Université de Nantes
–
École polytechnique de l'université de Nantes

Projet RIAM ArchiPEG

(convention ANR05RIAM014xx)

Lot 4.1 : Rapport d'étude sur les méthodes de conditionnement et de pré-analyse du flux vidéo

Olivier BROUARD
Laboratoire IRCCyN – équipe IVC
septembre 2006

Table des matières

Introduction générale 1

1 Le codeur H.264/AVC 3

1.1 Introduction . . . 3

1.2 Structure du codeur H.264/AVC . . . 3

1.2.1 Terminologie . . . 3

1.2.2 Le codec H.264 . . . 4

1.2.3 Les profils et niveaux . . . 6

1.3 Prédiction inter-images . . . 7

1.3.1 Gestion des images de référence . . . 7

1.3.2 Les tranches P . . . 8

1.3.3 Les tranches B . . . 11

1.4 Prédiction intra-image . . . 13

1.4.1 Prédiction des blocs 4×4 de luminance . . . 14

1.4.2 Prédiction des blocs 16×16 de luminance . . . 14

1.4.3 Prédiction des blocs de chrominance . . . 14

1.4.4 Codage des modes de prédiction . . . 15

1.5 Transformée et quantification . . . 16

1.5.1 Transformation . . . 17

1.5.2 Quantification . . . 17

1.6 Codage entropique . . . 17

1.6.1 Codage Exp-Golomb . . . 17

1.6.2 CAVLC . . . 18

1.6.3 CABAC . . . 18

1.7 Conclusion . . . 19

2 Optimisation du codeur H.264/AVC 21

2.1 Introduction . . . 21

2.2 Méthodes d'estimation de mouvement améliorées . . . 22

2.2.1 Recherche multi-résolutions . . . 22

2.2.2 Estimation du mouvement avec des recherches partielles . . . 24

2.2.3 Estimation du mouvement avec une nouvelle métrique . . . 28
2.3 Méthodes de réduction des modes de prédiction testés . . . 28
2.3.1 Sélection rapide du mode . . . 28
2.3.2 Décision pour les modes intra . . . 30
2.3.3 Décision pour les modes inter . . . 30
2.4 Choix des images références . . . 34
2.4.1 Recherche partielle . . . 34
2.4.2 Sélection des images de référence . . . 37
2.5 Conclusion . . . 38

3 H.264/SVC étendu à la graduabilité 39
3.1 Introduction . . . 39
3.2 Incorporation d'un schéma MCTF au sein du codeur H.264/AVC . . . 40
3.3 Extension du codeur H.264/AVC à la graduabilité . . . 42
3.3.1 Graduabilité temporelle . . . 42
3.3.2 Graduabilité en qualité (SNR) . . . 42
3.3.3 Graduabilité spatiale . . . 43
3.3.4 Combinaison des différentes graduabilités . . . 43
3.4 Conclusion . . . 43

4 Le schéma MCTF 45
4.1 Introduction . . . 45
4.2 La transformation en ondelettes . . . 45
4.2.1 La transformation en ondelettes 1-D . . . 45
4.2.2 La transformation en ondelettes 2-D . . . 47
4.2.3 Le schéma lifting . . . 49
4.3 Le filtrage temporel . . . 49
4.3.1 Principe . . . 50
4.3.2 Filtrage temporel et compensation du mouvement . . . 51
4.3.3 Modélisation lifting du filtrage court (Haar) . . . 53
4.3.4 Extension des schémas lifting au filtrage 5/3 . . . 54
4.3.5 Le filtrage temporel MCTF 1/3 . . . 56
4.4 Conclusion . . . 58

5 Objets vidéo et tubes spatio-temporels 59
5.1 Introduction . . . 59
5.2 Suivi temporel des objets dans une séquence vidéo . . . 59
5.2.1 Suivi des objets par mise en correspondance . . . 59

5.2.2 Suivi des objets par projection initialisation 61
5.3 Les tubes spatio-temporels 62
5.3.1 Approche par croissance de régions et fusionnement de tubes 63
5.4 Les objets vidéo 64
5.5 Conclusion 65

Conclusion **67**

Glossaire **69**

Bibliographie **70**

Table des figures

1.1 Schéma du codeur H.264/AVC [Richardson, 03]. . . . 4
1.2 Schéma du décodeur H.264/AVC [Richardson, 03]. . . . 5
1.3 Les profils dans H.264/AVC. . . . 6
1.4 Partitions de macroblocs : 16×16, 8×16, 16×8 et 8×8. . . . 9
1.5 Partitions de sous-macroblocs : 8×8, 4×8, 8×4 et 4×4 9
1.6 Partitions courantes et voisines 10
1.7 Prédiction bi-directionnelle de la tranche B [Flierl *et al.*, 03]. . . . 12
1.8 Mode temporel direct 13
1.9 Labellisation des échantillons de prédiction 4×4. . . . 14
1.10 Les modes de prédiction des blocs 4×4 de luminance [Richardson, 03]. . . . 14
1.11 Les modes de prédiction des blocs 16×16 de luminance [Richardson, 03]. . . . 15

2.1 Étapes de recherche pour les prédictions intra et inter avec 5 images de référence pour le codeur H.264/AVC. . . . 21
2.2 Modifications réalisées sur la transformation d'entiers. . . . 23
2.3 Formes de recherche pour une estimation du mouvement multi-résolutions. . . . 24
2.4 Processus de recherche de l'algorithme UMHexagonS avec une fenêtre de taille 32×32. 25
2.5 Forme de taille unitaire. . . . 27
2.6 Ilustration de la corrélation/continuité des vecteurs de mouvement. . . . 28
2.7 Définition de la compacité des vecteurs de mouvement d'un macrobloc. . . . 35

3.1 Sélection des vecteurs m_{P0} et m_{P1} ($m_{U0} = -m_{P0}$ et $m_{U1} = -m_{P1}$) pour l'étape de mise à jour. . . . 41
3.2 Concept général de l'algorithme de codage graduable pour la qualité (SNR). . . . 42

4.1 Représentation de l'image Lena à plusieurs résolutions. . . . 45
4.2 Bancs de filtres de synthèse et d'analyse pour une DWT 1-D. . . . 46
4.3 Décomposition ondelettes sur trois niveaux d'une image 2-D. . . . 48
4.4 Décomposition ondelettes sur trois niveaux de Lena en utilisant un banc de filtres (9,7). 48
4.5 Schéma fonctionnel général du processus de lifting. . . . 49
4.6 Décomposition sur quatre niveaux du GOP original. . . . 50

4.7 Prédiction du mouvement « en avant » entre deux images. 51
4.8 Traitement des zones recouvertes et découvertes par la méthode Ohm. 52
4.9 (a) Opérateur de prédiction dans le domaine spatio-temporel. La flèche en pointillés indique, les connexions multiples possibles du pixel n (b)Opérateur de mise à jour dans le domaine spatio-temporel. La flèche en pointillés indique, les connexions multiples possibles du pixel m 55
4.10 Schéma de décomposition temporelle pyramidale. 57

5.1 Illustration de la notion de tube spatio-temporel. 62
5.2 Mosaïque de la séquence *Lion* pour les images 0 à 190. 65

Liste des tableaux

1.1 Indice des images à court terme d'un DPB [Richardson, 03]. 8
1.2 Les modes de prédiction des blocs 4×4 de luminance [Richardson, 03]. 15
1.3 Les modes de prédiction des blocs 16×16 de luminance [Richardson, 03]. 16
1.4 Les modes de prédiction des blocs 8x8 de chrominance. 16

3.1 Décodages évalués pour le Call for Proposals : graduabilité spatiale, graduabilité temporelle et SNR. 39

Introduction générale

Les travaux présentés dans ce rapport ont été réalisés dans le cadre du projet RIAM ArchiPEG qui relève de la convention ANR05RIAM014xx. Ils correspondent à la première tâche du sous-projet 4 intitulé : Pré-analyse et conditionnement du flux vidéo en haute définition.

L'évolution des technologies des terminaux, des algorithmes de codage de la vidéo et des réseaux de communication vont permettre la mise en exploitation de nouveaux services audiovisuels tels la Télévision Numérique Terrestre et les réseaux mobiles de troisième génération. Les bandes passantes disponibles sur ces réseaux restent une ressource d'autant plus rare que l'étendue et le type des services visés sur ces plates-formes se diversifient. En effet, aux services de diffusion (un vers tous) s'ajoutent des services personnalisés (à la demande). Par ailleurs, la télévision haute définition va progressivement prendre le pas sur la télévision en définition standard. Enfin, la télévision sur terminaux mobiles (téléphones de troisième génération) va également faire son apparition. Ainsi, bien que la bande passante globalement disponible augmente, il est plus que jamais nécessaire d'optimiser la bande passante utilisée par chacun de ces nouveaux services.

Le dernier standard de codage vidéo développé par le JVT (Joint Video Team) regroupant les experts MPEG et ITU, à savoir MPEG-4 Part 10 (ou encore AVC ou H.264), vise à gagner jusqu'à 50% de la bande passante actuellement utilisée par MPEG-2 pour une qualité équivalente.

Or ces nouveaux algorithmes de compression sont de plus en plus complexes et nécessitent des capacités de traitement de plus en plus lourdes. De plus, l'introduction de services de Télévision Haute Définition introduit un nouvel ordre de grandeur dans la complexité de traitement. Pour l'instant, les premiers encodeurs AVC en définition standard sont loin d'avoir atteint leur potentiel de gain par rapport à MPEG-2. La principale raison est que la maturation des algorithmes de codage, qui contrairement au décodage sont du domaine compétitif et non norminatif, ne se fait pas en un jour. Ainsi, il a fallu dix ans à MPEG-2 pour gagner un facteur deux en débit pour une qualité équivalente. Il faudra bien encore cinq ans avant que les chercheurs et les industriels parviennent à exploiter tout le potentiel de MPEG-4.

L'objectif du projet ArchiPEG est de concevoir une architecture capable de réaliser la compression MPEG-4 AVC en temps réel et en haute définition. Celle-ci devant être modulaire, programmable, évolutive et graduable. Elle devra également intégrer un outil de pré-traitement vidéo générant des informations communes et utiles à tous les algorithmes de codage. Cet outil doit pouvoir caractériser le

mouvement physique ainsi que la complexité locale de l'image dans le but de choisir le meilleur mode de codage offert par le codeur H.264/AVC

Ce rapport bibliographique est divisé en plusieurs parties. La première partie de ce rapport réalise une présentation du codeur H.264/AVC. Le chapitre suivant fait l'état de l'art des techniques d'optimisation du codeur H.264/AVC. Le troisième chapitre présente l'extension à la graduabilité du codeur H.264/AVC, à savoir les graduabilités spatiale, temporelle et en qualité (SNR). Les deux derniers chapitres présentent les différentes techniques de filtrage temporel avec compensation de mouvement et les méthodes de segmentation spatio-temporelles et de suivi d'objets dans une séquence vidéo qui seront mis à profit par notre outil de pré-traitement.

Chapitre 1

Le codeur H.264/AVC

1.1 Introduction

En 1998, VCEG[1] se lance dans un projet appelé H.26L, dont le but est de multiplier par deux l'efficacité du codage vidéo par rapport à n'importe quelle norme existante alors. En 2001, MPEG[2] rejoint VCEG et le JVT[3] est créé pour concrétiser la nouvelle norme. En 2003, la norme H.264/MPEG-4 Part 10 est publiée [ISO/IEC, 03]. L'objectif de H.264/AVC[4] est un codage efficace et robuste et le transport de la vidéo. Les applications sont diverses : la communication vidéo (vidéoconférence et vidéotéléphonie), le codage haute qualité pour la diffusion vidéo et la vidéo en temps réel sur des réseaux par paquets (Internet). Ce chapitre présente les principales caractéristiques de la norme H.264/AVC.

1.2 Structure du codeur H.264/AVC

Cette section présente la structure de la norme H.264/AVC. Avant de détailler le codec et les profils de compression, il est nécessaire de définir certains éléments et termes qui seront employés.

1.2.1 Terminologie

La norme H.264 emploie une certaine terminologie que nous allons détailler [Richardson, 03] :

Une trame (d'une vidéo entrelacée) ou une *frame* (d'une vidéo progressive) est codée afin de produire une image codée. Un numéro de frame lui est assigné (signalé dans le flux binaire), qui ne correspond pas forcément à l'ordre de décodage. Les images précédemment codées, les images de référence, sont utilisées pour la prédiction inter-images. Ces images de référence sont organisées en une ou deux listes : la liste 0 et la liste 1.

Une image codée est composée d'un certain nombre de macroblocs, chacun contenant 16×16 échantillons de luminance et deux fois 8×8 échantillons de chrominance. Les macroblocs sont arrangés en tranches (*slice* en anglais), chaque tranche contenant un nombre de macroblocs compris entre un (un macrobloc par tranche) et le nombre total de macroblocs d'une image (une tranche par image). Ce nombre doit être constant dans une image. Une tranche I ne contient que des macroblocs I, une

[1] *Video Coding Experts Group*
[2] *Moving Picture Expert Group*
[3] *Joint Video Team*
[4] *Advanced Video Coding*

tranche P peut contenir des macroblocs I et P et une tranche B peut contenir des macroblocs B et I. Il existe également des tranches SI et SP.

Les macroblocs I sont obtenus par prédiction intra à partir d'échantillons décodés dans la tranche courante. Une prédiction est formée soit pour un macrobloc complet, soit pour chaque bloc 4×4 de luminance (et les blocs associés de chrominance) du macrobloc.

Les macroblocs P sont obtenus par prédiction inter à partir des images de référence. Un macrobloc codé inter peut être divisé en partitions de macroblocs : des blocs de taille 16×16, 16×8, 8×16 ou 8×8 de luminance (et les blocs de chrominance associés). En choisissant la partition 8×8, chaque sous-macrobloc peut alors être de la même manière divisé en partitions de sous-macroblocs, de taille 8×8, 8×4, 4×8 ou 4×4. La prédiction se fait depuis une image de référence.

Les macroblocs B sont obtenus par prédiction inter à partir de plusieurs images de référence. Chaque partition de macrobloc peut utiliser une ou deux images de référence, une image de la liste 0 et/ou une image de la liste 1.

Un GOP[5] (ou groupe d'images) est un ensemble d'images constituant une séquence. Il commence par une image I, suivie d'images P et/ou B. Le prochain GOP commence à l'image I suivante.

Ces quelques définitions nous permettent maintenant de décrire le codec H.264/AVC.

1.2.2 Le codec H.264

Tout comme les autres standards de codage vidéo, H.264/AVC [Wiegand *et al.*, 03] ne définit pas un codec mais la syntaxe d'un flux binaire codé de vidéo ainsi que la procédure de décodage de ce flux. En pratique, un codeur/décodeur conforme à la norme inclut les fonctionnalités décrites sur les figures 1.1 et 1.2.

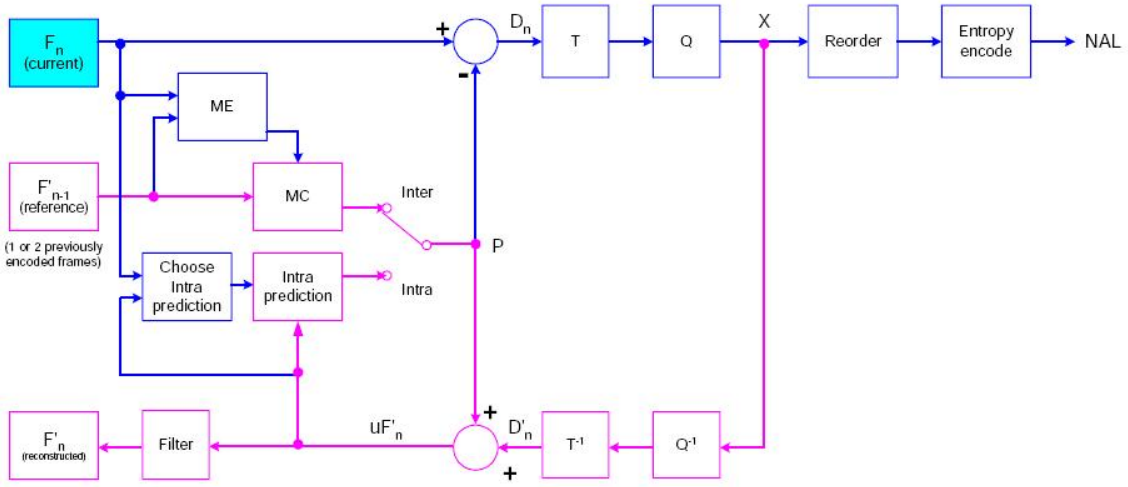


FIG. 1.1 – Schéma du codeur H.264/AVC [Richardson, 03].

[5] *Group of Pictures*

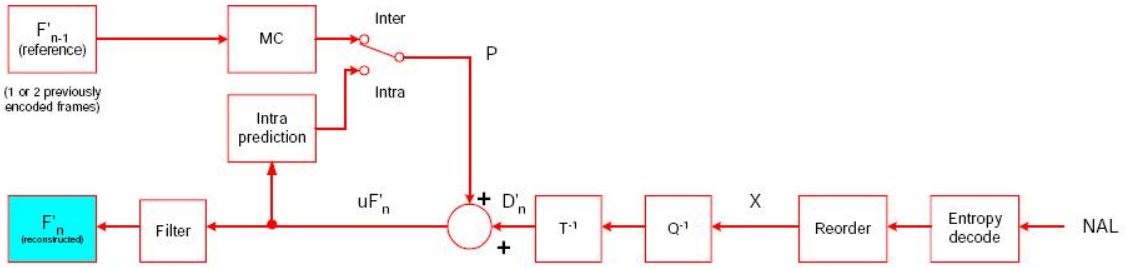


FIG. 1.2 – Schéma du décodeur H.264/AVC [Richardson, 03].

Le codeur (figure 1.1) inclut deux chemins pour le flux de données, le chemin « avant » (de gauche à droite) et le chemin de reconstruction (de droite à gauche). Sur la figure 1.2, le chemin de données du décodeur est présenté de droite à gauche, indiquant ainsi les similarités avec le codeur.

D'après la figure 1.1, sur le chemin avant, une trame ou une image F_n est partitionnée en macroblocs, une partition p est formée en fonction des échantillons reconstruits. Dans le mode intra, p est obtenue à partir d'échantillons de la tranche courante ayant déjà été codés, décodés et reconstruits (uF'_n sur la figure ; notons que les échantillons utilisés ne sont pas filtrés). En mode inter, p est obtenue par prédiction par compensation de mouvement à partir d'une ou deux images de référence sélectionnées dans la liste 0 et/ou la liste 1. Sur les figures précédentes, l'image de référence est l'image précédemment codée F'_{n-1}, mais cette référence peut être choisie parmi des images passées ou futures, ayant été codées, décodées et reconstruites.

La prédiction p est ensuite soustraite au bloc courant, générant un bloc résiduel D_n. Celui-ci est alors transformé (T) et quantifié (Q) afin de produire X, un ensemble de coefficients de transformation quantifiés. Ils sont réarrangés puis transmis au codeur entropique. Les coefficients obtenus et les informations nécessaires au décodage (mode de prédiction, table de quantification, vecteurs de mouvement, ...) sont codés en un flux binaire comprimé qui est passé au NAL[6] pour transmission ou codage.

En plus du codage et de la transmission, le codeur exécute également un décodage suivant le chemin de reconstruction. Cette étape est nécessaire pour fournir une référence pour les futures prédictions. Les coefficients X subissent un ré-échelonnement (Q^{-1}) et une transformée inverse (T^{-1}) pour produire un bloc de différence D'_n. La prédiction p est ajoutée à D'_n pour créer le bloc reconstruit uF'_n, c'est-à-dire, la version décodée du bloc original. Enfin un filtre est appliqué afin de réduire les effets de bloc. L'image de référence reconstruite est ainsi créée par une série de blocs F'_n.

Au niveau du décodage, le même processus est appliqué. Les macroblocs reçus par le NAL sont décodés, réarrangés pour obtenir les coefficients X. Ré-échelonnement et transformée inverse sont appliqués pour obtenir D'_n (identique au D'_n du schéma d'encodage). Grâce aux informations d'entête

[6] *Network Abstraction Layer*

du flux binaire, le décodeur crée la même prédiction p que celle du codeur, ajoute D'_n pour produire uF'_n. Celle-ci est alors filtrée pour générer chaque bloc décodé F'_n.

Cette description du codage/décodage H.264/AVC se veut générale. En effet, plusieurs possibilités existent dans l'utilisation de ces fonctions, ce sont les profils.

1.2.3 Les profils et niveaux

Le standard H.264/AVC propose trois profils et onze niveaux afin de définir les points de conformité. Les premiers indiquent un ensemble d'outils de codage pouvant être utilisés pour générer un flux compatible. Les seconds imposent des contraintes à certains paramètres clés du flux (par exemple la taille maximale d'une image). Le profil de base (*baseline profile*) supporte le codage intra et inter des tranches I et P et le codage entropique CAVLC[7]. Le profil principal (*main profile*) ajoute le support pour la vidéo entrelacée, le codage inter des tranches B, la prédiction pondérée et le codage entropique CABAC[8]. Enfin, le profil étendu (*extended profile*) ne supporte ni la vidéo entrelacée ni le codage CABAC mais ajoute des modes pour permettre une commutation efficace entre les flux binaires codés (avec tranches SP et SI) et améliore la résistance du codage face aux erreurs (*data partitionning*). La figure 1.3 résume les fonctionnalités incluses dans chaque profil.

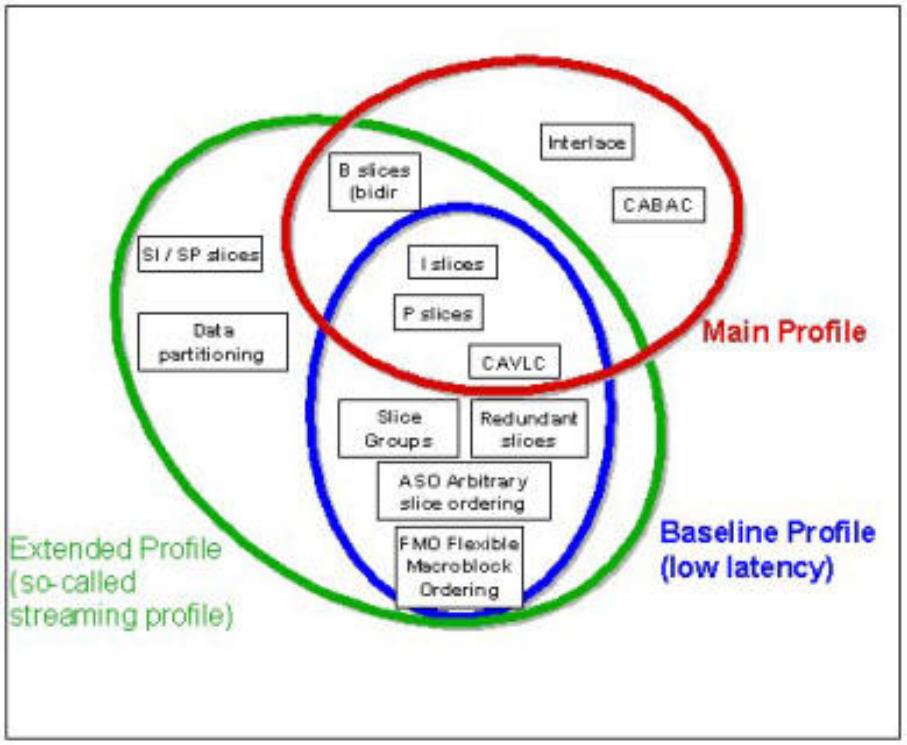


FIG. 1.3 – Les profils dans H.264/AVC.

