système de transmission périodique de paquets de contrôle à tous les participants de la session. Les principaux paquets de contrôle utilisés sont les *Sender Reports* et les *Receiver Reports* envoyés respectivement par le récepteur (ou les récepteurs) et par la source (ou les sources). C'est à partir de ces échanges d'informations que les différents services, notamment le service de statistiques, peuvent être offerts aux applications.

Nous décrirons brièvement dans les sections suivantes la structure des principaux types de paquets RTP et RTCP. Nous ne détaillerons que les champs jugés les plus intéressants dans le cadre de nos travaux.

B.1.3 Format du paquet de données RTP

Un paquet de données RTP est composé de deux parties: l'entête et la zone de données utiles ou *payload*. Le format de l'entête standardisé est défini dans la norme [SCFJ98]. Par contre, le format des données utiles est spécifique au type de média transporté.

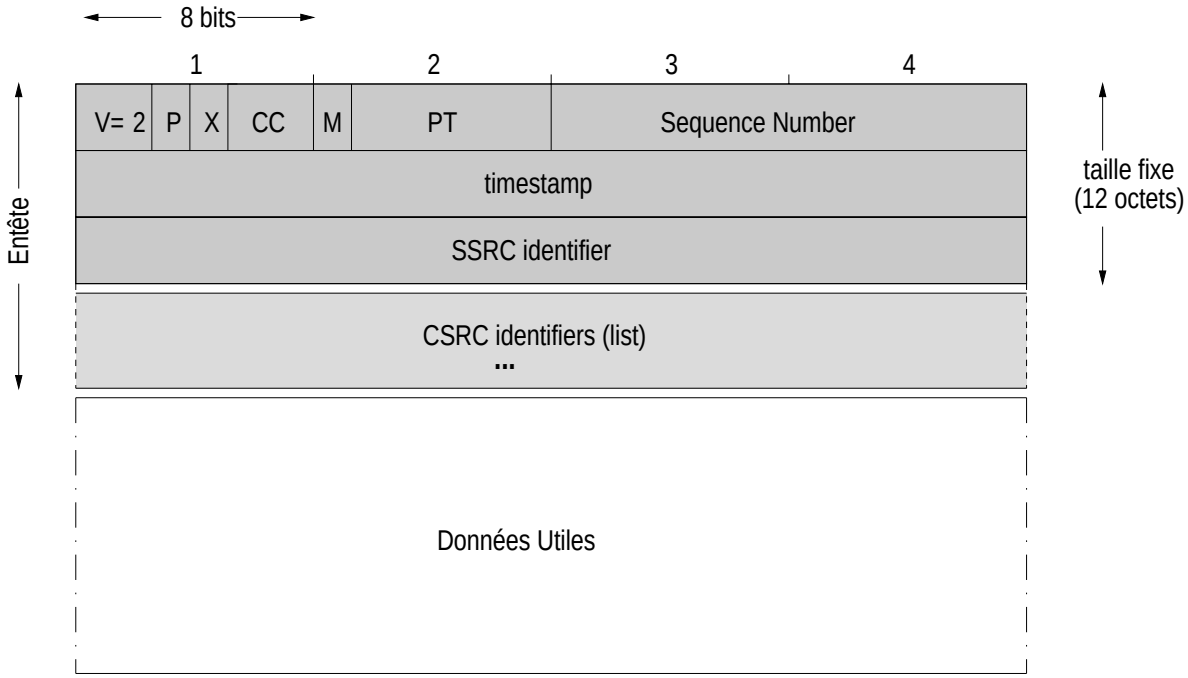

FIG. B.3 – *Format d'un paquet de données RTP*

L'entête d'un paquet RTP est composé d'une partie fixe de douze octets présents dans tous les paquets RTP et d'une partie variable que nous ne décrirons pas ici (voir figure B.3) .

Les champs les plus intéressants et nécessaires aux besoins des applications sont les

suivants:

M (Marker): Ce champ indique la présence de certains "événements" dans le paquet, comme par exemple la présence de la fin d'une image.

PT (Payload Type): Ce champ identifie le format de la zone de données utiles transportées. Ainsi, le récepteur est capable de déterminer, par exemple, s'il s'agit de données audio ou vidéo par exemple et s'il s'agit de flux MPEG-4 ou H.263+. Ceci lui permet de connaître le format d'encapsulation utilisé. Tous les formats existants, ainsi que leur valeur de type correspondant, sont répertoriés dans [Sa99].

Sequence number: Ce champ indique le numéro de séquence attribué au paquet RTP. RTP ne garantit pas la réception des paquets dans l'ordre de leur émission, toutefois ce champ permet au récepteur de reconstruire la séquence envoyée. Ce champ peut être utilisé pour déterminer l'emplacement des données du paquet dans le train binaire initial. Il permet aussi de déterminer les numéros de séquence et donc le nombre de paquets perdus. Sa valeur initiale est choisie aléatoirement à des fins de sécurité.

Timestamp: Ce champ représente une valeur d'estampille temporelle. Il indique l'instant de présentation du premier octet de données contenu dans le paquet. Cette estampille a une valeur relative dépendante du type de données. Son utilisation permet, par exemple, à l'application de synchroniser des flux audio et vidéo. La valeur initiale est aussi choisie aléatoirement.

SSRC (Synchronization SouRCe): C'est une valeur choisie aléatoirement qui identifiera de manière unique la source de ce paquet RTP, pendant toute la session RTP.

B.1.4 Format des paquets RTCP

Les principaux paquets de contrôle sont les *Receiver Reports* (RR) et les *Sender Reports* (SR) envoyés respectivement par le récepteur et par la source. Nous les décrivons brièvement ici.

B.1.4.1 Format d'un Receiver Report

Un Receiver Report est, comme son nom l'indique, un rapport de réception. Ce paquet ne contient pas de données. Son rôle est de fournir à la source ou aux sources, qui lui transmettent des données, des informations relatives aux conditions de réception de ces flux.
Les champs les plus intéressants sont les suivants:

Fraction Lost: Ce champ indique le taux de paquets perdus depuis le dernier rapport de réception. Cette information peut permettre à la source, par exemple, de décider de baisser momentanément son débit en cas de constatation d'un fort taux de pertes, afin de limiter d'éventuelles congestions.

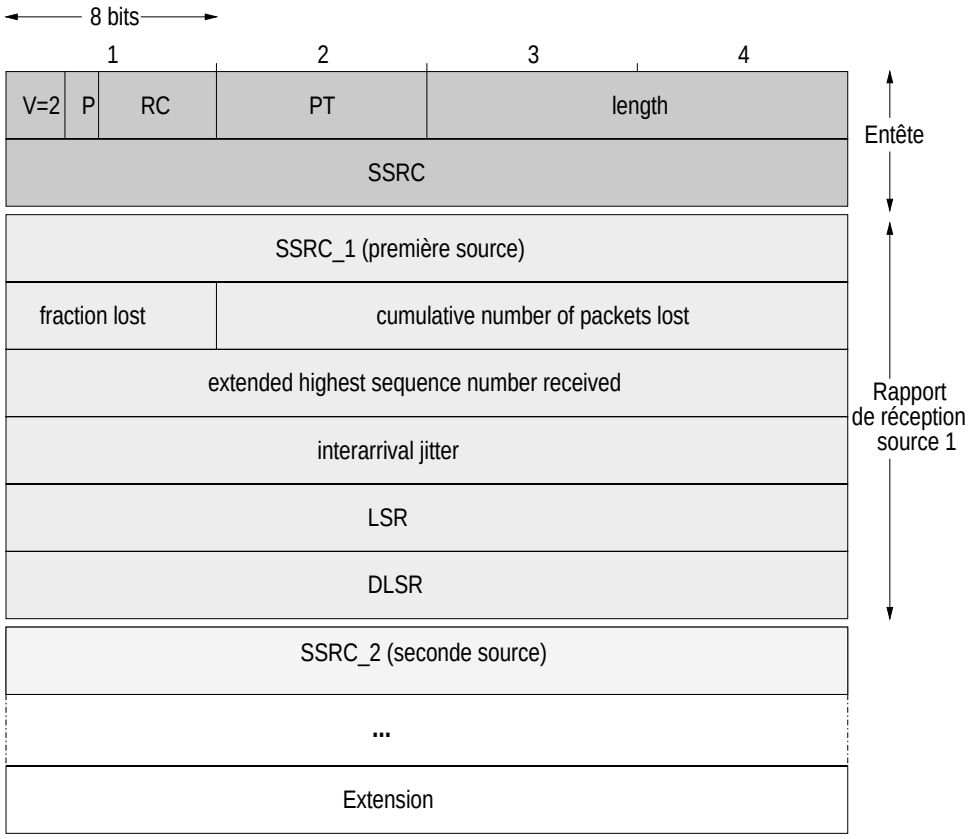

FIG. B.4 – *Format d'un paquet RTCP: Receiver Report*

Interarrival Jitter: Ce champ donne une estimation de la variance statistique du temps inter-arrivées des paquets de données, mesurée dans la même unité que celle de RTP timestamp. C'est cette variation de délais inter-arrivées que l'on nomme la gigue.