[7] *Context-Adaptive Variable-Lenght Codes*
[8] *Context-Based Arithmetic Coding*

Ces différents profils permettent différentes applications. Ainsi le profil de base est utilisé pour la vidéotéléphonie, la vidéoconférence et la communication sans fil. Le profil principal peut être utilisé pour la télévision et le stockage tandis que le profil étendu est destiné à la diffusion de vidéos en continu, au fur et à mesure du téléchargement. Néanmoins, chaque profil est suffisamment flexible pour supporter une large gamme d'applications.

Les différents profils permettent une large plage d'utilisation de la norme H.264/AVC. Nos travaux se feront avec le profil principal et la suite de ce chapitre se limitera donc aux fonctionnalités offertes par ce profil (ce qui inclut le profil de base).

1.3 Prédiction inter-images

H.264/AVC permet d'utiliser une ou deux images de référence pour la prédiction inter-images. La prédiction peut se faire à partir du passé, du futur ou bi-directionnelle. Pour cela, le codeur et le décodeur maintiennent une ou deux listes d'images de référence [Wiegand *et al.*, 99], selon qu'il s'agisse d'une tranche P ou d'une tranche B. Les techniques mises en œuvre dans la prédiction inter-images sont présentées par la suite.

1.3.1 Gestion des images de référence

Les images précédemment codées sont décodées et stockées dans une mémoire tampon appelée DPB[9], dont la taille est définie par un paramètre et l'indexage commence à 0. Le codeur et le décodeur conservent une ou deux listes d'images précédemment codées/décodées, la liste 0 et la liste 1 d'images de référence. Pour les tranches P, la liste 0 contient des images avant et après l'image courante selon l'ordre d'affichage. Ces images peuvent être des images de référence à court ou long terme (*short term* et *long term*), disponibles pour la prédiction. Les images à court terme sont directement identifiées par leur numéro d'image. Les images à long terme sont typiquement des images plus anciennes mais pouvant être utiles pour la prédiction. Elles sont identifiées par une variable *LongTermPictureNum* et restent dans le DPB jusqu'à un ordre explicite de suppression ou de remplacement.

Une image codée puis reconstruite (au niveau du codeur) ou décodée (au niveau du décodeur) est placée dans le DPB et est :

- soit marquée comme « inutile pour référence » (et ne sera pas utilisée pour les prochaines prédictions),
- soit marquée en tant qu'image à court terme (cas par défaut),
- soit marquée en tant qu'image à long terme,
- ou simplement envoyée à l'affichage.

Par défaut, les images à court terme de la liste 0 sont ordonnées du plus grand au plus petit *PicNum* (une variable dérivée du numéro d'image) et les images à long terme sont ordonnées depuis le plus petit

[9] *Decoded Picture Buffer*

vers le plus grand *LongTermPicNUm* . Un changement de cet ordre peut être signalé par le codeur. Pour chaque nouvelle image à court terme ajoutée à l'index 0, l'indice des autres images à court terme est incrémenté. Si le nombre d'images (court et long terme) est égal à la taille du DPB, l'image à court terme la plus ancienne (avec l'index le plus grand) est supprimée : le DPB agit comme un contrôleur de mémoire à fenêtre glissante.

Des commandes envoyées par le codeur permettent la gestion des index des images du DPB. Une image à court terme peut devenir une image à long terme ou des images à court et long terme peuvent être marquées « inutile référence ».

Dans le cas des tranches B, le codeur utilise deux listes d'images de référence : la liste 0 et la liste 1, contenant des images à court et long terme, passées ou futures selon l'ordre d'affichage. Les images à long terme se comportent de la même manière que celles à court terme. L'ordonnancement des images à court terme est le suivant :

Liste 0 : l'image passée la plus proche (selon l'ordre du compteur d'images ou POC[10]) est assignée à l'index 0, puis les autres images passées (en diminuant le POC) et les images futures (en augmentant le POC).

Liste 1 : l'image future la plus proche est assignée à l'index 0, puis les autres images futures (POC augmentant) et les images passées (POC diminuant).

Le tableau 1.1 donne un exemple de contenu d'un buffer de taille 5, l'image courante ayant un POC égal à 127.

Index	Liste 0	Liste 1
0	126	128
1	125	129
2	123	130
3	128	126
4	129	125
5	130	123

TAB. 1.1 – Indice des images à court terme d'un DPB [Richardson, 03].

Le codeur peut alors choisir une ou deux images de référence de la liste 0 et/ou de la liste 1 pour encoder chaque sous-macrobloc d'un macrobloc codé en inter. De plus, grâce à une possibilité de réarrangement de l'ordre des images dans les listes (en mettant l'image choisie à l'index 0), on réduit le coût de codage pour signaler la prédiction à partir de cette image. Dans la section suivante, nous allons décrire la prédiction dans les tranches P, puis les tranches B.

1.3.2 Les tranches P

La prédiction inter crée un modèle de prédiction à partir d'une ou plusieurs images de référence en faisant de la compensation de mouvement basée bloc. H.264/AVC inclut le support d'une large

[10] *Picture Order Count*

gamme de tailles de blocs et une meilleure précision pour les vecteurs de mouvement, à la différence des standards précédents.

Compensation de mouvement à structure d'arbre

Chaque macrobloc 16×16 peut être découpé de quatre manières différentes (voir la figure 1.4) et la compensation de mouvement peut s'appliquer sur une partition 16×16, deux partitions 16×8, deux partitions 8×16 ou quatre partitions 8×8. En choisissant le mode 8×8, les quatre sous-macroblocs peuvent encore une fois être divisés (voir la figure 1.5) en une partition 8×8, deux partitions 8×4, deux partitions 4×8 ou quatre partitions 4×4. Ces partitions et sous-macroblocs permettent un grand nombre de combinaisons pour chaque macrobloc. C'est la méthode dite de compensation de mouvement à structure d'arbre.

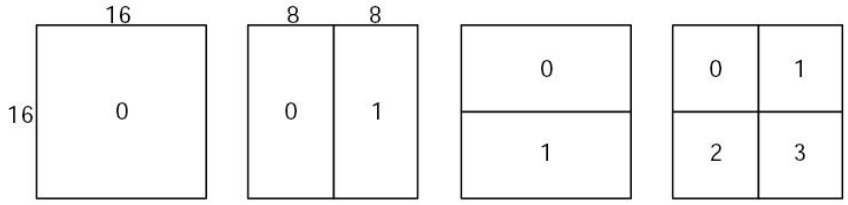


FIG. 1.4 – Partitions de macroblocs : 16×16, 8×16, 16×8 et 8×8.

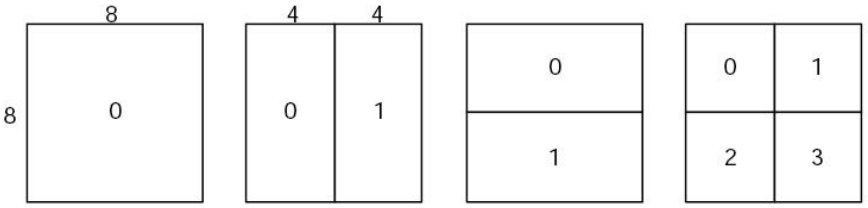


FIG. 1.5 – Partitions de sous-macroblocs : 8×8, 4×8, 8×4 et 4×4

Un vecteur de mouvement est nécessaire pour chaque partition ou sous-macrobloc. Ce vecteur doit être codé et transmis et le choix de la partition doit être codé dans le train binaire. Une grande partition (16×16, 8×16 ou 16×8) ne nécessitera pas beaucoup de bits pour coder les vecteurs de mouvement et le choix de la partition. En revanche le résidu obtenu pourra contenir une grande énergie dans les zones très riches. Une petite partition (8×8, 4×8, 8×4 ou 4×4) diminuera l'énergie résiduelle, mais nécessitera plus de bits pour coder tous les vecteurs et les choix de partition. Ainsi, le choix de la taille de partition a un impact important sur les performances de compression. En règle générale, une grande taille est appropriée pour les zones homogènes et une petite pour les zones de détails.

Chaque composante de chrominance (C_b et C_r) dans un macrobloc a une résolution moitié moindre que celle de la luminance. Chaque bloc de chrominance est partitionné selon la même méthode décrite précédemment, avec un quotient de deux horizontalement et verticalement (une partition 8×16 de

luminance correspond a une partition 4×8 de chrominance). Les composantes verticales et horizontales de chaque vecteur de mouvement sont dédoublées pour les blocs de chrominance.

Les vecteurs de mouvement

Chaque partition ou sous-partition d'un macrobloc codé inter est prédite à partir d'une zone de même taille d'une image référence. La différence de position entre les deux zones (le vecteur de mouvement) a une résolution au quart d'échantillon pour la luminance et au huitième d'échantillon pour la chrominance. Les échantillons de luminance et de chrominance n'existent pas à ces positions sous-échantillonnées dans l'image de référence. Il est nécessaire de les créer par l'interpolation des échantillons des voisins déjà codés. Cette interpolation se fait par étapes et utilise différentes formules. Pour plus d'informations sur l'interpolation réalisée, le lecteur est invité à se référer à l'article écrit par Wedi [Wedi, 03] et le livre sur le codeur H.264/AVC [Richardson, 03].

Prédiction des vecteurs de mouvement

Coder un vecteur de mouvement pour chaque partition coûterait un nombre de bits significatif, surtout pour des partitions de petites tailles. Or les vecteurs de mouvement des partitions voisines sont souvent très corrélés. Ainsi, chaque vecteur de mouvement est prédit à partir des vecteurs des partitions voisines déjà codées. Un vecteur prédit MV_p est calculé à partir des vecteurs de mouvement précédemment obtenus et la différence MVD entre le vecteur courant et le vecteur prédit est codée et transmise. La méthode de prédiction d'un MV_p dépend de la taille de la partition pour la compensation de mouvement et de la disponibilité des vecteurs voisins.

Soit E le macrobloc courant, partition de macroblocs ou de sous-macroblocs et A, B et C les partitions de macroblocs ou sous-macroblocs situées respectivement à gauche, au-dessus et à droite de E. S'il y a plusieurs partitions à gauche de E, on choisit la plus haute pour A. De même pour la partition B, c'est celle située la plus à gauche qui sera retenue. La figure 1.6 illustre ce principe.

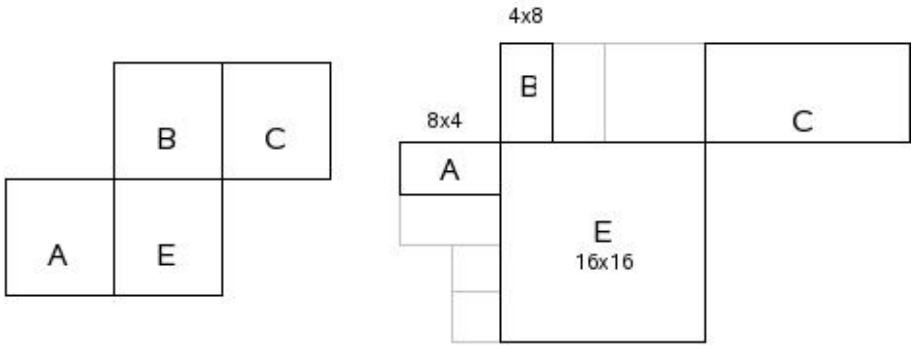


FIG. 1.6 – Partitions courantes et voisines (à gauche pour des partitions de même tailles et à droite pour des partitions de tailles différentes)

1. Pour les partitions transmises sauf 16×8 et 8×16, chaque composante de MV_p est la médiane respectivement des composantes des vecteurs de mouvement de A, B et C.
2. Pour les partitions de taille 16×8, MV_p pour la partition supérieure est prédit à partir de B et pour la partition inférieure MV_p est prédit à partir de A.
3. Pour les partitions 8×16, MV_p pour la partition gauche est prédit à partir de A et pour la partition droite à partir de C.
4. Pour les macroblocs $SKIP$ (voir ci-dessous), MV_p est généré selon le cas 1 ci-dessus (comme si le bloc était codé en mode inter 16×16).

Si un ou plusieurs blocs ne sont pas disponibles (par exemple en dehors de la tranche), le choix du MV_p est modifié en conséquence. Du côté du décodeur, le vecteur prédit MV_p est créé de la même façon et ajouté à la différence MVD. Dans le cas des macroblocs $SKIP$, il n'y a pas de MVD et le bloc compensé en mouvement est construit avec le vecteur de mouvement MV_p.

Outre ces modes de prédiction, les macroblocs peuvent également être codés en mode $SKIP$. Dans ce cas, ne sont transmis ni signal d'erreur de prédiction quantifiée, ni index de référence, ni vecteur de mouvement. Le macrobloc est reconstitué comme s'il s'agissait d'un macrobloc 16×16 codé inter référençant l'image à l'index 0 du DPB.

1.3.3 Les tranches B

Au contraire des normes précédentes, le concept des tranches B est généralisé dans H.264/AVC. La différence avec les tranches P est que les partitions de macroblocs d'une tranche B peuvent utiliser une moyenne pondérée de deux valeurs de prédiction à compensation de mouvement distinctes pour générer le signal de prédiction. On a alors quatre techniques de prédiction possibles : mode direct, prédiction à compensation de mouvement à partir de la liste 0 ou de la liste 1 ou double prédiction à compensation de mouvement. Différents modes peuvent être choisis pour chaque partition. Cependant dans le cas d'une partition 8×8, le mode de prédiction choisi sera appliqué à toutes les sous-partitions de celle-ci. Détaillons maintenant le fonctionnement des nouveaux modes : bi-prédiction et direct.

Bi-prédiction ou double prédiction

Dans ce mode, un bloc (de taille identique à la partition ou sous-partition de macrobloc courante) est créé à partir des listes 0 et 1 des images de référence. On obtient ainsi deux zones de référence, ce qui implique deux vecteurs de mouvement (voir figure 1.7). La prédiction finale est alors formée par la moyenne pondérée des blocs référencés par les deux vecteurs. Après le calcul du bloc prédit, le résidu est soustrait au macrobloc courant de manière classique.

D'autre part, chaque vecteur de la liste 0 et de la liste 1 de ce mode est prédit à partir des vecteurs de mouvement voisins ayant la même direction temporelle. Par exemple un vecteur du bloc courant

pointant sur une image passée est prédit à partir d'autres vecteurs voisins pointant également vers des images passées.

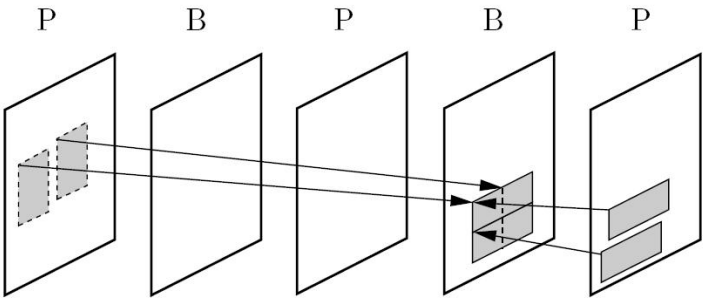


FIG. 1.7 – Prédiction bi-directionnelle de la tranche B [Flierl *et al.*, 03].

Prédiction directe

Dans ce mode, aucun vecteur de mouvement n'est transmis pour coder un bloc. À la place, le décodeur calcule les vecteurs pour la liste 0 et la liste 1 selon les vecteurs précédemment codés et utilise ces listes pour faire une bi-prédiction des échantillons résiduels décodés.

Un témoin dans l'entête de la tranche indique la méthode de calcul des vecteurs pour la prédiction directe. Deux méthodes existent :

- Le mode spatial direct : les listes prédites 0 et 1 de vecteurs sont calculées identiquement à la prédiction des vecteurs de mouvement décrite pour les tranches P. Si le macrobloc ou la partition, co-localisé dans la première image de référence de la liste 1, a un vecteur de mouvement d'ampleur inférieure à un demi échantillon de luminance (et pour d'autres cas), un ou les deux vecteurs sont fixés à zéro ; sinon les vecteurs des listes prédites sont utilisés pour la bi-prédiction.
- Le mode temporel direct : la méthode appliquée est la suivante :

  1. Trouver l'image de référence de la liste 0 pour le macrobloc ou partition co-localisé dans l'image de la liste 1.
  2. Trouver le vecteur de la liste 0, MV_c, pour le macrobloc ou partition co-localisé dans l'image de la liste 1.
  3. Interpoler le vecteur MV_c en fonction de la « distance » entre l'image courante et celle de liste 1 : c'est le nouveau vecteur MV_1 de la liste 1.
  4. Interpoler le vecteur MV_c en fonction de la « distance » entre l'image courante et celle de la liste 0 : c'est le nouveau vecteur MV_0 de la liste 0.

La figure 1.8 illustre ce processus. MV_0 et MV_1 sont calculés en fonction des distances temporelles TD_B et TD_D.

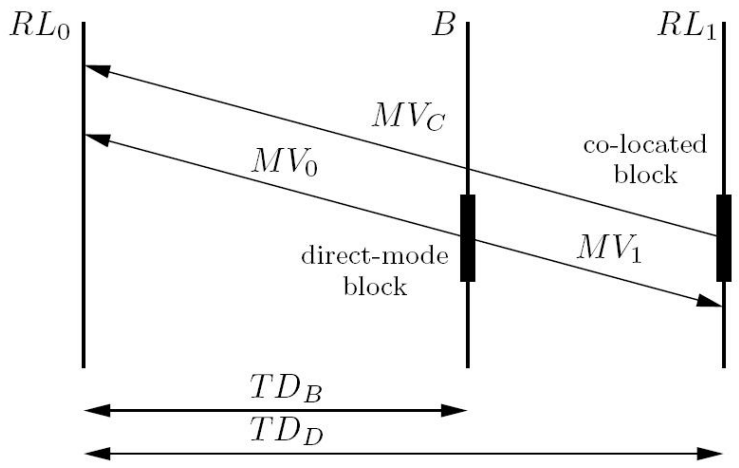


FIG. 1.8 – Dans le mode temporel direct, un bloc a deux vecteurs de mouvement dérivés MV_0 et MV_1 pointant vers deux images de référence RL_0 et RL_1. [Flierl *et al.*, 03].

Ces modes peuvent cependant être modifiés, par exemple si les macroblocs de référence ne sont pas disponibles ou s'ils sont codés en intra.

Nous venons de détailler les techniques de prédiction de mouvement à compensation de mouvement dans les tranches P et B. Il existe également un processus de prédiction pondérée. Celle-ci permet de diminuer ou d'augmenter « l'importance » d'une prédiction par un facteur déterminé par le codeur. Un facteur plus grand sera appliqué si l'image de référence est temporellement proche de l'image courante et un plus petit facteur sera appliqué sur une image de référence éloignée temporellement. Ce processus est intéressant par exemple dans le codage d'une scène de transition avec des effets de fondu. Nous présentons maintenant la prédiction intra-image qui permet de coder les images I.

1.4 Prédiction intra-image

À l'inverse de la prédiction temporelle, la prédiction intra n'utilise pas d'images de référence. Elle exploite la redondance spatiale d'une image, c'est-à-dire, le fait que les pixels dans une image soient très corrélés. Le codeur H.264/AVC a la particularité de travailler dans le domaine spatial, se référant aux échantillons voisins de blocs déjà codés et non dans le domaine transformé comme c'est le cas pour les codeurs MPEG-2 et MPEG-4. La prédiction dépend bien sûr de la taille de l'échantillon considéré. Il existe donc plusieurs méthodes en fonction du bloc : H.264/AVC permet la prédiction intra de blocs 4×4 ou 16×16 de luminance et de blocs 8×8 de chrominance. Le codeur sélectionne le mode pour chaque bloc qui minimise la différence entre le bloc prédit P et le bloc courant.

1.4.1 Prédiction des blocs 4×4 de luminance

La prédiction sur des blocs 4×4 de luminance est une spécificité de H.264/AVC par rapport aux autres standards qui utilisent des blocs 8×8. Cela apporte une meilleure précision dans la prédiction mais également plus de calculs. Neuf modes de prédiction sont possibles pour les blocs 4×4 selon la direction de prédiction. La figure 1.9 présente la façon de labelliser les pixels. Sur la figure 1.10, qui présente les neuf modes, les flèches indiquent la direction de prédiction. Pour les modes 3 à 8, les pixels prédits sont formés par une moyenne pondérée des pixels [A-M]. Par exemple pour le mode 4 (cf figure 1.10), le pixel d sur la figure 1.9 est prédit par l'arrondi de $(B/4 + C/2 + D/4)$. Le tableau 1.2 précise chaque mode [Richardson, 03].

M	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**
I	a	b	c	d				
J	e	f	g	h				
K	i	j	k	l				
L	m	n	o	p				

FIG. 1.9 – Labellisation des échantillons de prédiction 4×4.

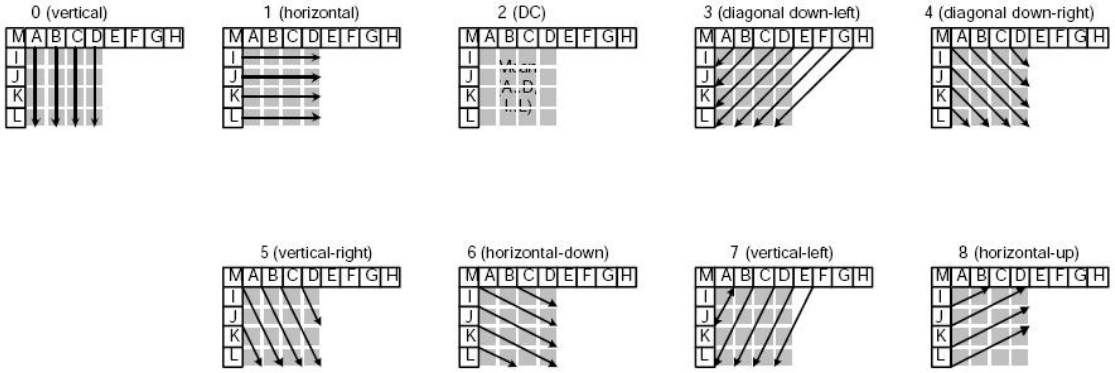


FIG. 1.10 – Les modes de prédiction des blocs 4×4 de luminance [Richardson, 03].

1.4.2 Prédiction des blocs 16×16 de luminance

La prédiction des blocs 4×4 est plus adaptée pour les zones de détails, mais nécessite plus de calculs. Le codeur H.264/AVC permet également la prédiction de macroblocs entiers de luminance (16×16). Quatre modes sont alors possibles, comme le montre la figure 1.11. Le tableau 1.3 précise les calculs réalisés.