LSR (Last SR timestamp): Ce champ indique "l'heure" à laquelle a été envoyé le dernier rapport reçu de la source.

DLSR (Delay since Last SR): Ce champ indique le temps qui s'est écoulé depuis la réception du dernier rapport provenant de la source. Ce champ et le champ précédent interviennent dans le calcul par la source du temps dit "d'aller-retour" entre la source et le récepteur. Le calcul est détaillé dans la section B.2. Ce temps peut être utilisé par les applications dans le cadre des mécanismes de contrôle de flux et de congestion.

B.1.4.2 Format d'un Sender Report

Ces rapports sont des rapports envoyés par la source (ou les sources). Or, comme une source peut à la fois être un récepteur, les Sender Reports peuvent contenir, comme

le montre la figure B.5, des blocs de réception consitués des mêmes champs présents dans un Receiver Report (voir figure B.4). Ainsi, ces paquets ont la double fonction de rapport de source et de rapport de réception. Nous ne décrirons ici que la partie relative aux informations fournies sur la source elle-même. Les champs illustrés sur la figure B.5 ont la sémantique suivante:

NTP timestamp: Ce champ indique la date absolue du système en utilisant le format préconisé dans NTP (*Network Time Protocol*) [Mil92], qui est donnée en secondes et a comme origine le 1er janvier 1900. Ce champ représente l'heure à laquelle ce rapport a été envoyé. C'est ce champ qui permet d'évaluer toutes les mesures statistiques relatives au temps comme la gigue, le délai et le temps d'aller- retour vu plus haut et décrit en B.2. Ce champ est décomposé en une partie entière et une partie fractionnaire, chacune étant codée sur 32 bits.

RTP timestamp: Il correspond à la même heure que NTP timestamp, mais dans la même unité et avec le même décalage aléatoire que pour le RTP timestamp des paquets de données. Il est notamment utilisé dans le calcul de la gigue.

B.2 Algorithme de calcul de RTT

Le principe de base consiste à évaluer le temps entre l'émission d'un paquet de contrôle et la réception de l'accusé de réception correspondant à ce paquet. Toutefois, la politique de RTP consiste à ne pas engorger le réseau de feedback. La norme précise que ceux-ci ne doivent pas occuper plus de 5% de la bande passante réservée à l'application.

C'est pourquoi, sur l'envoi d'un Sender Report le récepteur n'envoie pas d'accusé de réception immédiatement mais attend son prochain envoi de rapport de réception (Sender ou Receiver Report) pour fournir les informations nécessaires au calcul de RTT.

Les champs utiles pour ce calcul sont les champs LSR (Last Sender Report), qui indiquent l'heure d'envoi du dernier Sender Report reçu, et le champ DLSR (Delay Since Last Sender Report), qui indique le temps qui s'est écoulé depuis le dernier Sender Report reçu. La valeur de LSR (sur 32 bits) est déterminée à partir de l'estampille NTP (NTP timestamp) qui est exprimée en secondes et codée sur 64 bits. Elle correspond au 16 bits de poids faible de sa partie entière et au 16 bits de poids fort de sa partie fractionnaire. L'unité du champ DLSR est le 1/65536 secondes.

Ainsi, si la source reçoit un rapport à l'instant T le temps d'aller-retour peut se calculer en faisant $(T - LSR - DLSR)$ (voir figure B.6).

Du fait de l'asymétrie des liens sous-jacents, en termes de temps de transit, la valeur calculée n'est qu'approximative.

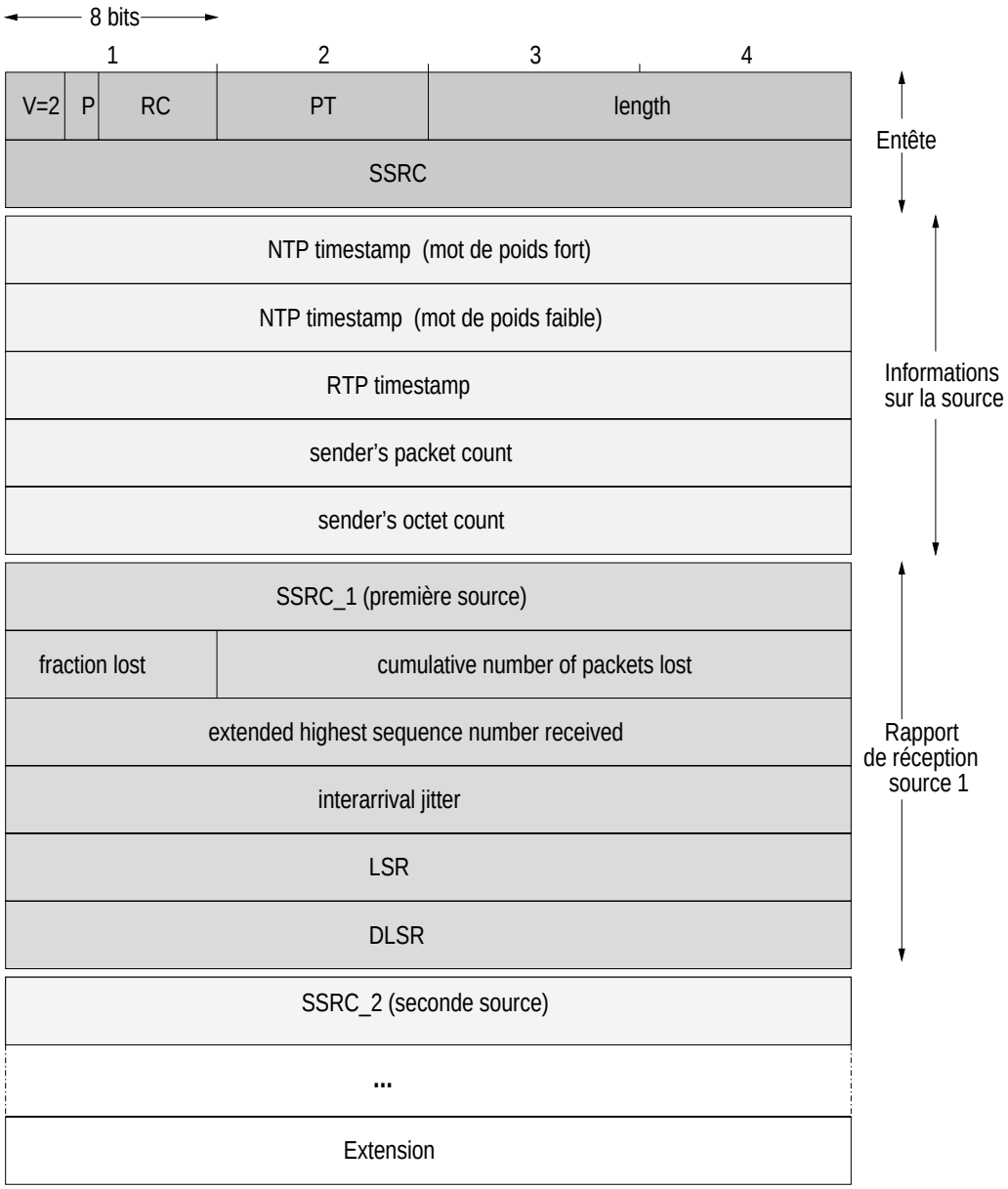

FIG. B.5 – *Format d'un paquet RTCP: Sender Report*

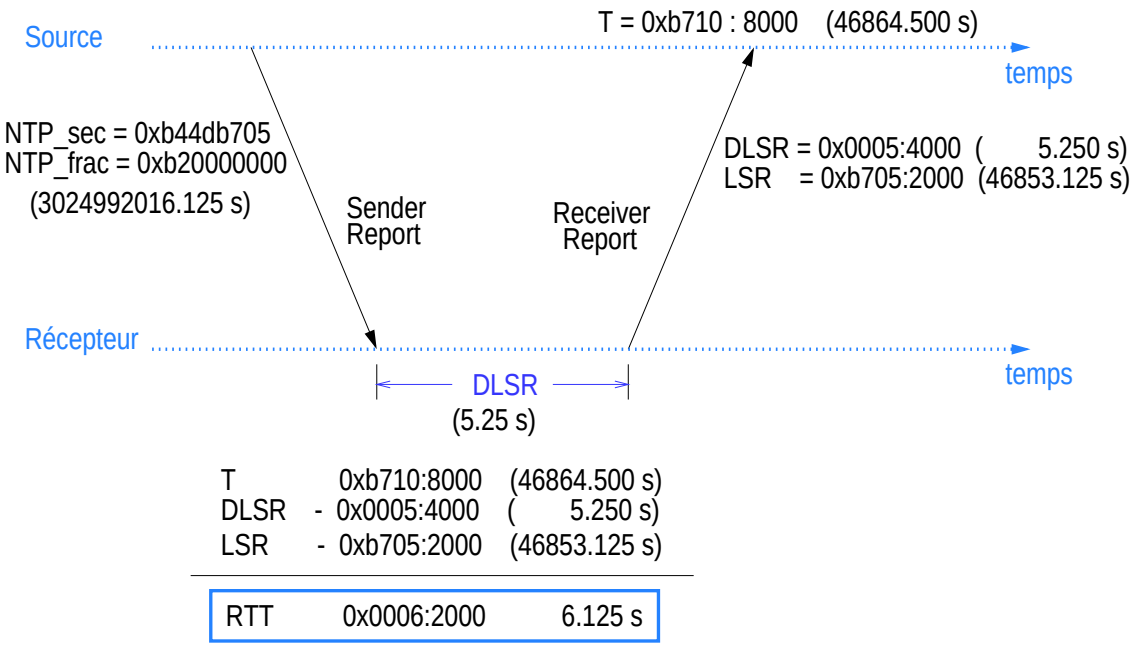

FIG. B.6 – *Exemple de calcul du Round-Trip Time (RTT) avec RTCP*

Annexe C

Construction d'un schéma de Lifting

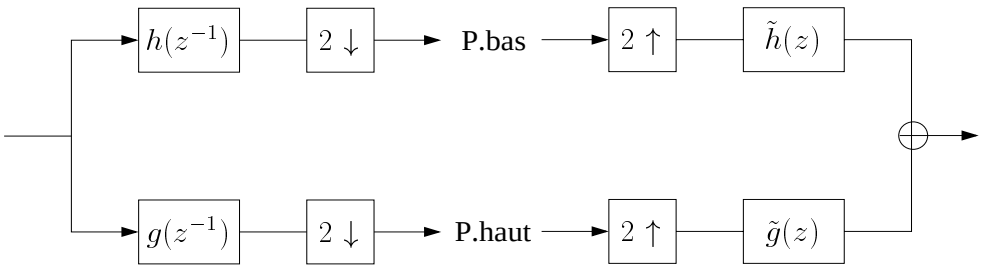

FIG. C.1 – *Schéma classique d'analyse-synthèse.*

Le but de cette annexe est de montrer comment on peut obtenir un schéma de lifting, à savoir les différents pas de lifting p_i (*Predict*) et u_i (*Update*), à partir d'un schéma classique de filtrage convolutif. La transformation classique utilise deux filtres h (passe-bas) et g (passe-haut) pour l'analyse, chaque filtrage étant suivi d'un sous-échantillonnage par deux. La transformation inverse consiste en premier lieu à suréchantillonner les entrées, puis à appliquer deux filtres $\tilde{h}$ (passe-bas) et $\tilde{g}$ (passe-haut) pour la synthèse. La reconstruction parfaite est assurée si les deux équations suivantes sont vérifiées:

$$\begin{cases} \frac{1}{2}\left[\tilde{g}(z)g(z) + \tilde{h}(z)h(z)\right] = I \\ \frac{1}{2}\left[\tilde{g}(z)g(-z) + \tilde{h}(z)h(-z)\right] = 0. \end{cases} \tag{C.1}$$

Dans le cas de filtres biorthogonaux ou duaux cette propriété est toujours vérifiée.

C.1 Représentation polyphase

Du fait des décimations par deux, on remarque qu'il est possible de réécrire h et g sous une forme polyphase par :

$$h(x) = h_e x_e + h_o x_o \tag{C.2}$$

$$g(x) = g_e x_e + g_o x_o \tag{C.3}$$

avec h_e, g_e les coefficients pairs des filtres h et g, et h_o et g_o respectivement les coefficients impairs.

La représentation polyphase de h et g nous permet d'écrire les deux matrices polyphases duales pour l'analyse et la synthèse du signal P et $\tilde{P}$:

$$P = \begin{pmatrix} h_e & h_o \\ g_e & g_o \end{pmatrix} \tag{C.4}$$

et

$$\tilde{P} = \begin{pmatrix} \tilde{h}_e & \tilde{g}_e \\ \tilde{h}_o & \tilde{g}_o \end{pmatrix} \tag{C.5}$$

La figure C.2 illustre la transformée correspondante avec les matrices polyphases.

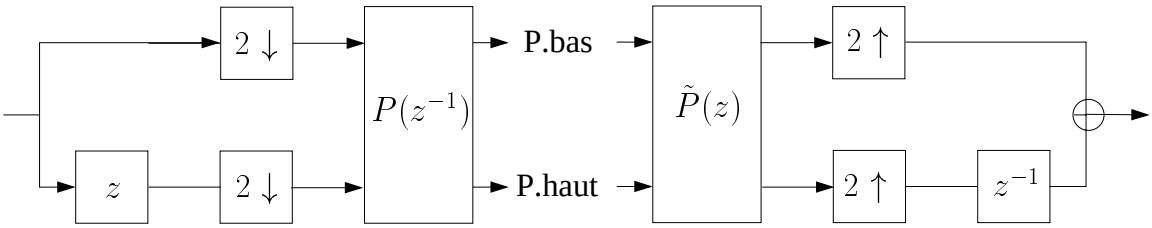

FIG. C.2 – *Représentation polyphase : analyse et synthèse.*

C.2 Factorisation lifting pour bancs de filtres à 2 bandes

A partir des conditions définies par les équations (C.1), on en déduit que la reconstruction sera parfaite lorsque $P(z^{-1})\tilde{P}(z) = I$. On montre alors que l'on peut factoriser la matrice polyphase pour la mettre sous la forme de l'équation C.6. Les matrices triangulaires supérieures obtenues représentent les pas de lifting, tandis que les triangulaires inférieures représentent les pas de lifting duaux :

$$\tilde{P} = \begin{pmatrix} K & 0 \\ 0 & 1/K \end{pmatrix} \times \prod_{i=0}^{M} \begin{pmatrix} 1 & u_i(z) \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ p_i(z) & 1 \end{pmatrix} \tag{C.6}$$

avec K et $1/K$ des facteurs de mise à l'échelle.

C.3 Exemples de factorisation lifting

Nous donnons ici les factorisations *lifting* correspondant à trois bancs de filtres classiquement utilisés dans la littérature liée au traitement de l'image. Les filtres traités ici sont :

- Haar : filtres 2-bandes, taille 2, orthogonaux, [Haa10],
- DAUB53 : filtres 2-bandes, 5-3, biorthogonaux, [CDF92],
- DAUB97 : filtres 2-bandes, 9-7, biorthogonaux, [CDF92, ABD92].

Les tables ci-dessous donnent respectivement les filtres de convolutions associés à ces trois bancs de filtres ainsi que la factorisation 'lifting" associée.