1.4.3 Prédiction des blocs de chrominance

Chaque bloc 8×8 de chrominance est prédit à partir des échantillons de chrominance déjà codés au-dessus et à gauche. Les deux composantes C_b et C_r utilisent toujours la même prédiction. Quatre

Mode	Description
Mode 0 (vertical)	Les échantillons supérieurs A, B, C et D sont extrapolés verticalement.
Mode 1 (horizontal)	Les échantillons gauches I, J, K et L sont extrapolés horizontalement.
Mode 2 (DC)	Tous les échantillons [a-p] sont prédits par la moyenne de [A-D] et [I-L].
Mode 3 (diagonale bas-gauche)	Les échantillons sont interpolés avec un angle de 45 ° entre le bas-gauche et le haut-droit.
Mode 4 (diagonale bas-droite)	Les échantillons sont interpolés avec un angle de 45 ° vers le bas et vers la droite.
Mode 5 (vertical-droit)	Extrapolation avec un angle de 26,6 ° à gauche de la verticale ($largeur/hauteur = 1/2$).
Mode 6 (horizontal-bas)	Extrapolation avec un angle de 26,6 ° en-dessous de l'horizontale.
Mode 7 (vertical-gauche)	Extrapolation (ou interpolation) avec un angle de 26,6 ° à droite de la verticale.
Mode 8 (horizontal-haut)	Interpolation avec un angle de 26,6 ° au-dessus de l'horizontale.

TAB. 1.2 – Les modes de prédiction des blocs 4×4 de luminance [Richardson, 03].

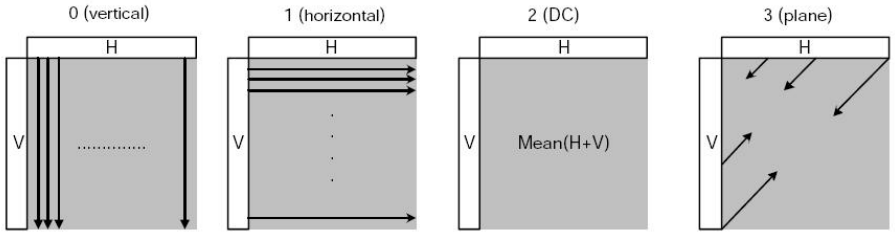


FIG. 1.11 – Les modes de prédiction des blocs 16×16 de luminance [Richardson, 03].

modes de prédiction existent, similaires aux modes 16×16 de luminance (voir la figure 1.11). Le tableau 1.4 présente ces différents modes.

1.4.4 Codage des modes de prédiction

Le choix du mode de prédiction doit être signalé au décodeur. Pour les blocs 4×4, cela peut demander un grand nombre de bits. Cependant, au voisinage de blocs 4×4, les modes sont souvent corrélés. Par exemple, soit A, B et E les blocs 4×4 gauche, haut et courant comme indiqué sur la figure 1.6. Si les blocs A et B on été prédits avec le mode 1, il est probable que le meilleur mode pour le bloc courant soit le mode 1 également. Ainsi pour minimiser le codage, il existe un système de prédiction du codage des modes de prédiction des blocs 4×4.

Pour chaque bloc E, le codeur et le décodeur calculent le mode de prédiction le plus probable, le minimum des modes de prédiction de A et B. Si l'un de ces blocs voisins n'est pas disponible, la valeur

Mode	**Description**
Mode 0 (vertical)	Extrapolation à partir des échantillons supérieurs (H).
Mode 1 (horizontal)	Extrapolation à partir des échantillons gauches (V).
Mode 2 (DC)	Moyenne de H et V.
Mode 4 (plan)	Une fonction linéaire plane est ajustée à H et V. Cela fonctionne bien sur les zones de luminance variant doucement.

TAB. 1.3 – Les modes de prédiction des blocs 16×16 de luminance [Richardson, 03].

Mode	**Description**
Mode 0 (DC)	Moyenne de H et V.
Mode 1 (horizontal)	Extrapolation à partir des échantillons gauches (V).
Mode 2 (vertical)	Extrapolation à partir des échantillons supérieurs (H).
Mode 3 (plan)	Une fonction linéaire plane est ajustée à H et V. Cela fonctionne bien sur les zones de luminance variant doucement.

TAB. 1.4 – Les modes de prédiction des blocs 8x8 de chrominance.

du bloc indisponible est assigné à 2 (mode DC).

Le codeur utilise un témoin, nommé $prev\_intra4x4\_pred\_mode$ pour chaque bloc 4×4. Si ce témoin est à 1, alors le mode de prédiction le plus probable est utilisé. Autrement un autre paramètre ($rem\_intra4x4\_pred\_mode$) est envoyé pour indiquer un changement. Si $rem\_intra4x4\_pred\_mode$ est plus petit que le mode le plus probable courant alors le mode de prédiction est positionné à $rem\_intra4x4\_pred\_mode$. Sinon le mode de prédiction est assigné à ($rem\_intra4x4\_pred\_mode+1$). De cette manière, seulement huit valeurs (de 0 à 7) de $rem\_intra4x4\_pred\_mode$ sont nécessaires pour indiquer le mode intra courant (0 à 8).

Dans le cas de la chrominance ou de blocs 16×16 de luminance, le mode de prédiction intra est indiqué dans l'entête du macrobloc, le codage du mode de prédiction n'est donc pas utilisé.

Nous venons de décrire les processus de codage intra des images, utilisant plusieurs modes et une prédiction du codage du mode. Une étape supplémentaire, introduite dans le standard, est appliquée à tous les macroblocs avant reconstruction : il s'agit d'un filtrage anti-bloc [List *et al.*, 03]. Celui-ci permet d'améliorer l'apparence des images décodées mais également de réduire l'énergie résiduelle, car la prédiction inter se fait sur les blocs filtrés.

La partie prédiction de H.264/AVC est maintenant complète, les prochaines étapes sont la transformation et la quantification.

1.5 Transformée et quantification

Après prédiction et soustraction de la prédiction au bloc original, le résidu est transformé puis quantifié, comme c'est le cas dans les autres standards. Cependant H.264/AVC apporte différentes

nouveautés.

1.5.1 Transformation

Le codeur H.264/AVC utilise trois transformées [Malvar *et al.*, 03] selon le type de données résiduelles à coder : une transformée pour les matrices 4×4 des coefficients des composantes continues de luminance des macroblocs intra (prédits en mode 16×16), une transformée sur les matrices des coefficients des composantes continues de chrominance (pour tous les macroblocs) et une transformée basée sur la TCD[11] (DCT[12] en anglais) pour tous les autres blocs 4×4. Cette dernière est une transformée en entiers séparables. La transformée inverse est réalisée par des opérations exactes sur des entiers. Ceci évite les discordances de transformée inverse. Pour plus de précision, le lecteur pourra se référer au livre écrit par Richardson [Richardson, 03].

1.5.2 Quantification

Après transformation des données résiduelles en coefficients, une quantification est appliquée. Cette opération entraîne des pertes mais permet une compression des données importantes. Le codeur H.264/AVC utilise une quantification scalaire, un calcul d'arrondi d'un nombre fractionnel en entier.

Pour chaque macrobloc, le paramètre de quantification (QP) sélectionne un des 52 quantificateurs. Les quantificateurs sont disposés de manière à obtenir une augmentation approximative de $12,5\%$ de la taille du pas de quantification lorsque QP augmente de 1. Comme pour la transformation, le lecteur pourra consulter l'œuvre de Richardson [Richardson, 03].

Les coefficients quantifiés d'un bloc sont en général analysés en zigzag et transmis pour la prochaine étape, le codage entropique.

1.6 Codage entropique

Les données arrivant à l'unité de codage entropique sont nombreuses et variées. Ce sont essentiellement les données résiduelles issues de la quantification mais également des entêtes, des vecteurs de mouvement, les méthodes de prédiction, les index des images de référence, ... Après une première étape de réarrangement des données dans un tableau, le codeur H.264/AVC applique plusieurs méthodes de codage entropique, pour les éléments syntaxiques et pour les coefficients de transformée quantifiés.

1.6.1 Codage Exp-Golomb

Ce codage à longueur variable [Golomb, 66] utilise un unique ensemble illimité de mots-codes défini pour tous les éléments syntaxiques (sauf pour les données résiduelles quantifiées). Les différentes tables

[11] Transformée en Cosinus Discret
[12] *Discrete Cosinus Transform*

de code à longueur variable (VLC) sont remplacées par une seule table, personnalisée en fonction des statistiques des données. Cette table est un code de Golomb exponentiel aux propriétés de décodage simples et régulières.

1.6.2 CAVLC

Le codage entropique CAVLC[13] [Bjøntegaard, 02] est utilisé pour le codage des coefficients de transformée quantifiés. Ce codage entropique parcourt les blocs 4×4 en zigzag et tire avantage des blocs 4×4 quantifiés :

1. Après prédiction, transformation et quantification, les blocs contiennent principalement des zéros. CAVLC les représente de manière compacte.
2. Les plus grands coefficients non nuls après le parcours en zigzag sont souvent des séquences de ± 1. CAVLC signale le nombre de ces coefficients de manière compacte.
3. Le nombre de coefficients non nuls dans les blocs voisins est corrélé. Le nombre de coefficients est codé avec une table de correspondance.
4. L'amplitude des coefficients non nuls tend à être plus grande au début du tableau réarrangé (près des coefficients de la composante continue). CAVLC en tire avantage en adaptant le choix des tables de correspondances (LUT[14]) en fonction des amplitudes déjà codées.

1.6.3 CABAC

La méthode CABAC[15] [Marpe *et al.*, 01, Marpe *et al.*, 03] disponible dans le profil principal, améliore encore le codage entropique. Le processus de codage applique les étapes suivantes :

1. Binarisation : CABAC utilise un codage arithmétique binaire, c'est-à-dire, ne codant que les unités binaires (0 ou 1). Les symboles non binaires sont « binarisés » ou convertis en code binaire avant le codage arithmétique. Chaque bit du symbole converti est appelé case (ou *bin* en anglais). Les étapes sont répétées pour chaque case ou bit du symbole binarisé.
2. Sélection du modèle de contexte. Un « modèle de contexte » enregistre la probabilité pour chaque case du symbole converti d'être un '1'ou un '0'.
3. Codage arithmétique : code chaque case en fonction du choix du modèle de probabilité.
4. Mise à jour de la probabilité : le modèle de contexte sélectionné est mis à jour par rapport à la valeur courante codée (si la case était un '1' alors le compteur de '1' est incrémenté).

Par rapport au CAVLC, CABAC garantit en général une réduction du débit binaire de 10 à 15% lors du codage de signaux télévisuels pour une même qualité.

[13] *Context-based Adaptive Variable Length Coding*
[14] *Look Up Table*
[15] *Context-Adaptive Binary Arithmetic Coding*

Le codage entropique, dernier élément du codeur H.264/AVC avant la génération du flux binaire est un élément important dans le codeur. La compression peut être efficace et permet une dernière optimisation du codage des données répétitives.

1.7 Conclusion

Ce chapitre traite du nouveau standard H.264/AVC. Nous avons détaillé, à partir d'une terminologie en premier lieu définie, les différentes étapes de compression. On constate que H.264/AVC permet essentiellement plus de précision et une meilleure prédiction dans les différentes unités de compression. L'ajout d'un filtre anti-bloc au sein du codec est une nouveauté. Toutes ces techniques de prédiction permettent d'améliorer la compression et de faire de H.264/AVC le meilleur codec (50% de débit en moins par rapport à MPEG-2 pour une même qualité [Wiegand *et al.*, 03]). Mais ce gain obtenu en terme de débit est réalisé au détriment des temps de calcul, qui même avec l'augmentation des performances des machines, deviennent très importants. Le prochain chapitre décrit les méthodes présentes dans la littérature qui permettent d'optimiser le codeur H.264/AVC et de réduire ainsi les temps de calcul.

Chapitre 2

Optimisation du codeur H.264/AVC

2.1 Introduction

Comme nous l'avons vu précédemment, au sein du codeur H.264/AVC, de nombreuses techniques de prédiction sont utilisées afin de réduire le flux de données tout en conservant la qualité des séquences vidéo. Ainsi, le codeur H.264/AVC permettrait de réduire le flux de données jusqu'à 50% par rapport au codeur MPEG-2 pour une qualité des images équivalente.

Mais cela a pour conséquence d'augmenter les temps de calcul. Par exemple, le logiciel de référence adopte une méthode de recherche *Full Search* pour la prédiction [Richardson, 03]. Il existe 7 modes inter différents (16×16, 16×8, 8×16, 8×8, 8×4, 4×8 et 4×4) et deux modes intra (16×16 et 4×4). Pour les modes inter, la recherche des vecteurs de mouvement s'effectue à l'aide de 5 images références. Ces différents modes de prédiction sont illustrés à la figure 2.1.

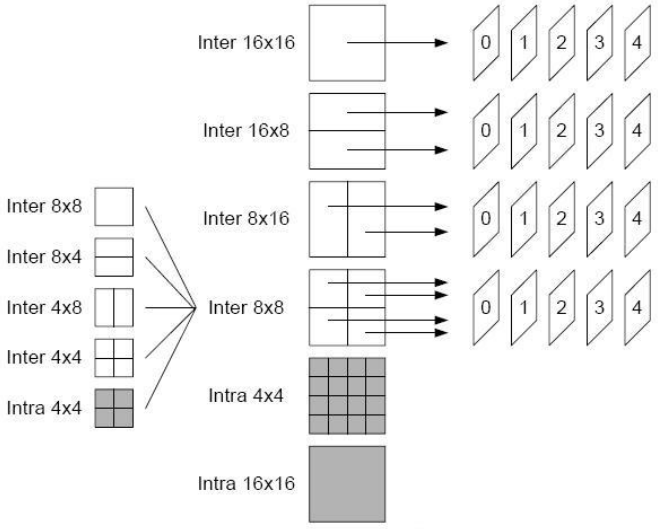


FIG. 2.1 – Étapes de recherche pour les prédictions intra et inter avec 5 images de référence pour le codeur H.264/AVC.

Ainsi, l'estimation de mouvements peut représenter 60% (1 seule image référence) à 80% (5 images références) du temps de calcul et cela peut augmenter si on utilise une fenêtre de recherche plus importante (48 ou 64). Afin de réduire ces temps de calcul, de nombreux travaux ont été proposés. Les

méthodes varient et sont axées sur trois stratégies bien distinctes :

- algorithmes d'estimation de mouvement plus rapide,
- réduction des modes de prédiction intra et inter,
- choix des images références.

Ces trois stratégies sont détaillées par la suite en décrivant les différentes techniques proposées dans la littérature.

2.2 Méthodes d'estimation de mouvement améliorées

De nombreux algorithmes ont été proposés afin d'accélérer l'estimation du mouvement : recherche à trois étapes [Koga *et al.*, 81] (TSS[1]), estimateur logarithmique [Jain, 81] (2-D LOGS[2]), recherche par descente de gradient [Liu, 96] BSDS[3]), recherche à quatre étapes [Po, 96] (FSS[4]), recherche sur une grille hexagonale [Zhu *et al.*, 02] (HEXBS[5]), etc. Ils obtiennent de bons résultats dans des petits intervalles de recherche et pour des images de taille faible. Mais pour des signaux SDTV[6] ou HDTV[7], la taille de l'image est grande et la fenêtre de recherche doit être suffisamment large pour obtenir des résultats satisfaisants. Pour des séquences telles que *Stefan* ou *Bus*, les algorithmes tels que HEXBS peuvent rencontrer un minimum local rapidement. Afin de résoudre ce problème de minimum local, les vecteurs de mouvement peuvent être prédits à partir de vecteurs des blocs voisins. Mais le vecteur de mouvement ainsi obtenu peut être « faux ». Les expériences montrent [Chen *et al.*, 02] que de tels algorithmes entraînent une perte de 1 à $2dB$ pour la qualité de l'image par rapport à la méthode de recherche complète (*Full Search*) sur les séquences *Stefan* et *Bus*.

2.2.1 Recherche multi-résolutions

Les méthodes d'estimation de mouvement multi-résolutions débutent la recherche à la plus faible résolution spatiale. Le vecteur ainsi obtenu est multiplié par un facteur proportionnel à la résolution supérieure pour devenir le point de recherche initial du prochain niveau. Cependant, les transformations ondelettes ou d'entiers nécessaires à une telle estimation de mouvements multi-résolutions ajoutent des calculs supplémentaires à l'algorithme. Choi et ses collaborateurs[Choi *et al.*, 05] proposent alors une transformation d'entiers rapide qui permet d'obtenir des coefficients d'ondelettes en réalisant une transformation d'entiers en deux étapes (cf figure 2.2).

La première étape de la transformation d'entiers décompose l'image entière en trois couches. Les coefficients obtenus dans ces trois couches sont utilisés dans la méthode d'estimation de mouvement

[1] *three step search*
[2] *2-D logarithmic search*
[3] *Block based gradient descent search*
[4] *Four step search*
[5] *Hexagon-based Search*
[6] *Standard Definition Television*
[7] *High Definition Television*

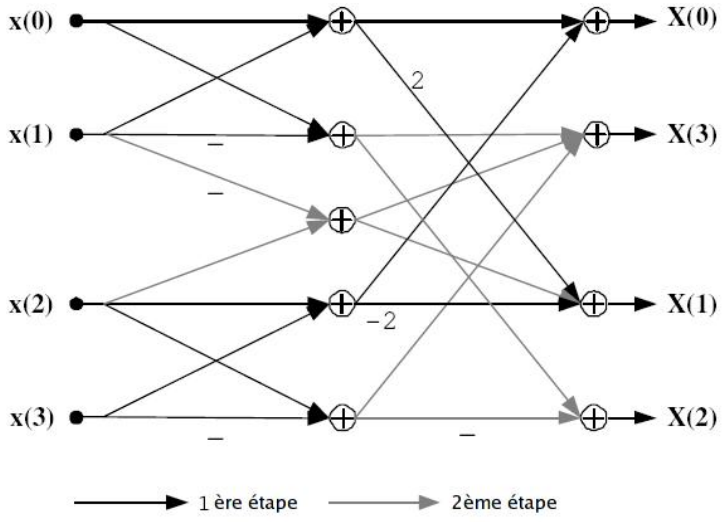


FIG. 2.2 – Modifications réalisées sur la transformation d'entiers.

proposée. Ensuite, ils réalisent l'estimation de mouvement multi-résolutions en utilisant plusieurs formes de recherches suivant les niveaux de décomposition, cela est illustré à la figure 2.3.

Les recherches multi-résolutions basées sur une décomposition à l'aide d'une transformation ondelettes dyadique possèdent des limites. En effet, le vecteur de mouvement est obtenu avec une précision de deux pixels, il est nécessaire de réaliser une recherche supplémentaire pour obtenir une précision plus fine. Le sous-échantillonnage provoque une distorsion du signal car les fréquences se recouvrent après une telle opération. L'un des moyens pour remédier à ce problème est classiquement de réaliser un filtrage passe-bas [Liu *et al.*, 06]. Dans leur algorithme, l'image de référence et l'image courante subissent d'abord un filtrage passe-bas. Ensuite, le sous-échantillonnage est réalisé sur les versions filtrées de ces images. Les hautes fréquences étant éliminées, le problème de recouvrement des fréquences est réduit.

Afin de ne pas trop augmenter la complexité de l'algorithme, ils utilisent un filtre passe-bas de Haar. Ainsi chaque pixel de l'image résultante est la somme du voisinage sur quatre pixels de l'image originale. Ensuite, les versions filtrées de l'image de référence et de l'image courante sont sous-échantillonnées dans les deux directions (verticale et horizontale). Ainsi, la recherche pour chaque point est réduit à seulement 25% d'un algorithme de recherche complète (*Full Search*). Ils proposent également de prédire le vecteur de mouvement à partir des vecteurs des quatre macroblocs 4×4 situés au dessus du macrobloc courant. Finalement, ils proposent une méthode adaptative pour dimensionner la taille de la fenêtre de recherche en fonction du vecteur de mouvement prédit. Leur algorithme obtient des performances similaires (en termes de débit et qualité) à la méthode classique de recherche complète (*Full Search*), tout en ayant une complexité beaucoup plus faible.

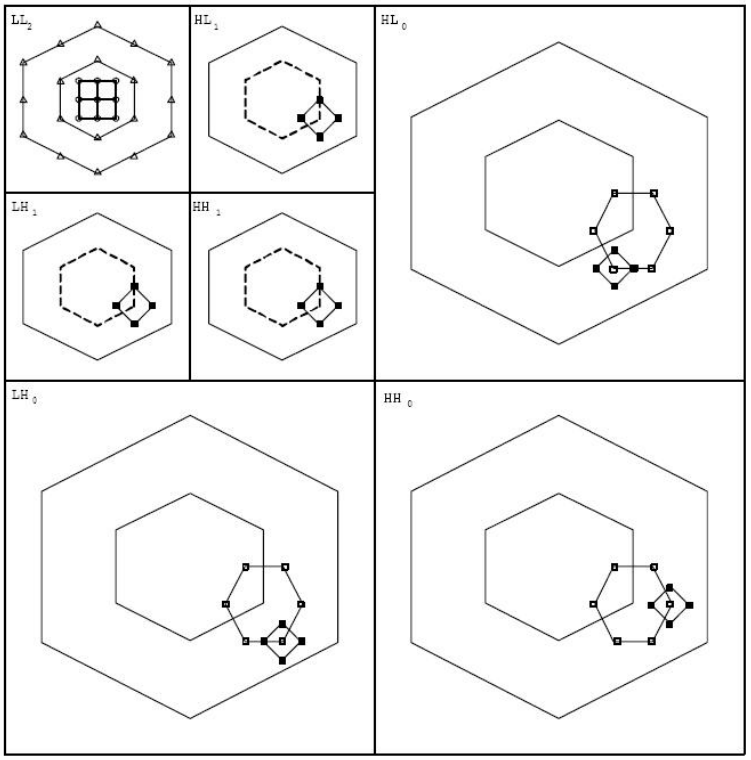


FIG. 2.3 – Formes de recherche pour une estimation du mouvement multi-résolutions.

2.2.2 Estimation du mouvement avec des recherches partielles

La plupart des méthodes présentes dans la littérature concernant le codeur H.264/AVC essaient d'accélérer le processus d'estimation de mouvement. Certaines ont même été adoptées par le JVT [Chen *et al.*, 02, Chen *et al.*, 03b, Chen *et al.*, 03a, Xu *et al.*, 03]. L'algorithme proposé par Chen et ses collaborateurs [Chen *et al.*, 02] et intitulé UMHexagonS (*Hybrid Unsymmetrical-cross Multi-Hexagon-grid Search*), utilise des stratégies hybrides et hiérarchiques de recherche de mouvement. Il est appelé hybride car il est constitué de quatre étapes avec différentes formes de recherches.

1. sélection du prédicteur et changements des modes de prédiction,
2. recherche sur une croix asymétrique (*Unsymmetrical-cross search*),
3. recherche sur de multiples grilles hexagonales de tailles variables (*Uneven multi-hexagon-grid search*),
4. recherche étendue sur une forme hexagonale (*Extended hexagon based search*).