Filtres de convolution

$$\text{Haar} \quad \begin{cases} h(z) &= \frac{1}{\sqrt{2}}(1+z^{-1}) \\ g(z) &= \frac{-1}{\sqrt{2}}(1-z^{-1}) \end{cases}$$

$$\text{DAUB53} \quad \begin{cases} h(z) &= \frac{-1}{4\sqrt{2}}(1+z^{-4}) + \frac{2}{4\sqrt{2}}(z^{-1}+z^{-3}) + \frac{6}{4\sqrt{2}}z^{-2} \\ g(z) &= \frac{1}{2\sqrt{2}}(1+z^{-2}) - \frac{2}{4\sqrt{2}}z^{-1} \end{cases}$$

$$\text{DAUB97} \quad \begin{cases} h(z) &= 0.03783(1+z^{-8}) - 0.02385(z^{-1}+z^{-7}) - 0.1106(z^{-2}+z^{-6}) \\ &\quad +0.3774(z^{-3}+z^{-5}) + 0.8527z^{-4} \\ g(z) &= 0.06454(1+z^{-6}) - 0.04069(z^{-1}+z^{-5}) - 0.4181(z^{-2}+z^{-4}) \\ &\quad +0.7885z^{-3} \end{cases}$$

Factorisations Lifting

$$\text{Haar} \quad \left\{ \tilde{P}(z) = \mathbf{p_1} : (1-\sqrt{2}).\mathbf{u_0} : (\tfrac{1}{\sqrt{2}}).\mathbf{p_0} : (1-\sqrt{2}) \right.$$

$$\text{DAUB53} \quad \left\{ \tilde{P}(z) = \mathbf{S_k} : (\sqrt{2}).\mathbf{S_{1/k}} : (\tfrac{1}{\sqrt{2}}).\mathbf{u_0} : (\tfrac{1}{4}[1+z^{-1}]).\mathbf{p_0} : (\tfrac{-1}{2}[z+1]) \right.$$

$$\text{DAUB97} \quad \begin{cases} \tilde{P}(z) = \mathbf{S_k} : (1.1496044).\mathbf{S_{1/k}} : (0.86986445) \\ \qquad .\mathbf{u_1} : (0.44350685[1+z^{-1}]).\mathbf{p_1} : (0.88291108[z+1]) \\ \qquad .\mathbf{u_0} : (-0.052980119[1+z^{-1}]).\mathbf{p_0} : ([-1.5861343[z+1]) \end{cases}$$

La notation $\mathbf{p_i}$: () correspond au $i$ème pas de lifting de *Prédiction* (*Predict*) et $\mathbf{u_i}$: () au $i$ème pas de lifting de *Mise à jour* (*Update*). Les $\mathbf{S_k}$ et $\mathbf{S_{1/k}}$ correspondent aux facteurs de mise à l'échelle de l'équation C.6.

Publications

Journaux :

1. J. Viéron and C. Guillemot. - Real-time constrained TCP-compatible rate control for video over the Internet. - *To be published in IEEE Transactions on Multimedia*, 2003.
2. J. Viéron, T. Turletti, K. Salamatian and C. Guillemot. - Source and channel adaptive rate control for multicast layered video transmission based on a clustering algorithm. - *Submitted to EURASIP Journal on Applied Signal Processing*, 2002.

Congrès Internationaux :

1. J. Viéron and C. Guillemot - Low-rate FGS video compression based on motion-compensated spatio-temporal wavelet analysis. -*In proceedings of the SPIE International Conference on Visual Communication and Image Processing*, VCIP'03, invited paper, July 2003, Lugano, Switzerland.
2. J. Viéron, C. Guillemot and S. Pateux- Contributions to Scalable Video Coding based on spatio-temporal wavelet decomposition. -*ISO/IEC JTC1/SC29/WG11-MPEG2002*, December 2002, Japan.
3. J. Viéron, C. Guillemot, and S. Pateux. - Motion compensated 2D+t wavelet analysis for low rate FGS video compression. - *In proceedings of the International Workshop on Digital Communications*, IWDC'02, invited paper, September 2002, Capri, Italy.
4. J. Viéron, T. Turletti, X. Hénocq, C. Guillemot, and K. Salamatian. - TCP-compatible rate control for FGS layered multicast video transmission based on a clustering algorithm. - *In proceedings of the International Symposium on Circuits And Systems*, ISCAS'02, invited paper, May 2002, Scottsdale,Usa.
5. J. Viéron and C. Guillemot. - Source adaptive TCP-compatible rate control for video over the Internet. - *In proceedings of the IEEE International Conference on Image Processing*, ICIP'01, October 2001,Thessaloniki, Greece.
6. F. Le Léannec, J. Viéron, X. Hénocq and C. Guillemot. - Hybrid sender and receiver driven rate control in multicast layered video transmission. - *In proceedings of the IEEE International Conference on Image Processing*, ICIP'00, vol.3, pp.532-535, September 10-13, 2000, Vancouver, Canada.

Congrès Nationaux :

1. J. Viéron, C. Guillemot et H. Nicolas. - Compression vidéo FGS à bas débit basée sur une décomposition ondelettes 2D+t. - *CORESA 2003 (COmpression et REprésentation des SIgnaux Audiovisuels)*, Janvier 2003, Lyon, France.
2. J. Viéron et C. Guillemot. - Contrôle de congestion TCP-compatible pour transmission vidéo sur l'Internet. - *CORESA 2001 (COmpression et REprésentation des SIgnaux Audiovisuels)*, Novembre 2001, Dijon, France.

Bibliographie

[14498] 14496-2 (ISO/IEC). – *Coding of audio visual objects : Visual.* – ISO/IEC JTC1/SC29/WG11, Tokyo, March 1998.

[ABD92] Antonini (M.), Barlaud (M.) et Daubechies (I.). – Image coding using wavelet transform. *IEEE Transactions on Image Processing*, vol. 1, n° 2, April 1992, pp. 205–220.

[ACS98] Agrawal (P.), Chen (J-C.) et Sreenan (C.J.). – Use of statistical methods to reduce delays for media playback buffering. *In : Proceedings of IEEE International Conference on Multimedia Computing and Systems*, pp. 259–263. – Austin, Texas, USA, 28 June 1998.

[ASH87] Adelson (E.H.), Simoncelli (E.) et Hingorani (R.). – Orthogonal pyramid trnsform for image coding. *In : Proceedings of SPIE International Conference on Visual Communication and Image Processing, VCIP'87*, pp. 50–59. – Cambridge, MA, USA, October 1987.

[AVS99] Albuquerque (C.), Vickers (B.J.) et Suda (T.). – An end-to-end source-adaptive multi-layered multicast (samm) algorithm. *In : Proceedings Packet Video Workshop, PV'99.* – New York City, NY, USA, April 1999.

[BA83] Burt (P.) et Adelson (E.). – The laplacian payramide as compact image code. *IEEE Transactions on Communications*, vol. 31, 1983, pp. 482–540.

[Ba98] Bormann (C.) et al. – RTP payload format for the 1998 version of ITU-T H.263 video (H.263+). *Request for Comments 2429, IETF Network Working Group*, October 1998.

[BB01] Bansal (D.) et Balakrishnan (H.). – Binomial congestion control algorithms. *In : Proceedings of the Conference on Computer Communications, IEEE Infocom'01*, pp. 631–640. – Anchorage, AK, USA, April 2001.

[BBFPP01] Bottreau (V.), Bénetière (M.), Felts (B.) et Pesquet-Popescu (B.). – A fully scalable 3D subband video codec. *In : Proceedings of IEEE International Conference on Image Processing, ICIP'01.* – Thessaloniki, Greece, October 2001.

[BBFS01] Bansal (D.), Balakrishnan (H.), Floyd (S.) et Shenker (Scott). – Dynamic behavior of slowly-reponsive congestion control algorithms. *In : Proceedings of Conference of the Special Interest Group on data COMMunnication, ACM SIGCOMM'01.* – San Diego, CA, USA, 27-31 August 2001.

[BCC+98] Braden (B.), Clark (D.), Crowcroft (J.), Davie (B.), Deering (S.), Estrin (D.), Floyd (S.), Jacobson (V.), Minshall (G.), patridge (C.), Peterson (L.), Ramakrishnan (K.), Shenker (S.), Wroclawski (J.) et Zhang (J.L.). – Recommendations on queue management and congestion avoidance in the internet. *RFC-2309*, April 1998.

[BFH+00] Byers (J.), Frumin (M.), Horn (G.), Luby (M.), Mitzenmacher (M.), Roetter (A.) et Shave (W.). – FLID-DL: Congestion control for layered multicast. *In : Proceedings of the Second International Workshop on Networked Group Communication, NGC'00*, pp. 71–81. – Stanford, CA, USA, November 2000.

[BFPT99] Bolot (J.C.), Fosse-Parisis (S.) et Towsley (D.). – Adaptive fec-based error control for internet telephony. *In : Proceedings of the Conference on Computer Communications, IEEE Infocom'99*, pp. 1453–1460. – NY, March 1999.

[BH00] Bacelli (F.) et Hong (D.). – *TCP is Max-Plus Linear.* – Rapport technique n° TR-3986, France, INRIA, August 2000.

[Bie88] Bierling (M.). – Displacement estimation by hierarchical blockmatching. *In : Proceedings of SPIE International Conference on Visual Communication and Image Processing, VCIP'88*, pp. 942–951. – Cambridge, MA,USA, November 1988.