La figure 2.4 illustre le processus de recherche typique de l'algorithme UMHexagonS avec une fenêtre de recherche de taille 32×32.

Les auteurs utilisent un prédicteur médian pour la prédiction des vecteurs de mouvements ($pred_{mv} = median(MV_A, MV_B, MV_C)$). L'ordre de recherche des modes de prédiction est modifié en fonction de

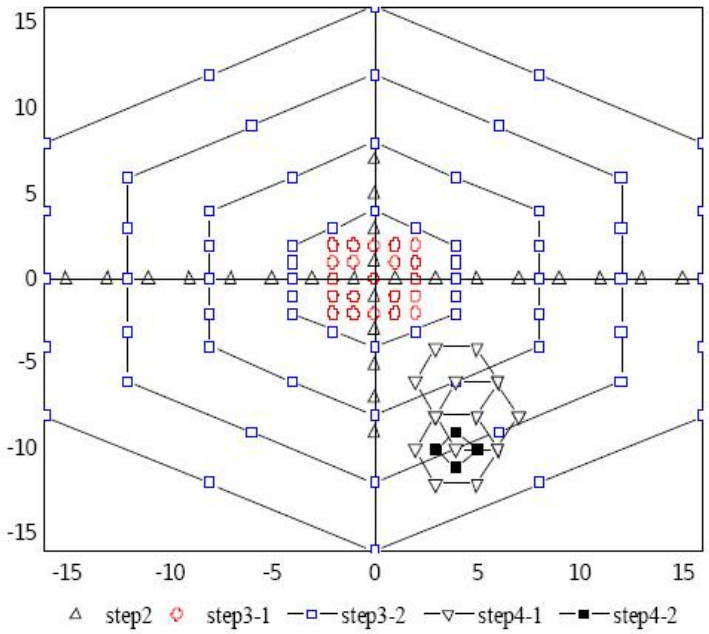


FIG. 2.4 – Processus de recherche de l'algorithme UMHexagonS avec une fenêtre de taille 32×32.

la taille des blocs. Un ordre de recherche hiérarchique (16×16 à 4×4) est choisi pour la prédiction et le vecteur de mouvement de la couche supérieure est utilisé comme candidat pour la prédiction des couches inférieures. Pour le mode 1 (16×16), la prédiction médiane, le vecteur (0,0), et les vecteurs de mouvement des blocs adjacents sont choisis comme candidats pour la prédiction du mouvement du bloc courant. Pour les autres modes, la prédiction médiane, le vecteur (0,0) et le vecteur de mouvement de la couche supérieure sont choisis comme candidats pour la prédiction du mouvement du bloc courant. La prédiction avec l'erreur la plus faible parmi ces candidats est choisie comme le centre de la recherche pour l'étape suivante.

Dans les séquences vidéos naturelles, les mouvements sont souvent plus importants dans la direction horizontale que dans la direction verticale. Le vecteur de mouvement optimal peut être prédit par une recherche asymétrique. La figure 2.4 (étape 2) illustre une recherche asymétrique avec un intervalle de recherche égal à I dans la direction horizontale et de $I/2$ dans la direction verticale.

La troisième étape est divisée en 2 sous étapes. La première est une recherche complète autour du centre avec un intervalle de recherche égal à 2 (cf figure 2.4, étape 3-1). Ensuite, une stratégie de recherche avec des grilles hexagonales multiples est utilisée (étape 3-2). Cette stratégie de recherche est basée sur la considération que la recherche asymétrique donne un point de départ exact pour la recherche et que les multiples grilles hexagonales (inégales) traitent correctement les mouvements larges et irréguliers. Les autres hexagones sont construits en étendant le premier avec différents facteurs allant de 1 à $W/4$. La recherche s'enchaîne de l'hexagone intérieur à l'hexagone extérieur.

Lorsque le vecteur de mouvement optimal dans l'étape trois est situé sur l'hexagone extérieur, le résultat est inexact et des améliorations sont effectuées à l'aide de méthodes de recherches hexagonales centrées (diamant à quatre points ou hexagone à six points). Pour les petites prédictions, (mode 7 : 4×4) la stratégie de recherche peut aller directement à cette dernière étape en négligeant les étapes précédentes car la prédiction du vecteur de mouvement pour les modes de petites tailles est assez précise.

Pour obtenir une précision sous-pixélique des vecteurs de mouvement, ils proposent également une nouvelle technique de recherche (*Center biased Fractional Pel Search*). Le vecteur prédit pour la recherche au pixel près contient l'information du vecteur de mouvement entier prédit et la partie fractionnelle. Cependant, on peut extraire la partie fractionnelle en utilisant la formule suivante :

$$frac_{MV_P} = (MV_P - MV)\beta, \tag{2.1}$$

où MV est le vecteur de mouvement entier du bloc courant, $\beta = 4$ pour le cas 1/4 de pixel et $\beta = 8$ pour le cas 1/8 de pixel. Ensuite, une recherche en diamant est utilisée, de part sa simplicité et son efficacité. La recherche se termine lorsque le minimum (SAD[8]) est situé au centre du diamant. Afin de terminer le processus plus rapidement, ils proposent une technique d'arrêt basée sur la détection des blocs nuls. Les tests réalisés sur des séquences au format HD montrent une baisse de $0.02dB$ (PSNR[9]) et une baisse des temps de calculs de plus de 75%.

Par la suite, quelques modifications ont été apportées à ces nouvelles techniques [Chen *et al.*, 03b, Chen *et al.*, 03a]. En effet, les auteurs reviennent sur deux points de l'algorithme UMHexagonS, qui sont la prédiction du centre de la recherche et le processus d'arrêt rapide. Pour la prédiction du vecteur de mouvement, ils suggèrent trois nouvelles méthodes en plus de la prédiction médiane :

- prédiction à partir de la couche supérieure,
- prédiction à partir du bloc correspondant,
- prédiction à partir de l'image référence voisine.

La SAD de blocs spatialement ou temporellement adjacents étant fortement corrélée, ils utilisent également ces quatre techniques pour prédire la SAD et ainsi permettre d'arrêter la recherche lorsque la SAD du bloc courant est inférieure à un seuil prédéterminé.

En moyenne, avec ces quelques modifications, les temps de calcul sont réduits de plus de 90% par rapport au logiciel de référence qui effectue une recherche complète, pour une perte de $0,04dB$ (PSNR).

Ma et Qiu [Ma, 03] proposent une méthode plus rapide que l'algorithme *UMHexagonS*. Ils partent de la constatation que les vecteurs de mouvements sont plus fortement concentrés le long des axes verticaux et horizontaux. La recherche initiale s'effectue sur quatre points plus le centre (0,0). Ces

[8] *Sum of Absolute Differences*
[9] *Peak Signal to Noise Ratio*

quatre points sont obtenus en fonction du vecteur de mouvement prédit par la médiane des vecteurs des blocs voisins. Ces quatre points sont désignés par les vecteurs MV_1 à MV_4 :

$$MV_1 = (max(MV_x), MV_{y\ predit}), \tag{2.2}$$

$$MV_2 = (minMV_x), MV_{y\ predit}), \tag{2.3}$$

$$MV_3 = (MV_{x\ predit}, min(MV_x)), \tag{2.4}$$

$$MV_4 = (MV_{x\ predit}, max(MV_x)), \tag{2.5}$$

où MV_x et MV_y sont les composantes horizontales et verticales des vecteurs de mouvement des blocs voisins et $MV_{y\ predit}$ et $MV_{x\ predit}$ sont les composantes du vecteur de mouvement prédit. Ensuite, on place le centre de la forme ainsi obtenue sur le vecteur de mouvement prédit. Puis, on déplace la forme sur le point qui minimise le coût débit – distorsion, on réitère cette dernière étape jusqu'à ce que le point minimisant cette erreur soit situé au centre (cf figure 2.5.

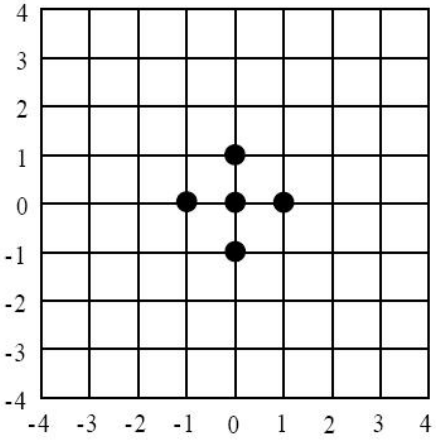


FIG. 2.5 – Forme de taille unitaire.

Par rapport à la recherche exhaustive du codeur de référence, la méthode proposée par Ma et Qiu est 370 fois plus rapide, pour une perte de $0,111dB$ (PSNR) en moyenne.

D'autres méthodes sont basées sur la corrélation spatiale [Su, 04] ou temporelle [Chen *et al.*, 04] entre les images. En effet, Su et Sun [Su, 04] prennent en considération la corrélation/continuité des vecteurs de mouvement entre les images de référence :

$$MV_n^{-k} = MV_n^{-k_1} + MV_{n-k_1}^{(k-k_1)}, \tag{2.6}$$

où MV_n^{-k} représente le vecteur de mouvement entre l'image courante n et l'image référence $n-k$, $MV_n^{-k_1}$ représente le vecteur de mouvement entre l'image courante n et l'image référence $n-k_1$. Et $MV_{n-k_1}^{(k-k_1)}$ représente le vecteur de mouvement entre l'image courante $n-k_1$ et l'image référence $n-k-k_1$ (cf figure 2.6).

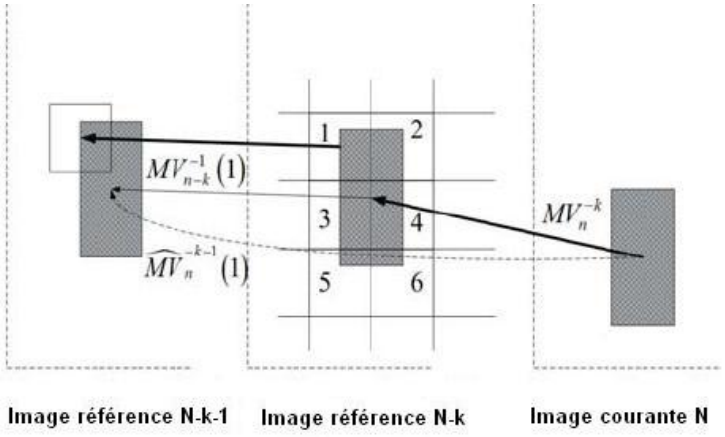


FIG. 2.6 – Ilustration de la corrélation/continuité des vecteurs de mouvement.

Chen et ses collaborateurs [Chen *et al.*, 04] proposent de réutiliser les informations des vecteurs de mouvement obtenus pour les images de référence précédentes. Leur algorithme utilise les informations stockées sur les vecteurs de mouvement pour composer le vecteur de mouvement de l'image courante sans réaliser la recherche complète sur chaque image référence.

2.2.3 Estimation du mouvement avec une nouvelle métrique

La plupart des algorithmes proposés dans la littérature essaient d'accélérer le processus de l'estimation de mouvement au détriment du débit et de la qualité. En effet, la méthode classique de recherche exhaustive obtient les meilleurs résultats, puisque tous les points sont testés. Au contraire, Mai et ses collaborateurs [Mai *et al.*, 06] proposent une méthode afin de réduire le flux de données tout en préservant la qualité perceptuelle. Leur algorithme est une méthode d'estimation de mouvement basée sur la SSIM [Wang *et al.*, 04] (*Structural Similarity*). La SSIM est une métrique de qualité basée sur la mesure de la dégradation de l'information structurale, qui est composée de :

- la luminance,
- le contraste,
- la structure.

La SSIM est intégrée au sein de leur codeur H.264 en lieu et place de la traditionnelle SAD. En moyenne, le flux de données est réduit de 20% et les temps de calculs diminuent de $2,5\%$ par rapport au codeur de référence, tout en maintenant la même qualité perceptuelle de vidéo reconstruite.

2.3 Méthodes de réduction des modes de prédiction testés

2.3.1 Sélection rapide du mode

Huang et ses collaborateurs [Huang *et al.*, 06] proposent un algorithme de sélection du mode en deux étapes. La première étape consiste à prédire quelques modes possibles de codage du macrobloc

courant en fonction des images et macroblocs précédents. Ensuite, ils affinent la décision en utilisant les règles de Baye basées sur les probabilités d'appartenance à un groupe, et un réseau de neurones à propagation arrière (BPN[10]).

Dans la première étape, les modes sont divisés en quatre types :

- grand : 16×16, 16×8 et 8×16,
- petit : 8×8, 8×4, 4×8 et 4×4,
- skip $SKIP$,
- intra : 16×16 et 4×4.

L'algorithme proposé essaie de choisir entre l'un de ces quatre types. Les seules informations dont dispose l'algorithme, sont les modes de codage utilisés pour coder l'image précédente et les macroblocs de l'image courante. Le type candidat est sélectionné en fonction du mode le plus utilisé pour coder ces treize macroblocs (neuf macroblocs dans l'image précédente et les quatre voisins précédemment codés dans l'image courante).

Dans la seconde étape, une décision pour affiner le choix est effectuée basée sur le résultat de l'étape précédente puisqu'il est possible que le type choisi à l'étape précédente ne soit pas optimal. Soient p_1 la probabilité pour que le type 1 soit le type optimal et p_2, p_3 et p_4 les probabilités que le type 1 ne soit pas optimal ; le type optimal pouvant être de type 2, type 3 ou type 4, respectivement. Ainsi, le problème de choix de type peut être transformé en problème de transition d'état à état, chaque type étant considéré comme un état. Ces probabilités de transition d'un état aux trois autres états sont formulées de la sorte :

$$Probabilite\ de\ transition = p_t(x|s), \tag{2.7}$$

où s et t représentent l'état actuel et l'état de destination respectivement. Choisir le type le plus probable, revient à sélectionner le type avec la probabilité de transition la plus élevée. Dans certaines situations, deux probabilités de transition d'un état donné s sont proches l'une de l'autre et il est difficile de prendre la décision. Dans ce cas là, un autre algorithme de prise de décision est nécessaire. Les auteurs utilisent un réseau de neurones à propagation arrière (BPN). Ainsi, le choix du type devient un problème de classification. Le BPN essaye de classifier le type final en utilisant l'information obtenue en codant le macrobloc courant. Après avoir décidé du type dans la première étape, les coûts débit – distorsion des différents modes sont obtenus, ceux-ci étant employés comme entrées du BPN.

Ils utilisent douze séquences vidéo au format CIF[11], cinq d'entre elles sont utilisées pour générer les paramètres du modèle (apprentissage) et du réseau de neurones à propagation arrière. Ils utilisent les sept dernières séquences pour tester leur algorithme. Leur méthode permet de réduire de 84% les temps de calcul et 50% au minimum, pour un débit et une qualité quasi équivalents.

[10] *Back-Propagation neural network*
[11] *Common Intermediate Format*

2.3.2 Décision pour les modes intra

Une méthode basée sur une transformation d'entiers et un seuil adaptatif [Su *et al.*, 06] a été proposée afin d'accélérer la décision pour le choix du mode de codage intra des macroblocs.

Le codeur H264/AVC réalise une optimisation débit – distorsion pour chaque macrobloc afin d'obtenir le meilleur mode de codage pour la prédiction intra (et inter). Or le mode optimal pour la prédiction intra dépend fortement de la direction de la texture (voir chapitre 1). Leur méthode [Su *et al.*, 06] réalise la détection des contours pour un bloc entier. Ils utilisent une simple transformation d'entiers (une seule fois) car celle-ci leur permet de ne pas réaliser les calculs dans chaque direction comme c'est le cas avec les opérateurs de Sobel. Cependant, les résultats obtenus avec cette transformation d'entiers comparés à ceux des opérateurs de Sobel ne sont pas exacts. De ce fait, un plus grand nombre de modes intra doit être testé. Ils utilisent ensuite un seuil sur la SAD du meilleur mode intra pour le bloc 4×4. Si celle-ci est inférieure au seuil, les calculs s'arrêtent pour les blocs de taille 4×4. Pour finir, ils comparent les coûts débit – distorsion obtenus pour le mode 4×4 retenu et le meilleur mode 16×16, le mode avec la meilleure valeur étant retenu. Les tests réalisés montrent une diminution des temps de calcul de 50% pour des séquences vidéo au format CIF où toutes les images sont codées en intra.

2.3.3 Décision pour les modes inter

Comme nous l'avons vu au chapitre 1, il existe sept modes inter de codage possibles pour les macroblocs ce qui permet d'améliorer la performance du codeur en terme de débit – distorsion, mais augmentent les temps de calcul. De nombreuses méthodes ont été proposées afin de sélectionner et de réduire le nombre de modes inter testés [Yu, 04b, Yu, 04a, Yu *et al.*, 05, Chen *et al.*, 05, Bu *et al.*, 06].

L'algorithme proposé par Yu [Yu, 04a] repose sur deux facteurs prédictifs fiables et robustes, qui sont la complexité intrinsèque du macrobloc et le mode utilisé dans l'image précédente pour le bloc situé à la même position. La réussite de l'algorithme proposé repose sur le fait que les tailles de blocs les moins probables ne sont pas testées. Le premier point est d'appliquer une mesure de la complexité pour chaque macrobloc. Pour mesurer la complexité des macroblocs, l'auteur propose un algorithme qui somme l'énergie des coefficients AC des macroblocs afin d'estimer le contenu.

Le critère pour évaluer la complexité R_B d'un macrobloc MB est,

$$R_B = \frac{ln(E_{AC})}{ln(E_{max})}, \tag{2.8}$$

où E_{AC} est l'énergie totale du bloc qui est égale à l'énergie cumulée de ses coefficients de DCT et E_{max} est la somme maximale des coefficients AC d'un bloc de même taille. Un macrobloc dont la valeur R_B est supérieure à $0,7$ est considéré comme un bloc contenant de nombreux détails.

Seulement trois catégories de blocs sont testées pour la mesure de complexité (16×16, 8×8 et 4×4). Celles-ci sont notées $MD16$, $MD8$ et $MD4$ respectivement. D'abord, un macrobloc de taille

16×16 est examiné. Il est noté $MD16$ s'il est reconnu comme étant homogène. Sinon, le macrobloc est décomposé en 4 blocs de taille 8×8. Un bloc 8×8 est reconnu comme contenant beaucoup de détails s'il satisfait deux conditions :

- R_B est supérieur à $0,7$ et est décomposé en 4 blocs de taille 4×4,
- un des 4 blocs décomposé contient beaucoup de détails.

Si un bloc 8×8 satisfait la première condition mais pas la deuxième, il est reconnu comme étant « peu détaillé ». Un macrobloc est noté $MD8$ s'il possède plus de deux blocs qui contiennent beaucoup de détails, sinon il est noté $MD4$. Afin d'améliorer la méthode, on tient compte du mode utilisé pour le macrobloc situé à la même position dans l'image précédente.

Pour des séquences contenant beaucoup de mouvements (*Stefan, Table tennis, Mobile*), le temps de calcul en moins varie entre 17% et 31% avec une perte de $0,1dB$ au maximum pour le PSNR et une augmentation du flux de données de $5,35\%$.

Yu et Martin proposent un algorithme de prédiction du mode de décision [Yu, 04b] plus efficace que la méthode de Yu [Yu, 04a]. L'efficacité de l'algorithme proposé est réalisée en introduisant deux mesures additionnelles, qui permettent d'identifier deux catégories de macroblocs :

- les macroblocs codés avec le mode SKIP,
- les macroblocs codés par les modes inter de grande taille (partition supérieure à 8×8).

En identifiant correctement ces deux catégories de macroblocs, le codeur est exempté de les examiner avec tous les modes inter possibles.

Le mode SKIP est normalement attribué à un macrobloc qui comprend pratiquement les mêmes informations pixéliques que le macrobloc correspondant dans l'image précédente. Les macroblocs codés avec le mode SKIP peuvent facilement être détectés en comparant les résidus entre le macrobloc courant et le macrobloc correspondant dans l'image précédente,

$$T(S_{residu}) = \begin{cases} 1, & S_{residu} < Th, \\ 0, & S_{residu} > Th, \end{cases}$$

$$S_{residu} = \sum_m \sum_n \left| \mathbf{B}_{m,n,t} - \mathbf{B}_{m,n,t-1} \right| .$$

Si $T(S_{residu}) = 1$, le macrobloc courant est codé en mode SKIP. Afin de ne pas effectuer ce calcul pour chaque macrobloc inutilement, les auteurs proposent d'ajouter une condition au macrobloc courant. Si un des voisins du macrobloc courant est codé en mode SKIP, alors on réalise la mesure de similarité temporelle. Ensuite, les vecteurs de mouvement de chaque bloc 8×8 appartenant à un macrobloc contenant beaucoup de détails sont examinés. Si ceux-ci sont suffisamment uniformes, l'algorithme vérifie les modes inter avec des partitions de taille supérieure à 8×8, sinon tous les modes inter possibles sont testés.

Pour des séquences au format QCIF contenant beaucoup de mouvements, la diminution des temps de calcul varient entre 28% et 50% avec une perte de $0,13dB$ au maximum pour le PSNR et une

augmentation du flux de $8,90\%$.

Chen et ses collaborateurs [Chen *et al.*, 05] proposent quant à eux une méthode basée sur les points suivants :

1. une détection rapide des blocs stationnaires et des blocs suivants les mêmes déplacements est réalisée pour « capturer » le plus de blocs 16×16 ;
2. une recherche prédictive des vecteurs déplacement avec une détection rapide des coefficients DCT nuls est appliquée sur les blocs 4×4 dans les macroblocs non retenus par l'étape 1 ;
3. une méthode de fusionnement ascendant est appliquée sur les vecteurs de déplacement des blocs 4×4.

Les blocs de taille 16×16 occupent généralement près de 70% [Chen *et al.*, 05] de l'image. Si ces blocs 16×16 sont isolés au départ, les procédures qui suivent peuvent être supprimées, ce qui diminue les temps de calcul. La prédétection des blocs 16×16 peut être effectuée en calculant la SAD pour un ensemble de vecteurs de mouvement prédits candidats, et ensuite on compare leur SAD avec un seuil. Si la SAD minimale est inférieure au seuil, le bloc courant est étiqueté comme un bloc 16×16 et les vecteurs de mouvement prédits correspondants deviennent les vecteurs de mouvement du bloc. Si le vecteur de mouvement prédit est égal à $(0, 0)$, le bloc est identifié comme un bloc stationnaire.

L'ensemble des vecteurs de mouvement prédits candidats sont obtenus à l'aide de deux techniques :

- prédiction de la médiane,
- prédiction par rapport à la dernière image,
- et le vecteur $(0, 0)$ (pour isoler les blocs stationnaires).