[BT98] Bolot (J.) et Turletti (T.). – Experience with rate control mechanisms for packet video in the internet. *Computer Communication Review*, no28, 1998, pp. 4–15.

[BVG96] Bolot (J.C.) et Vega-Garcia (A.). – Control mechanisms for packet audio in the Internet. *In : Proceedings of the Conference on Computer Communications, IEEE Infocom'96*, pp. 232–239. – San Francisco, CA, April 1996.

[CAL96] Cheung (S.Y.), Ammar (M.H.) et Li (X.). – On the use of destination set grouping to improve fairness in multicast video distribution. *In : Proceedings of the Conference on Computer Communications, IEEE Infocom'96*, pp. 553–560. – San Francisco, CA, March 1996.

[CDF89] Cohen (A.), Daubechies (I.) et Feauveau (J.C.). – *Biorthogonal bases of compactly supported wavelets.* – Rapport technique n° TM 11217-900529-07, AT&T Bell Laboratories, 1989.

[CDF92] Cohen (A.), Daubechies (I.) et Feauveau (J.C.). – Biorthogonal bases of compactly supported wavelets. *Comm. Pure & Appl. Math 45*, 1992, pp. 485–560.

[CGM+01] Calvert (K.L.), Griffioen (J.), Mullins (B.), Sehgal (A.) et Wen (S.). – Concast: Design and implementation of an active network service. *IEEE Journal on Selected Area in Communications (JSAC)*, vol. 19(3), March 2001, pp. 720–733.

[CJ89] Chiu (D.) et Jain (R.). – Analysis of the increase/decrease algorithms for congestion avoidance in computer networks". *Journal of Computer Networks and ISDN*, vol. 17, n° 1, June 1989, pp. 1–14.

[CLG89] Chou (P.A.), Lookabaugh (T.) et Gray (R.M.). – Optimal pruning with applications to tree-structured source coding and modeling. *IEEE Transactions on Information Theory*, vol. 35, March 1989, pp. 299–315.

[CP96] Chen (Y.) et Pearlman (W.A.). – Three-dimensionnal subband coding of video using the zerotree method. *In : Proceedings of SPIE International Conference on Visual Communication and Image Processing, VCIP'96*, pp. 1302–1309. – Orlando, FL, USA, March 1996.

[CW99] Choi (S-J.) et Woods (J.). – Motion-Compensated 3-D Subband Coding of Video. *IEEE Transactions on Image Processing*, vol. 8, n° 2, February 1999, pp. 155–167.

[CW02a] Chen (P.) et Woods (J.W.). – *Comparaison of MC-EZBC and H.26L TML8 on Digital Cinema Test Sequences.* – ISO/IEC JTC1/SC29/WG11-MPEG2002/8130, Cheju Island, March 2002.

[CW02b] Chen (P.) et Woods (J.W.). – *Contributions to Interframe Wavelet and Scalable Video Coding.* – ISO/IEC JTC1/SC29/WG11- MPEG2002/9034, Shanghai, October 2002.

[DS98] Daubechies (I.) et Sweldens (W.). – Factoring wavelet transforms into lifting steps. *J. Fourier Anal. Appl.*, vol. 4, n° 3, 1998, pp. 245–267.

[ec00] editorial committee (JPEG-2000). – *JPEG-2000 image coding system.* – Rapport technique, ISO/IEC JTC1/SC29/WG1N1646, March 2000.

[EMD98] El-Marakby (R.) et D (Hutchison). – Scalability improvement of the real-time control protocol (rtcp) leading to management facilities in the internet. *In : Proceedings of the third IEEE Symposium on Computers and Communications, ISCC'98*, pp. 125–129. – Athens, Greece, June 1998.

[FF99] Floyd (S.) et Fall (K.). – Promoting the use of end-to-end congestion control on the internet. *IEEE/ACM Transactions on Networking*, August 1999.

[FHP00] Floyd (S.), Handley (M.) et Padhye (J.). – A comparison of equation-based and aimd congestion control. – May 2000. Available at http://www.aciri.org/tfrc/.

[FHP01] Floyd (S.), Handley (M.) et Padhye (J.). – Tcp friendly rate control (tfrc):protocol specification. *Internet-draft draft-ietf-tfrc-02.txt*, May 2001.

[FHPW00] Floyd (S.), Handley (M.), Padhye (J.) et Widmer (J.). – Equation-based congestion control for unicast applications. *In : Proceedings of Conference of the Special Interest Group on data COMMunnication, ACM SIGCOMM'00*, pp. 43–56. – Stockholm, Sweden, August 2000.

[Flo91] Floyd (S.). – Connections with multiple congested gateways in packet-switched networks part 1: One way traffic. *ACM Computer Communication Review*, vol. 21, n° 5, October 1991, pp. 30–47.

[Gho97] Gholmieh (R. A.). – *Multicast Multilayer Videoconferencing Enhancement of a Multilayer Codec Implementation of the Receiver Driven Layered Multicast Protocol.* – Thèse de PhD, Texas A and M University, December 1997.

[Gol98] Golestani (J.). – A class of end-to-end congestion control algorithms for the internet. *In : Proceedings of International Conference on Network Protocols, ICNP'98.* – Austin, Texas, USA, October 1998.

[Haa10] Haar (A.). – Zur theorie der orthogonalen funktionen-systeme. *Math. Annal.*, vol. 69, 1910, pp. 331–371.

[HF98] Handley (M.) et Floyd (S.). – Strawman specification for tcp friendly (reliable) multicast congestion control (tfmcc). – December 1998. Note to the Reliable Multicast Research Group mailing list.

[HOK99] Hsu (C.Y.), Ortega (A.) et Khansari (M.). – Rate-control for robust video transmission over burst-error wireless channels. *IEEE Journal On Selected Areas in Communications*, vol. 17, n° 5, May 1999, pp. 756–772.

[HPFW01] Handley (M.), Pahdye (J.), Floyd (S.) et Widmer (J.). – Tcp friendly rate control (tfrc): Protocol specification. *Internet draft draft-ietf-tsvwg-tfrc-03.txt, work in progress*, July 2001.

[HW99] Hsiang (S.T.) et Woods (J.). – Invertible three-dimensional analysis/synthesis system for video coding with half-pixel accurate motion compensation. *In : Proceedings of SPIE International Conference on Visual Communication and Image Processing, VCIP'99*, pp. 537–546. – San Jose, CA, USA, February 1999.

[HW00a] Hsiang (S.T.) et Woods (J.W.). – Embedded image coding using zeroblocks of subband/wavelet coefficients and context modeling. *In : Proceedings of IEEE International Symposium on Circuits and Systems, ISCAS'00*, pp. 662–665. – Geneva, Switzerland, May 2000.

[HW00b] Hsiang (S.T.) et Woods (J.W.). – Embedded video coding using motion compensated 3-d subband/wavelet filter bank. *In : Proceedings of Packet Video Workshop, PV'00.* – Sardinia, Italy, May 2000.

[IP99] Islam (A.) et Pearlman (W. A.). – An embedded and efficient low-complexity hierarchical image coder. *In : Proceedings of SPIE International Conference on Visual Communications and Image Processing, VCIP'99*, pp. 294–305. – San Jose, CA,USA, 1999.

[Jac88] Jacobson (V.). – Congestion avoidance and control. *Computer Communication Review*, vol. 18, n° 4, August 1988, pp. 314–329.

[Jac90] Jacobson (V.). – Berkeley evolution from 4.3-tahoe to 4.3-reno. *In : Proceedings of the Eighteenth Internet Engineering Task Force.* – University of British Columbia, Vancouver, Canada, September 1990.

[JE96] Jacobs (S.) et Eleftheriadis (A.). – Providing video services over networks without quality of service guarantees. *In : Proceedings of WWW consortium Workshop on Real-time Multimedia and the Web, RTMW'96.* – INRIA-Sophia Antipolis, France, October 1996.

[Joh80] Johnston (J.D.). – A filter family designed for use in quadrature mirror filter banks. *In : Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP'80*, pp. 291–294. – Denver, Colorado, USA, April 80.

[KHFP02] Kohler (E.), Handley (M.), Floyd (S.) et Padhye (J.). – Datagram congestion control protocol (DCCP). *IETF Internet Draft draft-ietf-dccp-spec-00*, October 2002.

[Koo77] Koo (D.). – *Elements of optimization.* – Springer-Verlag, 1977.