Une recherche en diamant est utilisée pour obtenir les vecteurs de déplacement des blocs 4×4. Le centre de la fenêtre de recherche est déterminé par l'étape précédente (le vecteur de mouvement prédit obtenant la plus petite SAD). Si la SAD obtenue est inférieure à un seuil, la procédure peut se terminer rapidement, car il est inutile alors d'effectuer la DCT pour ce bloc 4×4. Si la différence entre les vecteurs déplacement des blocs voisins est inférieure à un seuil (ce qui signifie qu'ils se déplacent dans la même direction), alors ils fusionnent ensemble :

$$Dist_{MV_1,MV_2} = max\{|MV_{1_x} - MV_{2_x}|, |MV_{1_y} - MV_{2_y}|\}, \tag{2.9}$$

où MV_{1_x} et MV_{1_y} sont les composantes horizontale et verticale du vecteur de mouvement MV_1. Si deux blocs fusionnent, une recherche en diamant est effectuée avec une fenêtre de recherche $(-2, 2)$.

L'algorithme proposé permet un gain de $29,24\%$ en temps de calcul pour une perte de $0,096dB$ (PSNR) et une augmentation de $5,07\%$ pour le flux.

Il existe également des méthodes avec un processus de recherche plus sophistiqué et des techniques de prédiction plus robustes en sélectionnant un mode inter optimal pour chaque macrobloc [Yu *et al.*, 05].

Le schéma proposé par les auteurs comprend trois niveaux. Chaque niveau cible une catégorie de modes inter en fonction de la complexité du processus de recherche.

Le premier niveau de l'algorithme permet de déterminer les macroblocs codés en mode SKIP. Ceux-ci peuvent être détectés en fonction de la similarité temporelle. La similarité temporelle d'un macrobloc est testée si au moins un des deux macrobloc situés au dessus et à gauche est codé en mode SKIP. La décision pour la détection de similarité temporelle est définie par,

$$Decision = \left\lfloor \frac{X_{SAD}}{SAD_{courant}(t;t-1)} + \epsilon \right\rfloor, \tag{2.10}$$

où $SAD_{courant}(t;t-1)$ est la SAD des macroblocs des images t et $t-1$, ϵ représente une tolérance constante et X_{SAD} est la SAD du voisin codé en mode SKIP (si les deux voisins sont codés en mode SKIP, on calcule la moyenne des deux SAD). Si le résultat obtenu est non nul, cela indique que le macrobloc courant est codé en mode SKIP, sinon plusieurs recherches au niveau 2 doivent être réalisées.

Le second niveau permet d'identifier les macroblocs qui doivent être codés en mode inter avec une grande partition (8×16, 16×8 et 16×16). Ici, la mesure de complexité spatiale ($R_B = \frac{ln(E_{AC})}{ln(E_{max})}$) est utilisée afin de déterminer si le macrobloc courant a besoin d'être examiné par les modes inter pour les partitions de taille plus petite. Si le meilleur mode pour un macrobloc contenant beaucoup de détails est un mode inter avec une partition de taille 8×16 ou 16×8, une recherche plus précise dans le dernier niveau est requise.

Pour le troisième et dernier niveau, l'algorithme effectue les recherches pour les modes inter de taille 8×8 et inférieures. Chaque macrobloc est décomposé en quatre blocs de taille 8×8. Chaque bloc est testé avec le mode inter 8×8. Les autres modes inter avec des partitions de taille plus petite sont testés si la condition suivante est vérifiée,

$$LC_{8\times8}(N^{eme}) < LC'/4, \tag{2.11}$$

où $LC_{8\times8}(N^{eme})$ est le coût débit – distorsion du N^{eme} bloc 8×8 courant et LC' représente le coût débit – distorsion du meilleur mode obtenu au niveau précédent.

Pour les séquences contenant beaucoup de mouvement (*Stefan, Table tennis,...*), le temps de calcul en moins varie entre 41% et 55%. Les résultats montrent que l'algorithme proposé obtient les mêmes résultats que le standard H.264/AVC en terme de qualité d'image et de taux de compression tout en réduisant les temps de calcul.

La dernière de ces méthodes [Bu *et al.*, 06], réalise une prédiction sélective de la taille des blocs. Les auteurs ont constaté que lorsque les macroblocs de taille 16×16, 8×16 et 16×8 sont identifiés comme étant les meilleurs modes ceux-ci ont généralement une SAD faible. Et au contraire, lorsque leur SAD est importante, toutes les tailles de macroblocs doivent être testées. L'algorithme se divise en plusieurs étapes. Il teste d'abord les blocs de taille 16×16 et si la SAD est inférieure à un seuil, les blocs de taille

inférieure à 8×8 ne sont pas testés. Ensuite, ils proposent une méthode afin de déterminer si les blocs de taille 8×8 sont stationnaires afin de ne pas tester les modes de tailles inférieures (8×4, 4×16 et 4×4). Si la SAD obtenue pour le bloc de taille 8×8 est inférieure à un seuil, ils identifient ce bloc comme étant stationnaire. Dans le cas contraire, ils testent les autres partitions de tailles inférieures. Les tests réalisés sur six séquences au format CIF montrent une diminution des temps de calcul de 20% pour les séquences avec des mouvements importants et jusqu'à 45% pour des séquences avec peu de mouvement.

2.4 Choix des images références

La dernière catégorie de techniques permettant d'optimiser le codeur H.264/AVC, s'intéresse aux images références utilisées par ce celui-ci. En effet, nous avons vu précédemment que le codeur peut utiliser jusqu'à cinq images références pour la prédiction inter des macroblocs de l'image courante. Or par défaut, le codeur de référence utilise les cinq images précédant l'image courante. Cette dernière section présente plusieurs méthodes traitant des images références et plus particulièrement de leur sélection.

2.4.1 Recherche partielle

Avec le logiciel de référence, 80% [Huang *et al.*, 03] des vecteurs de déplacement optimaux pour coder l'image courante sont obtenus à partir de l'image précédente (ref_0). Huang et ses collaborateurs proposent une méthode où une recherche approfondie [Huang *et al.*, 03] pour les modes intra et inter sur l'image précédente est d'abord réalisée. Ensuite, ils étudient les informations disponibles, telles que, le mode choisi, l'erreur de prédiction intra, l'erreur de prédiction inter et les vecteurs de mouvement, afin de déterminer s'il est utile d'effectuer la recherche sur les autres images références.

Quand un macrobloc est partitionné en blocs plus petits pour la compensation de mouvement en utilisant une seule image référence, cela signifie que le mouvement n'est pas continu. Et lorsque le mode intra obtient de meilleurs résultats que la prédiction inter à partir de l'image précédente, le macrobloc doit appartenir à des zones découvertes ou de nouveaux objets. Il y a donc de grandes chances, que les meilleurs résultats soient obtenus avec les quatre autres images références dans ces deux cas.

La définition de la compacité des vecteurs de mouvement est donnée dans la figure 2.7. Si la compacité d'un macrobloc après l'estimation du mouvement avec l'image précédente est faible, il est inutile d'effectuer la recherche sur les quatre autres images.

Ils prennent également la texture en considération. En effet, la réduction de l'erreur de prédiction en utilisant plusieurs images références est plus significative aux limites des objets, lorsque des recouvrements ou des découvrements se produisent. Ils utilisent la SATD[12] après la prédiction intra pour représenter la complexité de la texture d'un macrobloc. Si un macrobloc contient beaucoup de texture

[12] *2-D* 4×4 *Hadamard Transformed SAD*

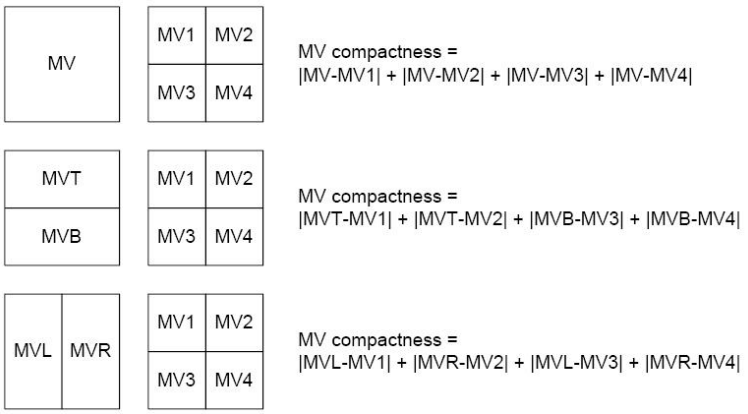


FIG. 2.7 – Définition de la compacité des vecteurs de mouvement d'un macrobloc.

(SATD élevée), il devient nécessaire d'effectuer la recherche dans les autres images références, mais le seuil de la SATD doit s'adapter avec les différentes scènes ou séquences.

Les résultats montrent que leur méthode permet de réaliser entre 10% et 67% de temps de calcul en moins, avec une perte de $0,2dB$ au maximum (PSNR), celle-ci étant en moyenne de $0,05dB$. Mais leur algorithme n'est pas parfait puisqu'en moyenne $28,20\%$ de calculs sont réalisés en trop (la recherche est effectuée dans les quatre autres images références, alors que le meilleur résultat est obtenu avec ref_0).

Des stratégies de recherche adaptative pour une estimation de mouvement pixélique avec un critère flexible de recherche multi-images ont également été proposées [Li *et al.*, 04]. Dans cet article, les stratégies adaptatives de recherche de mouvement sont utilisées pour chaque image de référence. Ensuite, un test permet de prendre la décision de continuer la recherche sur les deux images de référence les plus éloignées temporellement. Les auteurs définissent utilisent quatre méthodes pour prédire le vecteur de mouvement de l'image courante (deux spatiales et deux temporelles) :

- vecteurs de mouvement des blocs voisins,
- vecteurs de mouvement des blocs des couches supérieures,
- vecteurs de mouvement du bloc correspondant dans l'image précédente,
- vecteurs de mouvement dans l'image référence précédente.

Les vecteurs de mouvement sont généralement distribués dans une zone en forme de diamant. La taille de ce diamant est obtenue à partir des informations concernant les vecteurs déplacement, qui est en fait très proche de la médiane des vecteurs de mouvement des blocs voisins. Si la prédiction indique que l'amplitude des vecteurs de mouvement restent faible, alors une recherche en diamant est réalisée, sinon on effectue la méthode de recherche hexagonale. Si les vecteurs de mouvement et les coûts de codage ne sont pas trop importants, il est alors inutile d'effectuer la recherche sur les deux images références les plus éloignées temporellement de l'image courante.

Les tests réalisés avec des séquences au format CIF et QCIF, montrent que leur méthode réalise le

codage seize fois plus vite que la méthode UMHexagonS [Chen *et al.*, 02], pour une perte de $0,053dB$ (PSNR).

Une autre méthode, afin de ne pas effectuer totalement les recherches des vecteurs de mouvement sur toutes les images références, a été proposée [Li *et al.*, 06]. Les auteurs proposent un algorithme rapide d'estimation de mouvement sur plusieurs images références :

1. estimation classique de mouvement sur l'image précédente ref_0,
2. recherche rapide (quatre modifications) sur les quatre autres image références, ref_1 à ref_4,
3. choix de la meilleure référence (de ref_1 à ref_4),
4. comparaison entre la meilleure référence obtenue et ref_0,
5. si ref_0 est la meilleure $\Rightarrow$ FIN
   sinon estimation de mouvement classique sur la meilleure référence.

Pour les quatre images références restantes, les auteurs ont proposés quelques modifications afin d'accélérer le processus d'estimation de mouvement. Ils effectuent seulement une recherche au pixel près et non au 1/4 ou au 1/8 de pixel comme c'est le cas dans le codeur H.264/AVC. En effet, l'estimation de mouvement est réalisée sur plusieurs images références ce qui est une « sorte de compensation de mouvement à multi-hypothèses ». Or il a été prouvé [Girod, 00] que l'exactitude d'une estimation de mouvement au 1/2 pixel est moins importante dans le cas d'un compensation de mouvement à multi-hypothèses. Ils utilisent la corrélation temporelle entre les vecteurs de mouvement pour obtenir le vecteur de mouvement prédit et le centre de la fenêtre de recherche. L'estimation de mouvement n'est réalisée que pour des blocs de taille 8×8 sur des fenêtres de tailles réduites.

Le choix de la meilleure référence se fait en 2 étapes. La meilleure référence possible (PBR[13]) est choisie parmi les images références (de ref_1 à ref_4) pour chaque partition. Les coûts de codage correspondants aux différentes images références pour des blocs 8×8 sont obtenus à la fin de l'estimation de mouvement modifiée. Pour les partitions supérieures à 8×8, on utilise une combinaison des coûts de codage des blocs 8×8. Ensuite, les coûts débit – distorsion de la meilleure référence possible choisie et de ref_0 sont comparés,

$$MCost_{PBR} \geq \alpha \times MCost_{ref_0}, \quad avec\ \alpha > 1. \tag{2.12}$$

. Si l'inéquation est vérifiée, ref_0 est alors la meilleure référence, sinon une estimation de mouvement complète sur l'image référence retenue est réalisée. Ainsi, dans le pire des cas, seulement deux images subissent une estimation de mouvement complète. Avec de telles modifications, ils obtiennent entre 34% et 47% de temps de calcul en moins, avec une perte de $0,2dB$ au maximum pour le PSNR et une augmentation de $0,12\%$ du flux de données.

[13] *Possible Best Reference*

2.4.2 Sélection des images de référence

La dernière famille de méthodes pour l'optimisation du codeur H.264/AVC, concerne la sélection des images références. En effet, par défaut le codeur H.264/AVC de référence utilise les cinq images précédentes à l'image courante pour remplir la mémoire tampon. Aujourd'hui, encore très peu de méthodes permettant une sélection des ces images références ont été proposées. L'une d'entre elles repose sur la similarité des histogrammes pour sélectionner le meilleur ensemble d'images références [Ozbek, 05].

Les images clés qui sont les images les plus représentatives d'une séquence vidéo sont généralement utilisées pour la récupération, l'indexage et les résumés de vidéos [Ozbek, 05]. Ozbek et Tekalp proposent d'utiliser ces méthodes de sélection des images clés afin de réaliser un choix rapide des images références pour un groupe d'images.

L'image dont l'histogramme est le plus proche de l'histogramme moyen est choisie comme image clé principale pour le groupe d'images sélectionnées. Ils définissent trois cas de choix différents :

- l'image clé principale et les deux autres images les plus proches de l'histogramme moyen,
- l'image clé principale et les deux autres images les plus éloignées de l'histogramme moyen,
- l'image clé principale et l'image la plus éloignée de l'histogramme moyen.

Au niveau du codage, la méthode intervient directement sur la gestion de la mémoire tampon des images de référence. Le but est de conserver les images clés choisies dans la mémoire tampon et de modifier l'ordre de codage. Il est nécessaire de coder ces images d'abord afin qu'elles puissent être utilisées comme références. Ils obtiennent les meilleurs résultats avec le troisième cas (l'image clé principale et l'image la plus éloignée de l'histogramme moyen) avec 23% de temps de calcul en moins, mais également des variations de débit. Cependant, ces résultats indiquent que le codage vidéo H.264/AVC rapide avec gestion des multiples images de référence peut être atteint avec une qualité et un débit identique à la méthode de référence.

Plutôt que de considérer les histogrammes des images, Yuan et ses collaborateurs [Yuan *et al.*, 04] proposent d'étudier les variations des vecteurs de mouvement afin de choisir les images références. En simplifiant la structure d'une séquence à un ensemble d'images I ou B, les images clés sont alors des images I et l'intervalle entre images clés est l'intervalle entre deux images I (ce qui correspond au nombre d'images B). Le cas extrême d'une séquence ne contenant que des images I ne permet pas une compression efficace, car aucune corrélation temporelle n'est utilisée. Néanmoins, un intervalle trop grand n'apporte pas toujours une meilleure compression. Les images B sont alors codées en intra car les macroblocs ne peuvent trouver de correspondants dans les images I trop « lointaines ». De plus, en augmentant l'intervalle entre images clés, les erreurs résiduelles de prédiction des images B seront plus importantes. Yuan et ses collaborateurs proposent un algorithme qui repose sur l'évolution du mouvement dans la séquence. Des calculs de différences (norme euclidienne au carré des vecteurs de mouvement) entre deux images de référence permettent de suivre l'évolution du mouvement. Se-

lon les résultats de ces calculs et en fonction de plusieurs paramètres dont les valeurs sont obtenues empiriquement, l'image courante sera alors une image I ou P.

Les résultats de cette méthode sont intéressants, puisque le PSNR est augmenté de $0,4$ à $0,9dB$, pour un temps de codage 5 à 11% plus long. L'évaluation subjective de la qualité montre quant à elle une augmentation de $1,3$ sur 10.

Cette méthode est intéressante car elle démontre l'intérêt du choix des images de référence et de leur positionnement. Cependant, elle ne prend pas en compte le fait que les images B puissent également être utilisées comme images de référence dans le codeur H.264/AVC.

2.5 Conclusion

Ce chapitre a présenté un aperçu de plusieurs familles de méthodes d'optimisation de codage. Nous avons vu qu'il existe trois grandes familles de techniques permettant d'optimiser le codeur H.264/AVC. La première d'entre elles regroupe les méthodes d'estimation de mouvement qui accélèrent la recherche du vecteur de mouvement. Celle-ci est constituée des méthodes multi-résolutions et de techniques effectuant des recherches partielles dont certaines ont été adoptées par le JVT [Chen *et al.*, 02, Chen *et al.*, 03b, Chen *et al.*, 03a, Xu *et al.*, 03]. La deuxième famille regroupe les techniques qui permettent de réduire les modes de prédiction testés (intra et/ou inter). La dernière famille présente les techniques de sélection des images de référence (par défaut, le codeur utilise les cinq images précédant l'image courante). Cependant, encore très peu de méthodes réalisant un choix des images références ont été proposées dans la littérature. Cette notion d'image de référence est très importante, c'est la base de la prédiction inter et la gestion de ces images est donc primordiale. C'est donc sur ce point que nous allons chercher en particulier une optimisation du codage. On peut déjà envisager de construire le GOP en fonction des résultats obtenus à l'aide de l'étude de mouvement de la séquence vidéo. Une construction judicieuse du GOP permettra sans doute de réaliser une bonne compression et d'obtenir une bonne qualité visuelle. À cette fin, nous voulons utiliser un schéma d'analyse suivant l'axe temporel de la vidéo : MCTF (*Motion Compensated temporal filtering*, soit filtrage temporel à compensation de mouvement).

Chapitre 3

H.264/SVC étendu à la graduabilité

3.1 Introduction

Afin d'évaluer et de comparer les différentes méthodes de compression vidéo graduable, le groupe MPEG a lancé un *Call for Evidence*[CfE, 03], auquel ont répondu avec succès plusieurs universités et entreprises. Le groupe MPEG a ensuite rédigé et publié un *Call for Proposals*[CfP, 03], qui consistait à effectuer huit décodages différents à partir d'un train binaire unique et emboîté. Le tableau 3.1 présente les huit décodages proposés par le *Call for Proposals*.

Dimension	Fréquence	Débit
704×576	60 images/s	$6000kb/s$
704×576	60 images/s	$3000kb/s$
704×576	30 images/s	$1500kb/s$
352×288	30 images/s	$750kb/s$
352×288	30 images/s	$384kb/s$
352×288	15 images/se	$196kb/s$
176×144	15 images/s	$125kb/s$
176×144	15 images/s	$64kb/s$

TAB. 3.1 – Décodages évalués pour le Call for Proposals : graduabilité spatiale, graduabilité temporelle et SNR.

À ce jour, il semble qu'aucune étude n'ait été réalisée sur les préférences des utilisateurs face à un contenu de type vidéo graduable. Il serait intéressant de réaliser des tests subjectifs ayant pour objectif d'identifier le ou les paramètres à garantir dans le contexte de diffusion d'une vidéo graduable :

- optimiser la fréquence des images,
- optimiser la résolution des images,
- optimiser la qualité (SNR) images.

La diffusion de vidéos graduables étant limitée par des contraintes physiques : capacité du réseau (bande passante), résolution du terminal sur laquelle elles seront affichées, puissance du processeur du terminal, etc..., il s'agit d'optimiser le contenu afin de répondre au mieu aux attentes des utilisateurs finaux.

La suite de ce chapitre présente les possibilités de codage vidéo graduable offertes par le codeur H.264/AVC.

3.2 Incorporation d'un schéma MCTF au sein du codeur H.264/AVC

En octobre 2004, une extension graduable du codeur H.264/AVC [Schwarz *et al.*, 04] a été choisie pour être le point de départ du projet. Au début de l'année 2005, MPEG et le VCEG[1] se sont accordés pour finaliser le projet de codage vidéo graduable comme un amendement du standard H.264/AVC et l'extension graduable du codeur H.264/AVC a été sélectionnée comme le premier *Working Draft* [Reichel *et al.*, 05].

Le schéma de lifting utilisé dans les approches MCTF (voir le chapitre 4 pour plus d'explications) étant inversible, n'importe quelle technique de compensation de mouvement peut être utilisée dans les étapes de prédiction et de mise à jour du banc de filtres. L'estimation et la compensation de mouvement réalisées par le codeur H.264/AVC peuvent donc être réutilisées pour l'étape de prédiction[Schwarz *et al.*, 04, Schäfer *et al.*, 05]. Pour l'étape de mise à jour, une technique similaire permet également de ne pas refaire les calculs.

Au sein du codeur H.264/AVC, l'estimation du mouvement ne se fait pas pour une taille de bloc fixe. En effet, les macroblocs peuvent être partitionnés en blocs de taille 16×16, 16×8, 8×16 et 8×8 et les partitions de taille 8×8 peuvent être encore divisées en blocs de taille 8×4, 4×8 et 4×4. Pour chaque bloc, un ou deux vecteurs de mouvement sont retenus pour la compensation de mouvement correspondant à l'estimation de mouvement prédictive ou bi-prédictive. Pour les blocs de taille inférieure à 8×8, une seule image de référence est choisie pour le bloc de taille 8×8 auquel ils appartiennent. Tous les autres blocs de taille supérieure à 8×8 sont libres de sélectionner une image de référence différente, contenue dans la mémoire-tampon d'images de référence. Soit $s[l,t]$ un échantillon de la vidéo à la position $l = (x,y)$ à l'instant t et $l \in B$ où B est un bloc. Les opérateurs pour les étapes de prédiction et de mise à jour pour la décomposition temporelle basée MCTF avec un schéma de lifting utilisant les ondelettes de Haar et le bloc B s'écrivent de la forme suivante :

$$P_{Haar}(s[l,2t]) = s[l + m_{P0}, 2t - 2r_{P0}],\ l \in B, \tag{3.1}$$

$$U_{Haar}(h[l,t]) = \frac{1}{2} h[l + m_{U0}, t + r_{U0}],\ l \in B. \tag{3.2}$$

L'étape de prédiction de l'ondelette de Haar correspond au codage prédictif réalisé par le codeur H.264/AVC, utilisant le vecteur de mouvement m_{P0} et l'image de référence r_{P0}. L'étape de mise à jour repose elle aussi sur une compensation de mouvement basée blocs. L'algorithme permettant d'obtenir le vecteur de mouvement m_{U0} et l'index de l'image référence est décrit par la suite.

Si on utilise une transformation ondelettes de type 5/3, les opérateurs pour les étapes de prédiction et de mise à jour sont donnés par :

$$P_{5/3}(s[l,2t]) = \frac{1}{2}(s[l + m_{P0}, 2t - 2r_{P0}] + s[l + m_{P1}, 2t + 2 + 2r_{P1}]),\ l \in B, \tag{3.3}$$

$$U_{5/3}(h[l,t]) = \frac{1}{4}(h[l + m_{U0}, t + r_{U0}] + h[l + m_{U1}, t - 1 - r_{U1}]),\ l \in B. \tag{3.4}$$

[1] *Video Coding Experts Group*

Ici, l'étape de prédiction est la même que la bi-prédiction réalisée par le codeur H.264/AVC pour les images de type B. La prédiction utilise deux listes d'indices d'images références. Ces deux listes sont appelées liste 0 (m_{P0} et r_{P0}) et liste 1 (m_{P1} et r_{P1}) et peuvent contenir les mêmes ou différentes images de référence.

Pour l'étape de mise à jour, les vecteurs et les images de référence sont obtenus de la sorte. Pour chaque bloc 4×4 de luminance dans l'image $U(H_t)$, l'algorithme détermine le vecteur de mouvement et l'image référence associée. Pour chaque bloc $B_{4\times4}$, tous les vecteurs de mouvements m_{P0} et m_{P1} qui pointent sur ce bloc sont évalués. Les vecteurs m_{P0} et m_{P1} qui utilisent le plus grand nombre de pixels du bloc $B_{4\times4}$ comme référence pour la prédiction sont choisis (voir la figure 3.1),et les vecteurs de mouvements pour l'étape de mise à jour sont obtenus de la manière suivante, $m_{U0} = -m_{P0}$ et $m_{U1} = -m_{P1}$. Si aucun vecteur m_{P0} ne pointe sur le bloc $B_{4\times4}$, ou moins de 3/4 des pixels de $B_{4\times4}$ sont utilisés comme référence quels que soient les vecteurs m_{P0}, l'étape de mise à jour utilisant m_{U0} pour le bloc $B_{4\times4}$ est omise (ainsi que pour m_{U1} et m_{P1}). Ensuite, les vecteurs et les images références sont écrits dans la syntaxe employée par le codeur H.264/AVC.

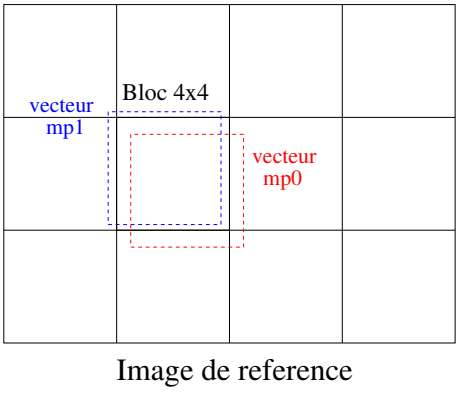


FIG. 3.1 – Sélection des vecteurs m_{P0} et m_{P1} ($m_{U0} = -m_{P0}$ et $m_{U1} = -m_{P1}$) pour l'étape de mise à jour.

Le filtre pour les effets de bloc est appliqué sur les images basses fréquences qui sont reconstruites dans les étapes de prédiction.

Pour les macroblocs ou le mode intra est retenu, les étapes de prédiction et de mise à jour correspondantes ne sont pas effectuées. Les macroblocs sont copiés dans les images de hautes fréquences H_k et codés en intra en utilisant les méthodes du codeur H.264/AVC. Pour la compensation de mouvement réalisée pendant l'étape de mise à jour, les coefficients de ces macroblocs codés en intra sont mis à zéro au préalable.

La syntaxe utilisée par le codeur H.264/AVC n'est pas affectée par l'extension d'un filtrage temporel avec compensation de mouvement, car toutes les données pour l'étape de mise à jour sont dérivées de données qui sont déjà présentes dans le train binaire.

3.3 Extension du codeur H.264/AVC à la graduabilité

3.3.1 Graduabilité temporelle

L'introduction d'un schéma MCTF au sein du codeur H.264/AVC[Schäfer *et al.*, 05] et l'utilisation d'une décomposition temporelle sur plusieurs niveaux permet d'obtenir une telle graduabilité temporelle. En effet, il suffit de supprimer du flux les images qui ne servent pas de référence pour les images restantes.

3.3.2 Graduabilité en qualité (SNR)

Pour la graduabilité en qualité (SNR), la couche de base (qualité à laquelle la vidéo est acceptable) peut être décodée par n'importe quel décodeur H.264/AVC [Schwarz *et al.*, 04]. Les couches d'amélioration pour la qualité sont obtenues en quantifiant plus finement l'erreur obtenue entre la couche de base et la version originale de la vidéo. Les couches d'améliorations ainsi obtenues augmentent le débit par un facteur de deux. La figure 3.2 illustre cette méthode. Afin de remédier à ce problème et d'obtenir des couches qui améliorent la qualité plus finement, il a été proposé une autre méthode [Schwarz *et al.*, 05]. Au contraire de la méthode proposé précédemment, les couches de qualité sont obtenues directement dans le domaine des coefficients transformés. Pour chaque représentation d'une couche d'amélioration qui correspond à une bi-section du pas de quantification et qui est transmis dans une unité séparée du NAL, le codage des couches d'amélioration des coefficients transformés est réalisé par un processus en trois passes. Ainsi, les couches d'amélioration pour la qualité peuvent être sectionnées n'importe où dans le flux de données du NAL.

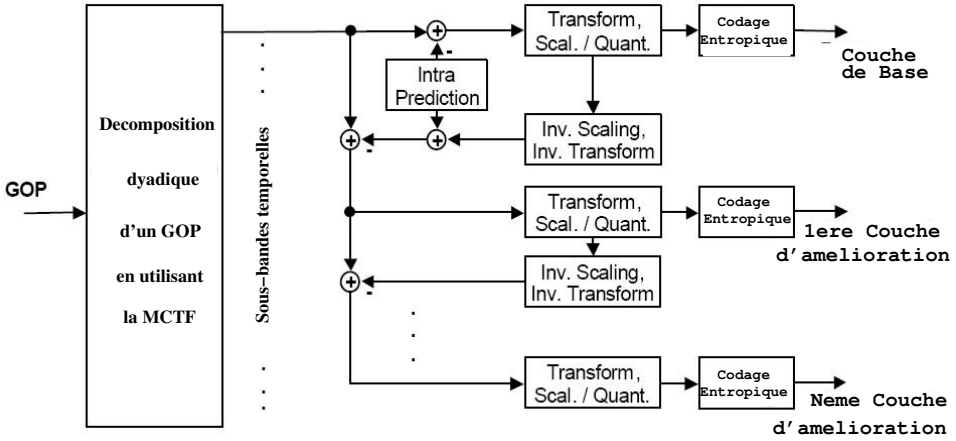


FIG. 3.2 – Concept général de l'algorithme de codage graduable pour la qualité (SNR).

3.3.3 Graduabilité spatiale

La vidéo est décomposée en plusieurs niveaux de résolutions, avec par exemple un facteur de deux dans les directions horizontale et verticale. Chaque résolution spatiale est codée indépendamment des autres. Il a été constaté que l'efficacité de codage d'une couche de résolution spatiale est affectée par la présence de couches de résolutions spatiale inférieures et que cela dépend du débit choisi et des caractéristiques de la vidéo. Il a également été observé qu'il est efficace de permettre au codeur de choisir librement les dépendances entre les couches qui peuvent être exploitées. Afin de permettre cela, différentes méthodes de prédiction ont été proposées :

- prédiction du macrobloc à partir du signal sur-échantillonné de la résolution inférieure ;
- prédiction des vecteurs de mouvement (sur-échantillonnage des vecteurs de mouvement de la résolution inférieure) ;
- prédiction de l'erreur de prédiction (sur-échantillonnage de l'erreur de prédiction obtenue à la résolution inférieure).

3.3.4 Combinaison des différentes graduabilités

Pour chaque résolution spatiale, un filtrage temporel avec compensation de mouvement est appliqué. Cette structure hiérarchique permet d'obtenir une représentation graduable temporellement de la séquence vidéo avec la possibilité de réaliser efficacement une graduabilité spatiale et en qualité (SNR). La redondance entre les différents niveaux est exploitée en utilisant différentes techniques de prédiction pour les paramètres de mouvement et les informations de texture.

Si l'on veut proposer un algorithme de codage H.264/AVC qui soit étendu à la graduabilité, il est nécessaire de respecter une certaine syntaxe dans le flux, afin que les parties non désirées au décodage soient efficacement supprimées. La syntaxe et la couche d'abstraction de réseau (NAL) proposées par le codeur H.264/AVC sont adaptées pour offrir une telle possibilité de graduabilité. Cependant, il est nécessaire, d'introduire de nouveaux bits de signalement à la couche d'abstraction de réseau, afin d'indiquer la présence de couches de raffinement de qualité (SNR) et de résolutions spatiales.

3.4 Conclusion

Dans ce chapitre, nous avons décrit l'extension à la graduabilité du codeur H.264/AVC. Grâce à l'incorporation d'un schéma MCTF au sein du codeur, il est alors possible de réaliser une graduabilité temporelle de la séquence vidéo. Celui-ci peut utiliser des filtres de Haar ou 5/3 et utilise les méthodes d'estimation et de compensation de mouvement du codeur H.264/AVC. Pour la graduabilité en qualité (SNR), quelques modifications sont apportées afin de pouvoir tronquer le flux à n'importe quel point et obtenir ainsi des couches fines d'améliorations de qualité. La graduabilité spatiale n'est pas intégrée au codeur, celle-ci est réalisée en amont, chaque résolution est ensuite codée, en commençant par la plus petite. Les informations de mouvement ainsi obtenues servent de prédiction pour les niveaux supérieurs.

La couche de base, c'est-à-dire, la plus faible résolution spatiale avec le plus faible niveau de qualité acceptable est cependant décodable par un décodeur H.264/AVC classique.

Chapitre 4

Le schéma MCTF

4.1 Introduction

Afin de comprendre le fonctionnement d'un tel schéma, il nous faut d'abord décrire précisément la technique de transformation en ondelettes et le principe du filtrage temporel. Nous pourrons alors détailler ce schéma.

4.2 La transformation en ondelettes

La DWT[1] permet une représentation d'une image sur plusieurs résolutions qui a été introduite par Mallat [Mallat, 89] (cf figure 4.1). La transformation s'appliquant sur l'image entière, elle permet une bonne représentation espace/échelle, pour un codage efficace.

FIG. 4.1 – Représentation de l'image Lena à plusieurs résolutions.

4.2.1 La transformation en ondelettes 1-D

Au codeur, la DWT 1-D [Rabbani, 02, p. 7–15] peut être appréhender comme l'application successive d'une paire de filtres passe-bas et passe-haut, suivis par des sous-échantillonneurs d'ordre deux à la sortie de chacun des filtres comme le montre la figure 4.2. La paire de filtres passe-bas et passe-haut

[1] *Discrete Wavelet Transform*

est appelée le banc de filtres d'analyse. Le filtre passe-bas préserve les basses fréquences d'un signal tout en atténuant ou éliminant les hautes fréquences, ainsi à la sortie de ce filtre, on a une version très approximée du signal original. Réciproquement, le filtre passe-haut préserve les hautes fréquences du signal telles que les bords, la texture et les détails, tout en éliminant ou atténuant les basses fréquences.

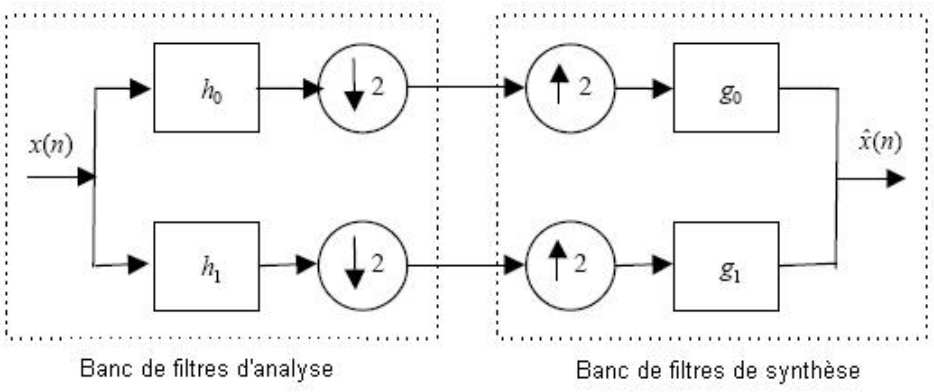


FIG. 4.2 – Bancs de filtres de synthèse et d'analyse pour une DWT 1-D.

Considérons un signal 1-D $x(n)$ et une paire de filtres passe-bas et passe-haut désignés par $h_0(n)$ et $h_1(n)$ respectivement. Un exemple de filtre passe-bas est, $h_0(n)$=(-1 2 6 2 -1)/8, qui est symétrique et a cinq coefficients entiers. Un exemple de filtre passe-haut est, $h_1(n)$ = (-1 2 -1)/2, qui est symétrique et est composé de trois coefficients entiers. Le banc de filtres d'analyse utilisé pour cet exemple a été proposé pour la première fois par Le Gall et Tabatabai [Le Gall, 88] et est souvent référencé comme le banc de filtres (5,3), indiquant un filtre passe-bas de longueur cinq et un filtre passe-haut de longueur trois. Les échantillons filtrés qui sont à la sortie de la DWT (analyse) sont appelés des coefficients d'ondelettes. À cause du sous échantillonnage critique, le nombre total de coefficients d'ondelettes est le même que le nombre d'échantillons du signal original.

Au décodeur, la reconstruction à partir des coefficients d'ondelettes est réalisée avec une autre paire de filtres passe-bas et passe-haut (g_0, g_1), connus sous le nom de banc de filtres de synthèse. En se référant à la figure 4.2, la sortie sous-échantillonnée du filtre passe-bas $h_0(n)$ est d'abord sur-échantillonnée par un facteur deux en insérant des zéros tous les deux échantillons. Le résultat est ensuite filtré avec le filtre de synthèse passe-bas, $g_0(n)$. La sortie sous-échantillonnée du filtre passe-haut $h_1(n)$ est également sur-échantillonnée et filtrée avec le filtre de synthèse passe-haut, $g_1(n)$. Les résultats sont additionnés pour produire un signal reconstruit $\hat{x}(n)$, qui si on ne quantifie pas les coefficients, sera identique au signal d'origine $x(n)$ grâce à la propriété de reconstruction parfaite.

Pour obtenir une reconstruction parfaite, les filtres d'analyse et de synthèse doivent satisfaire les

deux conditions suivantes [Vetterli, 85][Vetterli, 86] :

$$H_0(z)G_0(z) + H_1(z)G_1(z) = 2, \tag{4.1}$$

$$H_0(-z)G_0(z) + H_1(-z)G_1(z) = 0, \tag{4.2}$$

où $H_0(z)$ est la transformée en Z de $h_0(n)$ et $G_0(z)$ est la transformée en Z de $g_0(n)$, etc. La première de ces conditions permet une reconstruction exacte, alors que la seconde permet de supprimer l'aliasing. La condition de l'équation 4.2 peut être satisfaite en choisissant :

$$G_0(z) = -cz^{-l}H_1(-z) \; et \; G_1(z) = cz^{-l}H_0(-z), \tag{4.3}$$

où l est un entier constant et c est un facteur d'échelle. En combinant ce résultat avec l'équation 4.1, on indique que la paire de filtres d'analyse (h_0, h_1) doit être choisie pour satisfaire :

$$-cz^{-l}H_0(z)H_1(-z) + cz^{-l}H_1(z)H_0(-z) = 2. \tag{4.4}$$

La constante l représente un terme de délai qui impose une restriction sur l'alignement spatial des filtres d'analyse et de synthèse, tandis que la constante c affecte la normalisation du filtre. Le banc de filtres qui satisfait ces équations est connu comme le banc de filtres bi-orthogonal. Ce nom vient du fait que les fonctions h_0 et g_1 sont orthogonales l'une à l'autre ainsi que h_1 et g_0. Ensuite, on peut montrer qu'afin de satisfaire l'équation 4.4, les filtres d'analyse h_0 et h_1 ne doivent pas être de longueur égale. Si les filtres ont un nombre impair de coefficients, leur longueur ne peut différer que par un multiple de deux.

Après que le signal 1-D ait été décomposé en deux bandes, la sortie du filtre passe-bas est parfois (pour les premiers niveaux de décomposition) fortement corrélée et peut être soumise à une nouvelle étape de décomposition en deux bandes pour réaliser la décorrélation additionnelle.

4.2.2 La transformation en ondelettes 2-D

La DWT 1-D peut facilement être étendue à deux dimensions (2-D) en appliquant le banc de filtres d'une façon séparée [Le Gall, 88]. À chaque niveau de décomposition, chaque ligne d'une image 2-D est d'abord transformée en utilisant un banc de filtres d'analyse horizontal (h_0, h_1). Le même banc de filtres est ensuite appliqué verticalement sur chaque colonne des données précédemment filtrées et sous-échantillonnées. Le résultat d'une transformation en ondelettes au premier niveau se compose de quatre images filtrées et sous-échantillonnées, appelées des « sous-bandes ». Étant donné la nature linéaire du processus de filtrage, l'ordre dans lequel les filtres horizontaux et verticaux sont appliqués n'affecte pas les valeurs finales des sous-bandes 2-D. Dans une décomposition dyadique 2-D, la sous-bande de fréquence la plus basse (dénotée comme la sous-bande LL pour indiquer le filtrage passe-bas dans les deux directions) peut être ensuite décomposée en quatre sous-bandes plus petites.

La figure 4.3 montre une décomposition dyadique 2-D sur trois niveaux et les étiquettes correspondantes pour chaque sous-bande. Par exemple, l'étiquette de la sous-bande kHL indique qu'un filtre passe-haut horizontal (H) a été appliqué aux lignes, suivi par un filtre passe-bas vertical (L) appliqué aux colonnes sur le k^{ieme} niveau de décomposition de la DWT, ce qui implique que cette sous-bande contient les détails verticaux de l'image. Réciproquement, la sous-bande notée LH contient les détails horizontaux de l'image, puisqu'on a d'abord appliqué un filtre passe-bas sur les lignes, puis un filtre passe-haut sur les colonnes. Par convention, la sous-bande 0LL indique l'image originale. La figure 4.4 montre une décomposition de la DWT 2-D sur trois niveaux de l'image Lena, cela montre clairement la propriété de compactage de l'énergie de la DWT (la plupart de l'énergie se trouve dans les sous-bandes de basses fréquences).

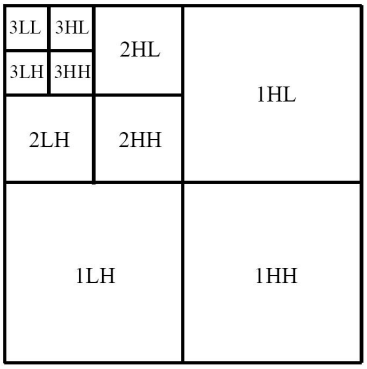


FIG. 4.3 – Décomposition ondelettes sur trois niveaux d'une image 2-D.

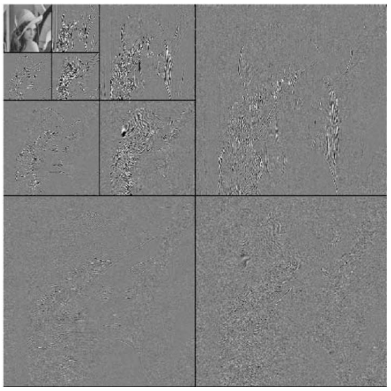


FIG. 4.4 – Décomposition ondelettes sur trois niveaux de Lena en utilisant un banc de filtres (9,7).

En principe la résolution est notée $1/2^i$, avec $i = 0, 1, .., n$, où n est le nombre de niveaux de décomposition. La résolution $1/2^0$ est la résolution originale de l'image.

On considèrera que la résolution la plus faible à laquelle l'image a été décomposée est la résolution zéro. Par exemple, en se référant à l'image 4.3, la sous-bande 3LL correspond à la résolution zéro pour une décomposition sur trois niveaux. Pour une décomposition de la DWT sur N_L niveaux, l'image

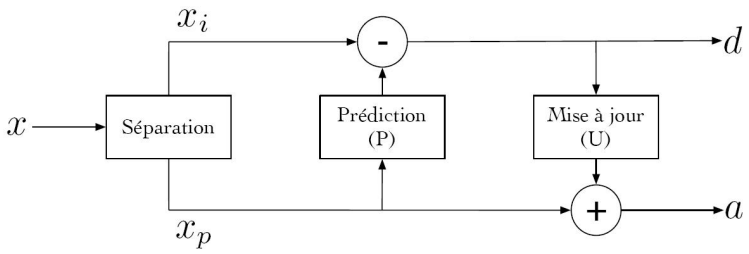


FIG. 4.5 – Schéma fonctionnel général du processus de lifting.

peut être reconstruite à $N_L + 1$ résolutions. En général, pour reconstruire une image à la résolution r ($r > 0$), les sous-bandes $(N_L - r + 1)$HL, $(N_L - r + 1)$LH et $(N_L - r + 1)$HH doivent être combinées avec l'image à la résolution $(r - 1)$. Ces sous-bandes appartiennent à la résolution r. La résolution zéro ne contient seulement que la sous bande N_LLL. Si les sous-bandes sont codées indépendamment, l'image peut être reconstruite à n'importe quel niveau de résolution en décodant simplement les portions du flux de données qui contiennent les sous-bandes correspondantes à cette résolution et aux résolutions précédentes. Par exemple, en se référant à la figure 4.3, l'image peut être reconstruite à la résolution deux en combinant la résolution un de l'image et les trois sous-bandes étiquetées 2HL, 2LH et 2HH.

4.2.3 Le schéma lifting

Par contraste avec l'utilisation limitée de la mémoire pour le calcul d'une DCT 8×8, la mise en œuvre d'une DWT 2-D requiert le stockage de l'image entière en mémoire. Une mise en œuvre alternative de la DWT a donc été proposée, connu sous le nom de schéma lifting [Daubechies, 98, Sweldens, 95, Sweldens, 97].

L'opération de lifting est constituée de plusieurs étapes. La première étape est de réaliser une transformation qui partage le signal 1-D d'origine en deux sous-signaux d'échantillons pairs et impairs (appelée *lazy wavelet transform*). Ensuite, on modifie ces valeurs en appliquant alternativement les étapes de prédiction et de mise à jour. Le schéma fonctionnel général du processus de lifting est illustré dans la figure 4.5.

Le schéma lifting possèdent donc plusieurs avantages. La transformée inverse se fait instantanément, il n'est plus nécessaire de sous-échantillonner les coefficients et il permet une reconstruction parfaite .

4.3 Le filtrage temporel

La transformée en ondelettes, grâce à son succès en codage d'image fixe, a rapidement été portée dans le domaine vidéo, en rajoutant une dimension de filtrage le long de l'axe du temps.