[KP97] Kim (B.J.) et Pearlman (W.A.). – An embedded wavelet video coder using three-dimensional set partitioning in hierachical trees (spiht). *In : Proceedings IEEE Data Compression Conference, DCC'97*, pp. 251–260. – Snowbird, UT, USA, 1997.

[KV88] Karlsson (G.) et Vetterli (M.). – Three-dimensional subband coding of video. *In : Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'88*, pp. 1100–1103. – New York City, NY, USA, April 1988.

[KXP00] Kim (B.J.), Xiong (K.Z.) et Pearlman (W.A.). – Low bit-rate scalable video coding with 3D set partitioning in hierachical trees (3d-spiht). *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 10, n° 8, December 2000, pp. 1374–1387.

[LB00a] Legout (A.) et Biersack (E. W.). – Pathological behaviors for RLM and RLC. *In : Proceedings of International Conference on Network and Operating System Support for Digital Audio and Video, NOSSDAV'00*, pp. 164–172. – Chapel Hill, North Carolina, USA, 2000.

[LB00b] Legout (A.) et Biersack (E. W.). – PLM: Fast convergence for cumulative layered multicast transmission schemes. *In : Proceedings of ACM SIGMETRICS'00*, pp. 13–22. – Santa Clara, CA, USA, 2000.

[LG02] Luby (M) et Goyal (V.). – Wave and equation based rate control building block. *IETF Internet Draft draft-ietf-rmt-bb-webrc-01*, June 2002.

[LGSH02] Luby (M.), Goyal (V.), Skaria (S.) et Horn (G.B.). – Wave and equation based rate control using multicast round trip time. *In : Proceedings of Conference of the Special Interest Group on data COMMunnication, ACM SIGCOMM'02.* – Pittsburgh, PA, USA, August 2002.

[LLL+01] Luo (L.), Li (J.), Li (S.), Zhuang (Z) et Zhang (Y-Q.). – Motion compensated lifting wavelet and its application in video coding. *In : Proceedings of IEEE International Conference on Image Processing, ICIP'01.* – Thessaloniki, Greece, October 2001.

[LM86] Lemarié (P.G.) et Meyer (Y.). – Ondelettes et bases hiltertiennes. *Revista Matematica Iberoamericana*, vol. 2, 1986, pp. 1–18.

[LM01] Liu (J.) et Moulin (P.). – Information-theoretic analysis of interscale and intrascale dependencies between image wavelet coefficients. *IEEE Transactions on Image Processing*, vol. 10, n° 10, November 2001, pp. 1647–1658.

[LPGLA98] Levine (B. N.), Paul (S.) et Garcia-Luna-Aceves (J. J.). – Organizing multicast receivers deterministically by packet-loss correlation. *In : Proceedings of the sixth ACM international conference on Multimedia.* – Bristol, UK, 1998.

[LS99] Leon (P. De) et Sreenan (C.). – An adaptive predictor for media playout buffering. *In : Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP'99*, pp. 3097–3100. – Phoenix, AZ, USA, March 1999.

[LTG99] Léannec (F. Le), Toutain (F.) et Guillemot (C.). – Packet loss resilient MPEG-4 compliant video coding for the internet. *Journal Image Communication, (Special Issue On "Real-time video over the Internet")*, vol. 15, n° 1-2, September 1999, pp. 35–56.

[LWCP96] Luo (J.), Wang (X.), Chen (C.W.) et Parker (K.J.). – Volumetric medical image compression with three-dimensional wavelet transform and octave zerotree coding. *In : Proceedings of SPIE International Conference on Visual Communication and Image Processing, VCIP96*, pp. 579–590. – Orlando, FL, USA, March 1996.

[Mal89] Mallat (S.). – A theory for multiresolution signal decomposition: the wavelet representation. *IEEE Transactions on Pattern Analysis and Machine intelligence*, vol. 11, n° 7, July 1989, pp. 674–693.

[MBHW01] Marpe (D.), Blatterman (G.), Heising (G.) et Wiegand (T.). – *Further results for CABAC entropy coding scheme.* – Rapport technique, Doc. VCEG-M59, VCEG 13th meeting, April 2001.

[MF97] Mahdavi (J.) et Floyd (S.). – TCP-friendly unicast rate-based flow control. – January 1997. Technical note sent to the end2end-interest mailing list.

[Mil92] Mills (D.). – Network time protocol (v3). *RFC 1305, IETF Internet Draft*, April 1992.

[MSMO97] Mathis (M.), Semke (J.), Mahdavi (J.) et Ott (T.). – The macroscopic behavior of the tcp congestion avoidance algorithm. *Computer Communication Review*, vol. 27, n° 3, July 1997.

[MVJ97] McCanne (S.), Vetterli (M.) et Jacobson (V.). – Low-complexity video coding for receiver-driven layered multicast. *IEEE Journal on Selected Areas In Communications*, vol. 15, n° 6, August 1997, pp. 983–1001.

[NS2] *The network simulator - ns 2.* – Rapport technique. Available at http://www.isi.edu/nsnam/ns.

[Ohm92] Ohm (J-R.). – Temporal domain subband video coding with motion compensation. *In : Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'92*, pp. 229–232. – San Francisco, USA, 1992.

[Ohm94] Ohm (J-R.). – Three-dimensional subband coding with motion compensation. *IEEE Transactions on Image Processing*, vol. 3, n° 5, November 1994.

[OHR02] Ohm (J.R.), Hanke (K.) et Rusert (T.). – *Improved capabilities of motion-compensated wavelet filters.* – ISO/IEC JTC1/SC29/WG11-MPEG2002/M8360, Fairfax, May 2002.

[OKM] Ott (T.), Kemperman (J.) et Mathis (M.). – The stationary behavior of ideal tcp congestion avoidance. – URL ftp://ftp.bellcore.com/pub/tjo/TCPwindow.ps.

[OkM96] Ott (T.), kemperman (J.) et Mathis (M.). – Window size behavior in tcp/ip with constant loss probability". *In : DIMACS Workshop on Performance of Realtime Applications on the Internet.* – Plainfield, NJ,USA, November 1996.

[Pax97] Paxon (V.). – *Measurements and Analysis of End-to-End Internet Dynamics.* – Thèse de PhD, Lawrence Berkeley National Laboratory, University of California, April 1997.

[PFTK98] Padhye (J.), Firoiu (V.), Towsley (D.) et Kurose (J.). – Modeling tcp thoughput: a simple model and its empirical validation. *In : Proceedings of Conference of the Special Interest Group on data COMMunnication, ACM SIGCOMM'98.* – University of British Columbia, Vancouver, Canada, August 1998.

[PINS02] Pearlman (W. A.), Islam (A.), Nagaraj (N.) et Said (A.). – Effficient low-complexity image coding with a set-partitioning embedded block coder. *submitted to IEEE Transactions on Circuits and Systems for Video Technology*, August 2002.

[PJF95] Podilchuck (C.I.), Jayant (N.S.) et Farvardin (N.). – Three-dimensionnal subband coding of video. *IEEE Transactions on Image Processing*, vol. 4, n° 2, February 1995, pp. 125–139.

[PKTK99a] Padhye (J.), Kurose (J.), Towsley (D.) et Koodli. (R.). – An empirical study of client interactions with a continuous-media courseware server. *IEEE Internet Computing*, April 1999.

[PKTK99b] Padhye (J.), Kurose (J.), Towsley (D.) et Koodli. (R.). – A model based tcp-friendly rate control protocol. *In : Proceedings of International Conference on Network and Operating System Support for Digital Audio and Video, NOSSDAV'99.* – Basking Ridge, NJ, USA, June 1999.

[Pos80] Postel (J.). – User datagram protocol. *RFC 791*, August 1980.

[Pos81] Postel (J.). – Transmission control protocol. *RFC 793*, September 1981.

[PPB01] Pesquet-Popescu (B.) et Bottreau (V.). – Three-dimensional lifting schemes for motion compensated video compression. *In : Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP'01.* – Salt Lake City, UT, USA, May 2001.

[PPBB00] Pesquet-Popescu (B.), Bénetière (M.) et Bottreau (V.). – Embedded color coding for scalable 3d wavelet-based video compression. *In : Proceedings of SPIE International Conference on Visual Communication and Image Processing, VCIP'00.* – Perth, Australia, June 2000.

[PSLB97] Paul (S.), Sabnani (K.K.), Lin (J.C.) et Bhattacharyya (S.). – Reliable multicast transport protocol (rmtp). *IEEE Journal On Selected Areas in Communications*, vol. 15, n° 3, April 1997, pp. 407–421.