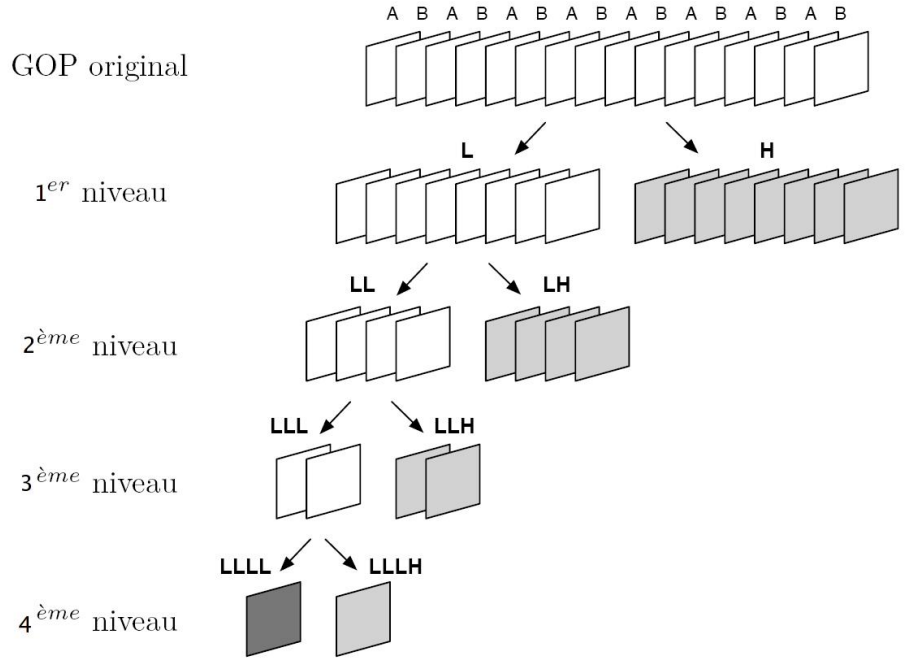


FIG. 4.6 – Décomposition sur quatre niveaux du GOP original.

4.3.1 Principe

Comme dans le domaine spatial, la transformée en ondelettes temporelle peut se faire classiquement par filtrage récursif sur les basse fréquences. À un niveau de résolution donné, un groupe de 2^p images est transformé en deux ensembles de 2^{p-1} images basses et hautes fréquences. Ce traitement peut ensuite être réitéré sur le groupe d'images basses fréquences.

Une illustration de décomposition en ondelettes temporelle sur quatre niveaux est montrée sur la figure 4.6

De ce GOP initial de 16 images, seules les images hautes fréquences et la dernière image basses fréquences (quatrième niveau) seront transmises.

Des GOP de 8, 16 ou 32 images sont généralement utilisés avec les filtres de Haar. Pour une paire d'images A et B, les équations de l'analyse temporelle par la transformée de Haar sont les suivantes :

$$\begin{aligned} L &= \tfrac{A+B}{2}, \\ H &= \tfrac{B-A}{2}. \end{aligned} \tag{4.5}$$

Cependant ce type de filtrage ne tient pas compte des informations de mouvement. Ainsi pour une séquence vidéo à fort mouvement, les images basses fréquences sont floues et les images hautes fréquences contiennent beaucoup d'énergie. Des techniques basées sur la compensation du mouvement sont alors apparues.

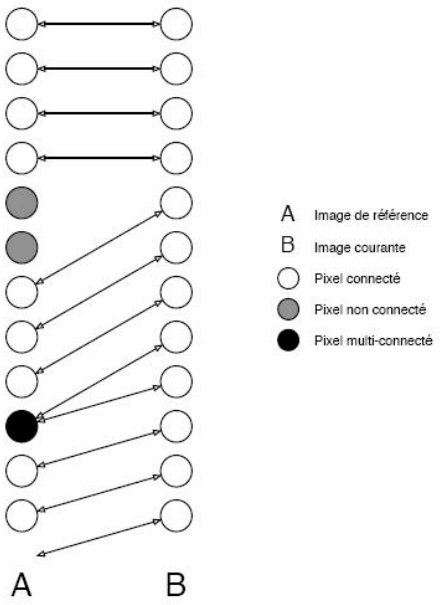


FIG. 4.7 – Prédiction du mouvement « en avant » entre deux images.

4.3.2 Filtrage temporel et compensation du mouvement

Les techniques usuelles de compensation du mouvement consistant à décrire les champs de vecteurs par blocs furent associées au filtrage temporel, donnant ainsi des schémas MCTF[2]. L'inconvénient de ces méthodes de filtrage temporel compensées en mouvement réside dans le fait qu'il existe des discontinuités aux frontières des blocs. Il en résulte des phénomènes de recouvrement/découvrement qui rendent la compensation en mouvement difficilement réversible. En général, les schémas de filtrage de type MCTF ne présentent pas de reconstruction parfaite.

Dans le cadre d'une estimation basée bloc puis en généralisant à l'utilisation de modèles arbitraires, Ohm résout le problème de la réversibilité pour des GOP de taille deux (filtrage de Haar) [Ohm, 94, Ohm *et al.*, 04].

Si l'on se place dans le cas d'une prédiction « en avant » (*forward*) par bloc de taille de 4×4 entre deux images A et B (voir figure 4.7), chaque pixel de B est alors connecté à un pixel de l'image A. Cependant, du fait du recouvrement des blocs dans l'image de référence, certains pixels de A n'ont pas de correspondants (en gris sur la figure) et d'autres en ont plusieurs (en noir). Ohm parle de pixels multi-connectés (*multiple connected pixels*) et de pixels non connectés (*unconnected pixels*).

Ohm choisit de lier arbitrairement chaque pixel multi-connecté de A avec un seul de ses candidats dans l'image B (par exemple le plus proche). Les candidats non retenus sont alors étiquetés comme des pixels découverts (*uncovered pixels*) par opposition aux autres pixels de B qui eux sont connectés (voir figure 4.7. De la même façon, les pixels non connectés de A sont par la suite désignés comme des

[2] *Motion Compensated Temporal Filtering*

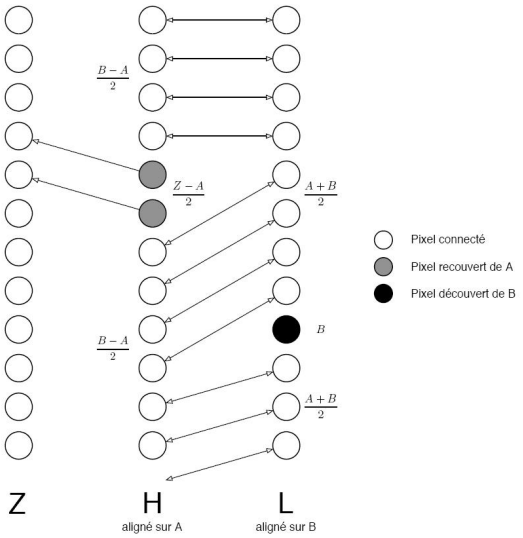


FIG. 4.8 – Traitement des zones recouvertes et découvertes par la méthode Ohm.

pixels recouverts (*covered pixels*).

On obtient ainsi des couples de pixels qui peuvent être filtrés afin d'obtenir l'image basses fréquences L alignée spatialement sur B et l'image hautes fréquences H alignée spatialement sur A. Soit $n = (x, y)$ les coordonnées des pixels de B et $v_{A \leftarrow B} = (dx, dy)$ le vecteur estimé du bloc de B vers le bloc A correspondant. Le filtrage proposé par Ohm est le suivant :

$$L(n) = \frac{A(n + v_{A \leftarrow B}) + B(n)}{2}, \tag{4.6}$$

$$H(n + v_{A \leftarrow B}) = \frac{B(n) - A(n + v_{A \leftarrow B})}{2}. \tag{4.7}$$

Pour les pixels découverts de B et recouverts de A, Ohm propose une autre méthode de filtrage car ces pixels n'ont pas de correspondants. Il propose que les pixels découverts de B soient recopiés tels quels dans l'image L et que les pixels recouverts de A soient prédits à partir de l'image précédente Z comme cela est illustré dans la figure 4.8. Le filtrage pour ces pixels particuliers est alors le suivant :

$$L(n) = B(n), \tag{4.8}$$

$$H(n) = \frac{Z(n + v_{Z \leftarrow A}) - A(n)}{2}. \tag{4.9}$$

Grâce à ce schéma, lors de la synthèse et sans transmission d'informations supplémentaires, le décodeur peut régénérer les mêmes cartes de pixels découverts/recouverts à partir des champs de mouvement décodés. La reconstruction est alors parfaite. Cependant en fonction des séquences vidéo, le nombre de pixels découverts peut augmenter rapidement, rendant l'estimation de mouvement moins performante.

4.3.3 Modélisation lifting du filtrage court (Haar)

Comme pour la transformation ondelettes 2-D, on peut également réaliser un filtrage temporel basé sur une approche de schéma lifting. Pesquet-Popescu et Bottreau [Pesquet-Popescu, 01] proposent en 2001 une formulation lifting de la décomposition temporelle de Choi et Woods [Choi, 99]. Le schéma lifting a pour caractéristique de réduire les calculs et de garantir une reconstruction parfaite. Bien que le filtrage de Haar soit de complexité faible, l'approche proposée par Pesquet-Popescu et Bottreau a permis de faire progresser le domaine de l'analyse vidéo.

Ils montrent que les compensations en mouvement prises en compte lors de la transformée temporelle peuvent s'interpréter comme des opérateurs de prédiction (*prediction*) et de mise à jour (*update*) non linéaires.

Les équations qui permettent d'obtenir les images basses et hautes fréquences pour l'approche de Choi et Woods [Choi, 99] sont les suivantes :

$$H(n) = \frac{B(n) - A(n + v_{A \leftarrow B})}{\sqrt{2}}, \tag{4.10}$$

$$L(n + v_{A \leftarrow B}) = \frac{B(n) + A(n + v_{A \leftarrow B})}{\sqrt{2}}. \tag{4.11}$$

Et pour les pixels non connectés, l'image hautes fréquences est obtenue de la même manière (équation 4.10). L'image des basses fréquences est obtenue en multipliant les valeurs des pixels de référence par le facteur $\sqrt{2}$:

$$L(n) = \sqrt{2}.A(n). \tag{4.12}$$

Dans l'approche proposée par Pesquet-Popescu et Bottreau [Pesquet-Popescu, 01], l'opérateur de prédiction correspond à un opérateur de compensation en mouvement C ($CoC' = Id$) suivi éventuellement d'un opérateur d'interpolation I (pour un mouvement sous pixélique). L'opérateur de mise à jour correspond également à une compensation en mouvement, avec les mêmes vecteurs que pour la prédiction mais avec des signes opposés, suivie également d'une interpolation. L'analyse temporelle des pixels connectés, dans leur version lifting, peut s'écrire sous la forme :

$$H(m,n) = \frac{B(m,n) - I\{C\{A(m,n)\}\}}{\sqrt{2}}, \tag{4.13}$$

$$L(p,q) = I\{C'\{H(p,q)\}\} + \sqrt{2}A(p,q). \tag{4.14}$$

Dans un schéma de lifting, ce sont les mêmes opérations de compensation et d'interpolation qui sont réalisées au codeur et décodeur, seuls certains signes changent. Les équations s'inversent et les formules pour la reconstruction des images A et B peuvent alors être écrites sous la forme :

$$A(p,q) = \frac{L(p,q) - I\{C'\{H(p,q)\}\}}{\sqrt{2}}, \tag{4.15}$$

$$B(m,n) = I\{C\{A(m,n)\}\} + \sqrt{2}H(m,n). \tag{4.16}$$

4.3.4 Extension des schémas lifting au filtrage 5/3

Une conséquence importante concernant l'analyse temporelle multi-résolution, est le choix de la longueur des filtres. En effet, les filtres longs exploitent mieux la corrélation temporelle existante entre les images successives. C'est pourquoi de nombreux chercheurs ont essayé de développer des méthodes d'analyse temporelle basée lifting en utilisant des filtres longs, tel que le filtre 5/3.

Description du filtre 5/3

Une approche basée sur un filtre 5/3 obtient de meilleurs résultats en terme de compression que le banc de filtres de Haar mono-directionnel [Tillier *et al.*, 03]. En effet, la méthode proposée par les auteurs dans cet article, repose sur l'utilisation de filtres longs, afin d'exploiter au maximum la redondance temporelle entre les images.

Soit $x_t(n)$ où t est l'index temporel et n est la variable spatiale ; h_t est l'image haute fréquence et l_t est l'image basse fréquence. Les opérations de lifting temporel sont réalisées sur les images compensées en mouvement, les images impaires x_{2t+1} sont prédites à partir des images paires x_{2t} et x_{2t+2}. Ils obtiennent ainsi deux champs de vecteurs de mouvement pour chaque t : « en avant » (*forward*) qui prédit x_{2t+1} à partir de x_{2t} et noté v^+_{2t+1} et « en arrière » (*backward*) qui prédit x_{2t+1} à partir de x_{2t+2} et noté v^-_{2t+1}. La transformation temporelle 5/3 peut s'exprimer :

$$h_t(\mathbf{n}) = x_{2t+1}(\mathbf{n}) - \alpha x_{2t}(\mathbf{n} + \mathbf{v}^+_{2t+1}(\mathbf{n})) - \beta x_{2t+2}(\mathbf{n} + \mathbf{v}^-_{2t+1}(\mathbf{n})), \tag{4.17}$$

$$l_t(\mathbf{m}) = x_{2t}(\mathbf{m}) - \gamma h_{t-1}(\mathbf{m} - \mathbf{v}^-_{2t-1}(\mathbf{p})) - \delta h_t(\mathbf{m} - \mathbf{v}^+_{2t+1}(\mathbf{q})). \tag{4.18}$$

L'équation de mise à jour 4.18 implique que $\mathbf{p}$ et $\mathbf{q}$ satisfassent, $\mathbf{p} + v^-_{2t-1}(\mathbf{p}) = \mathbf{m}$ et $\mathbf{q} + v^+_{2t+1}(\mathbf{q}) = \mathbf{m}$. Les coefficients $\alpha = \beta = 1/2$ et $\delta = \gamma = 1/4$ correspondent au filtre bi-orthogonal 5/3.

L'opérateur de prédiction décrit par l'équation 4.17 est toujours bi-directionnel, offrant ainsi une meilleure prédiction qu'un opérateur mono-directionnel, en particulier en cas de mouvements lisses. Ils ont proposé un algorithme capable d'estimer le meilleur champ de vecteurs de mouvement (« en avant » et « en arrière ») et minimisant l'énergie des sous-bandes de détails pour le filtre 5/3, cependant, lorsque l'image ne peut pas être bi-prédite (nouveau plan), ce dispositif est inutile.

L'opérateur de mise à jour décrit par l'équation 4.18 peut être bi-directionnel, mono-directionnel ou nul, en fonction des connexions des pixels. On n'applique pas le filtre passe-bas sur les pixels non connectés des deux côtés. Ces pixels créent des changements brusques dans l'image approximée, ce qui peut entraîner des artéfacts.

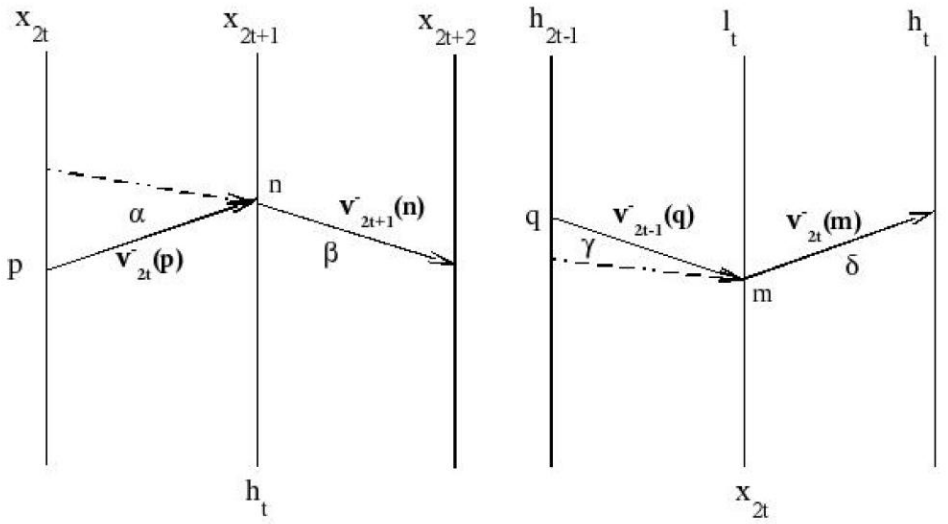
Filtre 5/3 uniforme

Afin d'éviter l'apparition des artéfacts, Tillier et Pesquet-Popescu [Pau, 04] proposent une méthode avec un schéma de prédiction alternative basée sur la forme classique d'un banc de filtres 5/3. Un banc de filtres 5/3 classique utilise pour chaque étape du lifting une recherche « en avant » et « en arrière » des

vecteurs de mouvement, alors que la méthode proposée réalise deux recherches de mouvement dans la même direction.

Un des avantages de cette nouvelle méthode est qu'elle ne produit pas de pixels non connectés (*unconnected*), améliorant considérablement la qualité visuelle des sous-bandes approximées.

Les opérateurs de prédiction et de mise à jour utilisent deux champs de vecteurs de mouvement dans la même direction, comme cela est illustré dans la figure 4.9.



(a) Opérateur de prédiction dans le domaine spatio-temporel. La flèche en pointillés indique, les connexions multiples possibles du pixel n

(b) Opérateur de mise à jour dans le domaine spatio-temporel. La flèche en pointillés indique, les connexions multiples possibles du pixel m

FIG. 4.9 – Opérateurs de prédiction et de mise à jour utilisant deux champs de vecteurs de mouvement dans la même direction.

Les nouvelles équations de lifting sont :

$$h_t(\mathbf{n}) = x_{2t+1}(\mathbf{n}) - \alpha x_{2t}(\mathbf{n} - \mathbf{v}^-_{2t}(\mathbf{p})) - \beta x_{2t+2}(\mathbf{n} + \mathbf{v}^-_{2t+1}(\mathbf{n})),$$

avec

$$\begin{cases} \alpha{=}\beta = 1/2 \ si \ \exists \ \mathbf{p} \ tq \ \mathbf{p} + \mathbf{v}^-_{2t}(\mathbf{p}) = \mathbf{n}, \\ \alpha{=}0, \ \beta = 1 \ sinon, \end{cases}$$

et

$$l_t(\mathbf{m}) = x_{2t}(\mathbf{m}) + \gamma h_{t-1}(\mathbf{m} - \mathbf{v}^-_{2t-1}(\mathbf{q})) + \delta h_t(\mathbf{m} + \mathbf{v}^-_{2t}(\mathbf{m})),$$

avec

$$\begin{cases} \gamma{=}\delta = 1/4 \ si \ \exists \ \mathbf{q} \ tq \ \mathbf{q} + \mathbf{v}^-_{2t-1}(\mathbf{q}) = \mathbf{m}, \\ \gamma{=}0, \ \delta = 1/2 \ sinon. \end{cases}$$

Un dispositif attrayant de ce nouveau schéma de lifting temporel est que chaque pixel est toujours connecté à un autre de l'image précédente.

Comparé au filtre 5/3 « classique » (*plain*), la prédiction n'est pas toujours bi-directionnelle. Cependant, l'absence de connexions dans une direction est souvent dû aux nouvelles zones découvertes, qui peuvent être prédites seulement dans une seule direction. Le pourcentage de pixels mono-connectés durant l'étape de mise à décroîtcroit à tous les niveaux de résolution temporelle.

4.3.5 Le filtrage temporel MCTF 1/3

Lorsque le filtrage temporel est correctement aligné avec les mouvements de la vidéo, l'étape de mise à jour permet de réduire le bruit et l'aliasing temporel dans les séquences vidéo avec un taux d'images réduit. Cependant, la séquence est composée de zones (recouvrements/découvrements) où l'estimateur de mouvement risque d'échouer. Dans ces régions, le filtre passe bas d'analyse temporelle va être appliqué sur de mauvaises trajectoires du mouvement. Quand cela se produit, les performances de compression baissent, mais surtout, les étapes de mises à jour font apparaître des artéfacts sur les images basses fréquences temporelles, diminuant significativement leur qualité visuelle.

Un des moyens permettant d'éviter ces potentiels artéfacts dans les images basses fréquences temporelles, est de supprimer l'étape de mise à jour dans le schéma de lifting permettant de réaliser la décomposition temporelle [Mehrseresht, 06]. Les images dites basses fréquences sont alors obtenues par un sous échantillonnage temporel de la séquence originale :

$$l_k = f_{2k}, \tag{4.19}$$

où, l_k est l'image basse fréquence temporelle et f_{2k} est l'image originale de la vidéo. La transformation temporelle est réduite à une DWT 1/3 dans lequel le filtrage temporel passe-bas est réalisé par une fonction « delta ». Les schémas de filtrage temporel avec compensation de mouvement utilisant un filtre d'ondelettes 1/3 sont souvent appelés UMCTF[3] ».

Le schéma UMCTF

Le schéma UMCTF [van der Schaar, 03, Turaga *et al.*, 05] permet un filtrage temporel adaptatif :

- nombre variable de niveaux de décomposition basé sur le contenu de la vidéo ou du niveau de complexité désiré ;
- sélection adaptative des filtres permettant différents filtrages temporels ;
- sélection adaptative des filtres, à l'intérieur et entre les niveaux de décomposition spatiale et temporelle ;

[3] *unconstrained motion-compensated temporal filtering*

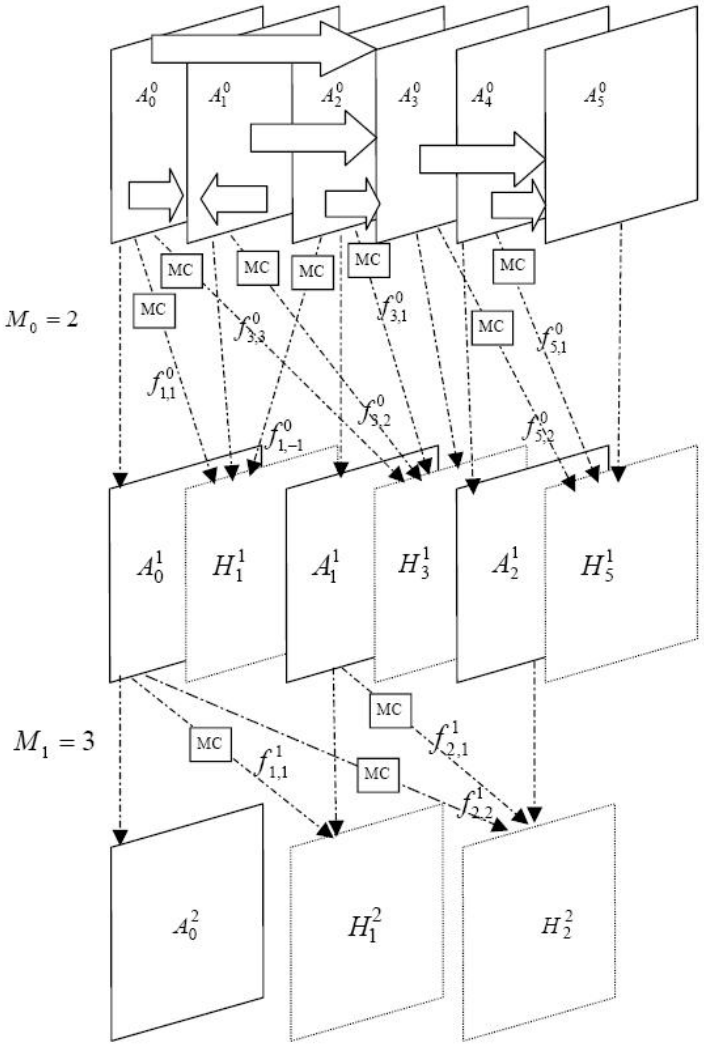


FIG. 4.10 – Schéma de décomposition temporelle pyramidale.