[PYM99] Podolsky (M.), Yano (K.) et McCanne (S.). – A rtcp-based retransmission protocol for unicast rtp streamming multimedia. *Internet-draft draft-podolsky-avt-rtpx01.txt*, October 1999.

[Ram93] Ramchandran (K.). – *Joint optimization techniques in image and video coding with applications to multiresolution digital broadcast.* – Thèse de PhD, Columbia University, 1993.

[RC99] Radha (H.) et Chen (Y.). – Fine granular scalable video for packet networks. *In : Proceedings of Packet Video Workshop, PV'99.* – Columbia University, NY, USA, April 1999.

[RCL97] Ribas-Corbera (J.) et Lei (S.). – *Rate Control for low-delay video communications.* – Rapport technique, ITU-Telecommunications Standardization, Video coding expert group, Doc Q15-A20, June 1997.

[REH99] Rejaie (R.), Estrin (D.) et Handley (M.). – Quality adaptation for congestion controlled video playback over the internet. *In : Proceedings of Conference of the Special Interest Group on data COMMunnication, ACM SIGCOMM'99.* – Cambridge, UK, September 1999.

[RHE99] Rejaie (R.), Handley (M.) et Estrin (D.). – RAP: An end-to-end rate-based congestion control mechanism for realtime streams in the Internet. *In : Proceedings of the Conference on Computer Communications, IEEE Infocom'99.* – San Diego, USA, June 1999.

[Riz00] Rizzo (L.). – pgmcc: a TCP-friendly single-rate multicast congestion control scheme. *In : Proceedings of Conference of the Special Interest Group on data COMMunnication, ACM SIGCOMM'00*, pp. 17–28. – Stockolm, Sweden, August 2000.

[RKTS94] Ramjee (R.), Kurose (J.F.), Towsley (D.F.) et Schulzrinne (H.). – Adaptive playout mechanisms for packetized audio applications in wide-area networks. *In : Proceedings of the Conference on Computer Communications, IEEE Infocom'94*, pp. 680–688. – Toronto, Canada, June 1994.

[RO00] Rhee (I.) et Ozdemir (V.). – *TEAR: TCP Emulation At Receivers - flow control for multimedia streaming.* – Rapport technique, Technical report NCSU, April 2000.

[RR99] Ramesh (S.) et Rhee (I.). – *Issues in Model-based Flow Control.* – Rapport technique n° TR 99-15, North Carolina State University, Department of Computer Science, 1999.

[Sa99] Schultzrinne (H.) et al. – RTP profile for audio and video conferences with minimal control. *RFC 1890, IETF Internet Draft draft-avt-profile-new-05*, February 1999.

[Sal99] Salamatian (K.). – Joint source-channel coding applied to multimedia transmission over lossy packet network. *In : Proceedings Packet Video Workshop, PV'99.* – New York City, NY, USA, April 1999.

[SB86] Smith (M.J.T.) et Barnwell (T.P.). – Exact reconstruction techniques for tree-structured subband coders. *IEEE Transactions on Image on Acoustics, Speech, and Signal Processing*, vol. 3, June 1986, pp. 434–441.

[SCFJ96] Schultzrinne (H.), Casner (S.), Frederick (R.) et Jacobson (V.). – RTP: A transport protocol for real-time applications. – Request for Comments 1889, IETF Network Working Group, January 1996.

[SCFJ98] Schultzrinne (H.), Casner (S.), Frederick (R.) et Jacobson (V.). – Rtp: A transport protocol for real-time applications. – IETF Internet Draft draft-avt-rtp-new-01, August 1998. Work in progress.

[SFC+00] Speakman (T.), Farinacci (D.), Crowcroft (J.), Gemmel (J.), Lin (S.), Tweedly (A.) et al. – *PGM Reliable Transport Protocol Specification.* – Internet draft, Internet Engineering Task Force, february 2000. draft-speakman-pgm-spec-06.txt.

[SFG01] Steinbach (E.), Faerber (N.) et Girod (B.). – Adaptive playout for low-latency video streaming. *In : Proceedings of IEEE International Conference on Image Processing, ICIP'01*, pp. 962–965. – Thessaloniki, Greece, October 2001.

[Sha93] Shapiro (J.M.). – Embedded image coding using zerotrees of wavelet coefficients. *IEEE Transactions on Signal Processing*, vol. 41, n° 12, December 1993, pp. 3445–3462.

[SP96] Said (A.) et Pearlman (W.A.). – A new, fast, and efficient image codec based on set partitionning in hierarchical trees. *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, n° 3, June 1996, pp. 243–249.

[SR00] Saparilla (D.) et Ross (K.W.). – Optimal streaming of layered video. *In : Proceedings of the Conference on Computer Communications, IEEE Infocom'00*, pp. 737–746. – Tel-Aviv, Israel, March 2000.

[SS98] Sisalem (D.) et Schulzrinne (H.). – The loss-delay based adjustment algorithm : A tcp-friendly adaptation scheme. *In : Proceedings of International Conference on Network and Operating System Support for Digital Audio and Video, NOSSDAV'98.* – Cambridge, UK, July 1998.

[ST01] Secker (A.) et Taubman (D.). – Motion-compensated highly scalable video compression using adaptive 3d wavelet transform based on lifting. *In : Proceedings of IEEE International Conference on Image Processing, ICIP'01.* – Thessaloniki, Greece, October 2001.

[Ste94] Stevens (W.R.). – *TCP-IP Illustrated, Volume 1, The Protocols.* – Addison-Wesley Professionnal Computing Series, 1994.

[Ste97] Stevens (W.). – Tcp slow start, congestion avoidance, fast retransmit, and fast recovery algorithms. *RFC 2001*, January 1997.

[Sul99] Sullivan (G.). – *Draft text of Recommendation H.263 version 2 ("H.263+") for decision.* – Rapport technique, ITU-T, March 1999.

[SW00a] Sisalem (D.) et Wolisz (A.). – Lda+: A tcp-friendly adaptation scheme for multimedia communication. *In : IEEE International Conference on*

Multimedia and Expo, ICME'00, pp. 1619–1622. – New York City, NY, USA, August 2000.

[SW00b] Sisalem (D.) et Wolisz (A.). – Mlda: A tcp-friendly congestion control framework for heterogeneous multicast environments. *In : Proceedings of International Workshop on Quality of Service, IWQoS'00.* – Pittsburgh, PA,USA, June 2000.

[Swe95] Sweldens (W.). – The lifting scheme : a new philosophy in biorthogonal wavelet constructions. *Wavelet Applications in Signal and Image Processing III, Proceedings SPIE 2569*, 1995, pp. 68–79.

[Tau00] Taubman (D.). – High performance scalable image compression with EBCOT. *IEEE Transactions on Image Processing*, vol. 9, n° 7, July 2000, pp. 1158–1170.

[TFPB98] Turletti (T.), Fosse-Parisis (S.) et Bolot (J.C.). – Experiments with a layered transmission scheme over the internet. *In : Proceedings of the Conference on Computer Communications, IEEE Infocom'98.* – San Francisco, April 1998.

[TPN94] Tan (T.K.), Pang (K.K.) et Ngan (K.N.). – A frequency scalable coding scheme employing pyramid and subband techniques. *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 4, n° 2, April 1994, pp. 203–207.

[TRK97] Tham (J.Y.), Ranganath (S.) et Kassim (A.A.). – Highly scalable wavelet-based video codec. *IEEE Journal on Selected Areas In Telecommunications, (Special Issue On "Very Low Bitrate Coding")*, 1997.

[TZ94] Taubmann (D.) et Zakhor (A.). – Multirate-3d subband coding of video. *IEEE Transactions on Image Processing*, vol. 3, n° 5, September 1994, pp. 572–588.

[TZ98] Tan (W.T.) et Zakhor (A.). – Internet video using error resilient scalable compression and cooperative transport protocol. *In : Proceedings of IEEE International Conference on Image Processing, ICIP'98*, pp. 458–462. – Chicago, IL, USA, October 1998.

[TZ99] Tan (D.) et Zakhor (A.). – Real-time internet video using error resilient scalable compression and tcp-friendly transport protocol. *IEEE Transactions on Multimedia*, vol. 1, n° 2, May 1999, pp. 172–186.

[VAS00a] Vickers (B. J.), Albuquerque (C.) et Suda (T.). – Source adaptive multilayered multicast algorithms for real-time video distribution. *IEEE/ACM Transactions on Networking*, vol. 8(6), December 2000, pp. 720–733.

[VAS00b] Vickers (Brett J.), Albuquerque (Célio) et Suda (Tatsuya). – Source-adaptive multilayered multicast algorithms for real-time video distribution. *IEEE/ACM Transactions on Networking*, vol. 8, n° 6, december 2000, pp. 720–733.

[VG03] Viéron (J.) et Guillemot (C.). – Real-time constrained tcp-compatible rate control for video over the internet. *To be published in IEEE Transactions On Multimedia*, 2003.

[VH90] Vetterli (M.) et Herley (C.). – Wavelets and filter banks: Relationships and new results. *In : Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'90*, pp. 1723–1726. – Albuquerque, USA, April 1990.

[VRC98] Vicisano (L.), Rizzo (L.) et Crowcroft (J.). – TCP-like congestion control for layered multicast data transfer. *In : Proceedings of the Conference on Computer Communications, IEEE Infocom'98*, pp. 996–1003. – San Francisco, USA, March 1998.

[WC02] Woods (J.W.) et Chen (P.). – *Improved MC-EZBC with Quarter-pixel Motion Vectors*. – ISO/IEC JTC1/SC29/WG11- MPEG2002/8366, Fairfax, May 2002.

[WCH02] Woods (J.W.), Chen (P.) et Hsiang (S-T.). – *Exploration Experimental Results and Software*. – ISO/IEC JTC1/SC29/WG11- MPEG2002/8524, Klagenfurt, July 2002.

[WH01] Widmer (J.) et Handley (M.). – Extending equation-based congestion control to multicast applications. *In : Proceedings of Conference of the Special Interest Group on data COMMunnication, ACM SIGCOMM'01.* – San Diego, USA, August 2001.

[WO86] Woods (J. W.) et O'Neil (S.D.). – Subband coding of images. *IEEE Transactions on Image on Acoustics, Speech, and Signal Processing*, vol. 34, n° 5, October 1986, pp. 1278–1288.

[WS95] Wright (G.R.) et Stevens (W.R.). – *TCP-IP Illustrated, Volume 2, The Implementation*. – Addison-Wesley Professionnal Computing Series, 1995.

[WXCM99] Wang (A.), Xiong (Z.), Chou (P.A.) et Mehrotra (S.). – Three-dimensional wavelet coding of video with global motion compensation. *In : Proceedings of IEEE International Conference on Data Compression, DCC'99*, pp. 404–413. – Snowbird, UT, USA, March 1999.

[WZ98] Wang (Y.) et Zhu (Q.F.). – Error control and concealment for video communication: A review. *Proceedings of the IEEE*, vol. 86, n° 5, May 1998, pp. 974–997.

[XLXZ00] Xu (J.), Li (S.), Xiong (Z.) et Zhang (Y.-Q.). – On boundary effects in 3-d wavelet video coding. *In : Proceedings of Symposium on Optical Science and Technology*. – San Diego, CA, USA, July 2000.

[XLZ00] Xu (J.), Li (S.) et Zhang (Y.-Q.). – A wavelet codec using 3-d escot. *In : Proceedings of IEEE-Pacific-Rim Conference on Multimedia, PCM'00*. – Sydney, Australia, December 2000.

[XXLZ01] Xu (J.), Xiong (Z.), Li (S.) et Zhang (Y.-Q.). – 3-d embedded subband coding with optimal truncation (3-D ESCOT). *Journal Applied and Computational Harmonic Analysis, (Special Issue On "Wavelet Applications in Engineering")*, vol. 10, May 2001, pp. 290–315.

[XXLZ02] Xu (J.), Xiong (Z.), Li (S.) et Zhang (Y.-Q.). – Memory-constrained 3D wavelet transform for video coding without boundary effects. *IEEE Tran-*

sactions Circuits and Systems for Video Technology, vol. 12, September 2002, pp. 812–818.

[YKT96] Yajnik (M.), Kurose (J.) et Towsley (D.). – Packet loss correlation in the mbone multicast network. *In : Proceedings IEEE Global Internet Conference*. – London,UK, November 1996.

[YL00] Yang (Y.) et Lam (S.). – *General AIMD Congestion Control*. – Rapport technique n° TR-2000-09, Austin, University of Texas, May 2000.

[ZPPPH02] Zhan (Y.), Picard (M.), Pesquet-Popescu (B.) et Heijmans (H.). – *Long temporal filters in lifting schemes for scalable video coding*. – ISO/IEC JTC1/SC29/WG11- MPEG2002/M8680, Klagenfurt, July 2002.

Abstract

This thesis deals with video data transmission over heterogeneous and time varying packet networks, such as the Internet. The goal of this work is the study of new models of fine grain scalable representation of video signals, as well as rate control (congestion control) mechanisms for its transmission over networks. First, we propose a new rate control algorithm for unicast applications coupling a source adaptive TCP-compatible congestion control protocol based on RTP/RTCP with a global source rate control model emcompassing timing and buffering models of the source, and this in order to minimize the expected distortion at the receiver. This algorithm allows to reduce very significantly the timeouts effects, hence to improve the quality of the decoded signal, for a comparable usage of the bandwidth. In contrast with previous works, the protocol proposed here is fully dedicated to the multimedia applications. Then, we have designed a new hybrid sender and receiver driven TCP-compatible rate control taken into account the source rate-distortion characteristics for multicast layered video transmission. This regulation leans on a concise representation of the networks state delivered by a mechanism of aggregation of the receivers reports and is made according to the optimization of a cost metric representation of the quality perceived by the overall receivers. After having studied the channel area aspects, we have focused on the design of a new finely scalable video compression algorithm allowing an easier adaptation of the compressed streams to varying network conditions. The proposed scheme rely on a motion compensated spatio-temporal wavelet analysis and aims low bitrate applications. The different techniques are integrated in acomplete coding architecture relying on a rate-constrained quadtree motion estimation and an EBCOT-3D coding of the spatio-temporal subbands.

Key words

Congestion control, rate control, Internet, TCP-compatible, unicast, multicast, video coding, real-time, fine granularity scalable video coding, spatio-temporal wavelet

Résumé

L'étude menée dans cette thèse s'inscrit dans le contexte general de la transmission de données vidéo temps-réel sur des réseaux de paquets hétérogènes aux caractéristiques variant dans le temps, tels que l'Internet. L'objectif de cette thèse concerne l'étude de nouveaux modèles de représentation scalable à grain fin de signaux vidéo et de techniques de régulation de débit (contrôle de congestion) associées pour la transmission. Dans ce cadre, nous proposons, tout d'abord, un nouvel algorithme de régulation de débit point-à-point couplant un protocole de contrôle de congestion TCP-compatible, basé sur le protocole RTP/RTCP, avec un modèle de régulation global intégrant les modèles de délais et de buffers de la source, dans le but de minimiser la distorsion du signal décodé au récepteur. Le modèle global proposé permet de réduire de manière significative les pertes dûes aux retards et donc de minimiser la distorsion tout en maximisant l'utilisation de la bande passante. A l'inverse des approches proposées dans la littérature le protocole développé est dédié à la transmission multimédia et prend en considération les différentes contraintes inhérentes à ce type de flux. Dans un second temps, nous avons développé un nouvel algorithme de contrôle de débit TCP-compatible hybride orienté émetteur-récepteur prenant en compte les caractéristiques débit-distorsion de la source pour la transmission multicast de vidéo en couches. Cette régulation s'appuie sur une représentation concise de l'état du réseau fournie par un mécanisme d'agrégation des rapports des récepteurs et se fait suivant un critère d'optimisation de la qualité perçue par l'ensemble des récepteurs. Après nous être intéressés à l'aspect canal, nous proposons l'architecture complète d'un nouvel algorithme de compression vidéo bas débit finement scalable permettant une régulation fine du débit de la source, non permise par les codeurs vidéo scalables standards. Le schéma de codage vidéo nouvelle génération proposé se base sur l'utilisation d'une décomposition ondelettes, compensée en mouvement, dans les dimensions spatiale et temporelle (2D+t). L'architecture repose sur une estimation de mouvement hiérarchique basée sur un quadtree contraint en débit. Les sous-bandes spatio-temporelles sont quant à elles codées entropiquement par un algorithme EBCOT-3D.

Mots clés

Contrôle de congestion, régulation de débit, Internet, TCP-compatible, point à point, multipoint, codage vidéo, temps-réel, codage scalable à granularité fine, ondelettes spatio-temporelles