– nombre variable d'images H successives à l'intérieur et entre les niveaux, pour une graduabilité temporelle flexible (non dyadique) et différents perfectionnements du filtrage temporel.

Si on choisit de supprimer l'étape de mise à jour, le schéma UMCTF devient un schéma MCTF 1/3 où les images basses fréquences sont obtenues par sous-échantillonnage de la séquence vidéo originale. La figure 4.10 illustre un exemple de décomposition où l'étape de mise à jour est supprimée et le nombre d'images H varie entre les niveaux de décomposition.

Lorsque les connexions de pixels sont faibles lors de l'estimation de mouvement, le filtrage passe-bas temporel crée des artéfacts visuels ; cependant lorsque celles-ci sont bonnes, il est intéressant d'utiliser la corrélation temporelle et de calculer la moyenne temporelle des images plutôt que de créer une version sous-échantillonnée temporellement des séquences vidéos. Dans de tels cas, les images L peuvent

être créées en utilisant des filtres passe-bas « non delta ». Durant, une décomposition multi-résolution temporelle, les images deviennent de plus en plus éloignées temporellement lorsque le niveau de décomposition augmente. Ce qui a pour effet d'augmenter le nombre de mauvaises connexions pour l'estimation de mouvement. Il est nécessaire de supprimer le filtrage passe-bas pour des niveaux de décomposition temporelle élevés. Le schéma UMCTF [Turaga *et al.*, 05] permet une sélection adaptative des filtres passe-bas (delta ou non-delta) en fonction de la qualité des connexions de l'estimation du mouvement.

Le schéma UMCTF est une méthode qui permet de réaliser un filtrage efficace et flexible pour le codage vidéo à base d'ondelettes. Celui-ci est basé sur la mise en œuvre d'un schéma lifting réalisant un filtrage temporel avec un ensemble de paramètres permettant de choisir la structure de la décomposition et les filtres. Cela améliore l'efficacité du codage et la qualité des vidéos décodées. Il permet également d'améliorer la graduabilité temporelle, puisque la décomposition n'est pas forcément dyadique comme c'est le cas avec les schémas MCTF utilisant des filtres de Haar. Les images décodées pour des niveaux de décomposition élevés ne contiennent pas d'artéfacts visuels.

4.4 Conclusion

Ce chapitre a présenté premièrement la transformation ondelettes 2-D et le schéma de lifting. Par la suite, nous avons vu leur extension au domaine temporel. Le schéma MCTF permet donc de réaliser un filtrage temporel avec compensation de mouvement. On a vu que celui-ci peut être réalisé en utilisant différents types de filtres : Haar, 5/3 , 1/3, etc. Les filtres 5/3 étant plus longs, ceux-ci exploitent mieux la redondance temporelle entre les images successives d'une vidéo, mais l'étape de mise à jour peut introduire des artéfacts visuels sur les images basses fréquences. Une solution afin de remédier à ce problème est d'obtenir les images basses fréquences par un simple sous-échantillonnage temporel de la séquence originale. L'étape de mise à jour est alors supprimée, on utilise ainsi un filtre 1/3 pour le schéma MCTF.

C'est donc ce type de filtre, que nous voulons essayer de mettre en œuvre au sein de notre outil d'analyse de la vidéo. Celui-ci devant nous permettre de prendre des décisions en amont du codeur H.264/AVC pour choisir les images de référence et les modes de prédiction. Mais l'analyse seule des images hautes fréquences ne peut apporter de résultats suffisamment précis, il est nécessaire d'étudier le contenu de la vidéo. C'est pourquoi, le chapitre suivant présente les techniques de segmentation spatio-temporelle et de suivi d'objets qui ont retenu notre attention.

Chapitre 5

Objets vidéo et tubes spatio-temporels

5.1 Introduction

L'unité de base pour le codeur H.264/AVC est le macrobloc de taille 16×16 qui peut être partitionné en blocs de taille inférieure (jusqu'à 4×4). Quel que soit le contenu de la vidéo, la prédiction essaie de minimiser l'erreur pour chacun de ces macroblocs. Or pour les plans de la vidéo qui contiennent les mêmes objets, il serait judicieux de garder une certaine uniformité et cohérence dans les modes de codage choisis pour les différents objets de la vidéo. Cependant le codeur H.264/AVC en lui même ne possède aucun outil lui permettant d'obtenir des informations sur le contenu de la vidéo. En effet, si l'on souhaite réaliser un codage fonction des objets présents dans la vidéo, il est nécessaire de développer une méthode d'analyse réalisée en amont du codeur pour lui transmettre de telles informations par la suite. Cet outil doit être capable de détecter les objets et les suivre à travers les différentes images de la séquence vidéo. Il existe de nombreuses techniques de segmentation et de suivi d'objets présentes dans la littérature. Un objet vidéo étant caractérisé par sa forme, sa texture et son mouvement, les méthodes de segmentation utilisent ces critères pour les détecter et les suivre temporellement. La suite de ce chapitre présente différentes méthodes basées sur une priorité spatiale, temporelle ou une combinaison des deux. Cependant, certaines de ces méthodes n'utilisent qu'un nombre réduit d'images successives pour réaliser cette segmentation qui peut alors s'avérer inexacte, l'idéal étant de réaliser celle-ci à long terme. On parlera alors de tubes spatio-temporels, ceux-ci étant présentés dans la deuxième partie de ce chapitre.

5.2 Suivi temporel des objets dans une séquence vidéo

5.2.1 Suivi des objets par mise en correspondance

De nombreuses méthodes pour suivre les objets dans une séquence vidéo sont basées sur la mise en correspondance de régions obtenues après deux segmentations aux temps $(t-1)$ et t. Marquès et Llach [Marquès, 98] proposent une méthode de suivi des objets vidéo. La partition obtenue au temps $(t-1)$ est d'abord resegmentée en utilisant un critère basé texture afin de valider l'homogénéité spatiale des régions. Pour cela, les auteurs utilisent une technique de partage des eaux dans l'espace de couleur. Ensuite, le mouvement entre les deux images $(t-1)$ et t est estimé et les partitions obtenues au temps $(t-1)$ sont compensées et projetées sur l'image courante t. Pour finir, l'algorithme associe les partitions

obtenues au temps $(t-1)$ aux objets connus au temps t. Ce processus est réalisé en trois étapes :

- dans un premier temps, seules les régions de t recouvrant entièrement les régions de $(t-1)$ compensées sont appariées ;
- ensuite, les régions de t qui sont recouvertes à plus de 50% par une région de $t-1$ compensée sont étiquetées tout en considérant que celles-ci sont voisines et que la distance de couleur entre les deux régions est faible ;
- pour finir, toutes les régions incertaines restantes sont affectées à un des objets en utilisant la technique de partage des eaux dans l'espace couleur en prenant pour germes les régions précédemment affectées.

Il existe d'autres méthodes [Alatan *et al.*, 98, Wang *et al.*, 00] de mise en correspondance permettant de suivre les objets dans une séquence vidéo.

L'une de ces méthodes [Alatan *et al.*, 98] est un algorithme où plusieurs cartes de segmentation sont obtenues pour chaque image. La segmentation est obtenue à l'aide d'un « arbre couvrant récursif de pixels minimum » (RSST[1]) en utilisant seulement l'information de couleur. Le but étant d'obtenir des régions dont les limites correspondent avec les limites des objets réels (sémantique) présent dans la scène. Cette méthode d'arbre couvrant segmente l'image courante en plusieurs régions d'intensité uniforme. Le mouvement entre deux images consécutives est également estimé. Pour la segmentation de mouvement, ils utilisent également l'algorithme d'arbre couvrant récursif de pixels minimum pour obtenir une segmentation basée mouvement.

La segmentation basée mouvement obtenue pour l'image $(t-1)$ et l'information de mouvement estimée, leur permettent d'obtenir une prédiction de segmentation basée mouvement pour l'image courante. Pour finir, ils calculent la différence entre deux images consécutives, afin d'obtenir une segmentation basée sur ces variations. Ensuite, l'algorithme combine les différentes segmentations, pour obtenir une segmentation optimale et suivre les objets. Ils obtiennent ainsi une carte de segmentation en objets vidéo pour les temps $(t-1)$ et t. Ils réalisent ensuite la mise en correspondance des objets du temps $(t-1)$ et t ,et utilisent trois règles afin de distinguer les différents objets détectés :

- une règle qui permet de suivre les objets précédemment détectés,
- une règle détectant les objets qui commencent à se déplacer,
- une règle permettant de repérer les objets se divisant en plusieurs objets (mouvements différents).

Certains auteurs proposent d'utiliser d'autres critères que le mouvement pour suivre correctement les objets [Wang *et al.*, 00]. Ils estiment d'abord le mouvement global (mouvement de la caméra) et ensuite ils utilisent un algorithme basé sur une projection, qui est appelé la méthode des « espaces / montagnes »(« gap / mountain »). Le fond et les objets en mouvement étant assimilés respectivement à l'« espace » et aux « montagnes ». Une fois qu'ils ont obtenu la carte de segmentation pour chaque image de la séquence vidéo, ils appliquent l'algorithme de suivi des objets. L'algorithme proposé repose

[1] *Recursive Shortest Spanning Trees*

sur plusieurs règles qui combinent :

- la trajectoire des objets,
- la taille des objets,
- la distribution des niveaux de gris des objets,
- la texture des objets.

Des variables basées sur ces informations sont d'abord calculées et ensuite les résultats obtenus pour le suivi des objets sont validés ou non à partir de ces quatre variables. Dans l'image courante, chaque objet est alors associé avec quatre ensembles de variables et chaque « piste » dans l'image précédente est également associée avec quatre ensembles de variables. Ils proposent une technique permettant d'associer l'objet avec la meilleure piste existante. Il se peut qu'après ce processus, il reste des objets et des pistes qui n'ont pas pu être associés. Ces situations correspondent souvent à des objets qui commencent à se déplacer, ou qui ne sont plus en mouvement, ou alors à des recouvrements d'objets.

L'algorithme considère d'abord la possibilité de collisions entre les « pistes », en tenant compte de la dispersion (variable calculée pour suivre les objets suivant leur taille) des objets et des « pistes ». Les objets et « pistes » restants sont étiquetés comme étant des objets qui commencent à se déplacer, ou au contraire, comme stoppant leur mouvement.

5.2.2 Suivi des objets par projection initialisation

Castagno et Sodomaco [Castagno, 98] proposent une méthode de segmentation basée sur plusieurs caractéristiques de l'image :

- le mouvement,
- la luminance,
- la chrominance,
- la texture.

Leur technique de segmentation utilise un algorithme de « clustering flou » qui se réalise en deux étapes. Dans la première étape, une mesure de fiabilité *a priori* basée sur les différentes caractéristiques de l'image est calculée. Une première segmentation de l'image est ainsi obtenue. Ils calculent ensuite pour chaque région et chaque caractéristique de l'image une mesure de fiabilité *a posteriori*. Ces mesures servent ensuite d'initialisation pour l'algorithme de « clustering flou ».

Une autre méthode, présente dans la littérature [Mansouri *et al.*, 00], repose sur la minimisation de la projection de la région géodésique (forme et intenisté) du temps $(t-1)$ au temps t. L'algorithme proposé utilise plusieurs critères :

- la forme de la région au temps $(t-1)$ et au temps t,
- la luminance de la région au temps $(t-1)$ et au temps t,
- les bords de la région qui doivent correspondre à des zones de gradients forts.

Benois-Pineau et Nicolas [Benois-Pineau, 02] proposent une méthode afin de traiter les zones de recouvrements dans une séquence vidéo. Ils utilisent les informations de mouvement pour déterminer la forme de la zone de recouvrement entre deux régions. Ensuite, ils calculent la DFD[2] pour obtenir l'ordre de profondeur entre les deux régions. Cette information locale d'ordre de profondeur entre deux régions adjacentes sert par la suite à obtenir l'ordre de profondeur global pour la scène vidéo, qui est formulée comme la recherche au sein d'un graphe optimal.

5.3 Les tubes spatio-temporels

La plupart des méthodes n'utilisent que deux images successives pour réaliser la segmentation vidéo. Il est alors très difficile d'obtenir une segmentation exacte dans certains cas. En effet, les zones de recouvrements et découvrements sont difficilement assimilables à une région ou à un objet vidéo. Et dans le cas d'une segmentation temporelle, si les mouvements sont faibles entre deux images successives, on rencontre des problèmes pour distinguer les régions ou les objets.

Il est alors nécessaire de réaliser une segmentation à long terme, c'est-à-dire, qu'on ne se limite plus à seulement deux images. En effet, en utilisant un contexte temporel sur plusieurs images, on améliore ainsi la stabilité et la cohérence des résultats. Pour ces techniques basées long terme, on cherche à obtenir des tubes spatio-temporels (cf figure 5.1), c'est-à-dire, des régions ou des objets qui ont une texture et un mouvement homogènes et stables sur plusieurs images.

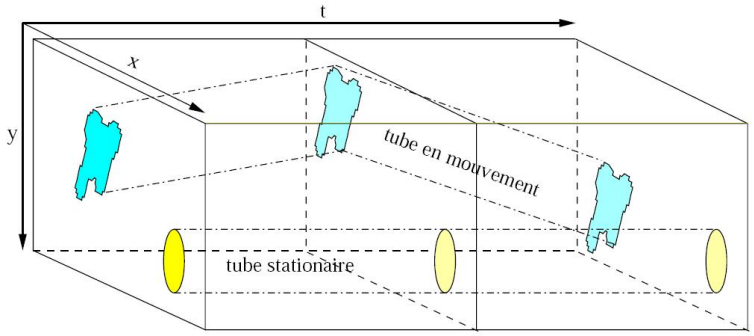


FIG. 5.1 – Illustration de la notion de tube spatio-temporel.

Les techniques de segmentation spatio-temporelle peuvent être séparées en trois catégories distinctes :

- segmentation avec priorité spatiale,
- segmentation avec priorité temporelle,
- segmentation conjointe (spatiale et temporelle).

La première de ces catégories est la plus répandue [Megret, 02]. Les techniques de segmentation avec priorité spatiale vont, dans un premier temps, segmenter indépendamment chaque image de la

[2] *Displaced Frame Difference*

vidéo. Ensuite, des régions spatio-temporelles sont formées en suivant temporellement les segments spatiaux. Cette catégorie comprend les techniques de segmentation basée mouvement (similarité des mouvements entre blocs/pixels voisins ou estimation du mouvement global), de segmentation basée sur la couleur et/ou la texture et les méthodes de suivis d'objets.

Les techniques de segmentation basée mouvement avec priorité spatiale reposent sur des informations de mouvement à court terme, généralement entre deux images successives. Afin de prendre en considération des informations à long terme, une autre catégorie de méthodes a été développée. Celles-ci utilisent des trajectoires, qui peuvent représenter les mouvements de points dans des intervalles de temps importants. L'estimation des trajectoires est réalisée dans un premier temps, en utilisant des techniques de mise en correspondance temporelle des points ou de suivi de zones texturées. Ensuite, les trajectoires correspondant au même objet en mouvement sont regroupées en utilisant une segmentation basée mouvement. Ces méthodes peuvent être à leur tour divisées en deux grandes catégories qui sont, les méthodes utilisant les similarités de mouvement et les méthodes utilisant une estimation globale du mouvement.

Contrairement aux deux premières catégories, qui donnent priorité aux regroupements spatiaux ou temporels, la dernière famille de techniques joint les méthodes de segmentation spatiale et temporelle. Elles traitent simultanément les dimensions spatiales et temporelles en considérant la vidéo comme des blocs de pixels spatio-temporels. Cette approche est plus en corrélation avec le système visuel humain qui suit conjointement les objets dans l'espace et le temps.

5.3.1 Approche par croissance de régions et fusionnement de tubes

Porikli et Wang [Porikli, 04] proposent un algorithme de segmentation automatique qui prend les avantages des techniques de segmentation basée couleur, texture, forme et mouvement. La séquence vidéo est d'abord découpée en plans, qui sont définis comme les images consécutives ayant des caractéristiques semblables entre deux changements de scènes. Pour chaque groupe d'images ainsi obtenu, un ensemble de données spatio-temporelles est calculé. Chaque pixel est en fait décrit par un vecteur de caractéristiques (couleur, différence entre images successives, texture, etc.). Ils utilisent ensuite une technique de croissance de régions à partir de marqueurs (germes) et fusions de tubes.

Leur algorithme se décompose en quatre grandes étapes. La première est le prétraitement des images. En effet, les images sont filtrées afin de supprimer le bruit et de simplifier les composantes couleur. Les couleurs sont également quantifiées en estimant un nombre de couleurs dominantes afin de simplifier davantage celles-ci et de diminuer les temps de calculs. Pour finir avec cette première étape, un vecteur de caractéristiques est calculé pour chaque pixel. On obtient ainsi un ensemble de données spatio-temporelles pour le groupe d'images sélectionnées.

La deuxième étape de l'algorithme concerne la croissance de régions à partir de marqueurs (germes).

Afin de choisir les germes, on calcule les gradients de couleurs dans chaque direction (x, y et t). Les minima locaux sont choisis comme marqueurs. Un tube est alors augmenté en regroupant les points voisins ayant les mêmes caractéristiques afin de garder une homogénéité de couleur et de texture. Pour chaque nouveau tube, un vecteur de caractéristiques est calculé et si la distance de couleur est faible, le point candidat est ajouté au tube et le vecteur de caractéristiques du tube est mis à jour.

Une fois les croissances de tubes terminées, on supprime les tubes de taille négligeable ou allongés dus à une texture fine ou à des ocntours. De tels tubes augmentent les temps de traitement de l'algorithme. Les points appartenant à ces tubes sont d'abord étiquetés comme non classés et sont ensuite joints au tube qui a la plus faible distance de couleur. Ensuite, on calcule les descripteurs de chaque tube. Les descripteurs renferment les informations sur le tube, telles que, le mouvement, la forme et les caractéristiques couleur. Pour fusionner les tubes entre eux, ils proposent plusieurs critères de similarité qui sont :

- un critère de similarité de mouvement entre tubes,
- un critère de compacité avant et après regroupement des tubes,
- un critère sur le ratio de la taille des frontières avant et après fusion des tubes,
- un critère sur le nombre d'images dans lesquelles les deux tubes sont présents,
- un critère de similarité de couleur et de texture.

Les paires de tubes ayant le meilleur score de similarité fusionnent et les descripteurs du nouveau tube ainsi obtenu sont mis à jour. Le fusionnement de tubes continue jusqu'à ce qu'il n'en reste plus que deux. La segmentation obtenue est représentée en arbre hiérarchique multi-résolution, l'utilisateur pouvant ainsi choisir les résultats de la segmentation à différents niveaux suivant le niveau de détails désiré.

5.4 Les objets vidéo

La plupart des méthodes de segmentation essayent de regrouper les régions suivant des critères de texture et de mouvement. Comme on l'a vu précédemment [Porikli, 04], on peut obtenir des tubes spatio-temporels. Cependant, la fusion de ces tubes afin d'obtenir des objets vidéos ayant un contenu plus significatif n'est pas encore très bien formalisée . Dans sa thèse, Chaumont [Chaumont, 03] propose un modèle pour formaliser la notion d'objet vidéo et une technique de segmentation adaptée.

Le modèle proposé par Chaumont suppose qu'un objet vidéo est défini par un mouvement propre. Son approche est basée sur l'utilisation d'un mouvement long terme par objet et la stabilité de la texture d'un objet sachant son mouvement. Le modèle définit un objet k par son mouvement Θ_k et sa mosaïque M_k (voir l'illustation d'une mosaïque sur la figure 5.2). Il utilise un maillage actif [Marquant *et al.*, 00] qui lui permet d'obtenir une description fine du mouvement par objet.

La segmentation a pour but de rechercher les différents objets vérifiants le modèle (le nombre d'objets K, le mouvement long terme de chacun des objets Θ_k et la texture de chacun de ces objets

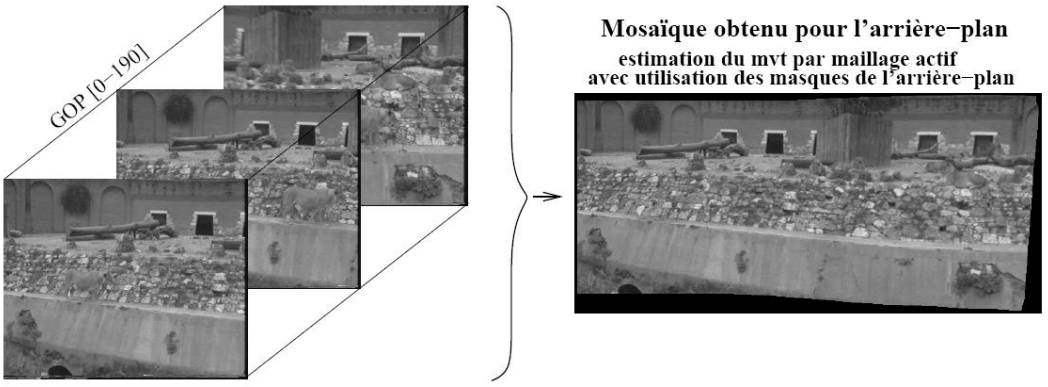


FIG. 5.2 – Mosaïque de la séquence *Lion* pour les images 0 à 190.

M_k). Le mouvement de chaque objet est mis en concurrence afin d'affecter les pixels aux objets les plus probables. Cette mise en concurrence est réalisée en deux étapes :

- une étape d'initialisation : localisation grossière des objets (germes) et estimation des mouvement des objets, Θ_k ;
- mise en concurrence des mouvements Θ_k et des textures M_k pour obtenir la segmentation finale.

Contrairement à l'approche de mise en concurrence de mouvements affines [Wang, 94], Chaumont utilise un modèle de mouvement plus complexe, obtenu par un maillage.

Il propose d'aborder le problème de segmentation comme la minimisation d'une fonctionnelle E qui prend en considération les probabilités d'appartenance des pixels aux objets. Mais cela nécessite de connaître le nombre d'objets ainsi que le mouvement de chacun d'eux. Il propose d'obtenir des germes via un algorithme de « clustering flou affine ». Ce « clustering flou affine » permet d'obtenir :

- un nombre fixé de modèles affines présents dans le groupe d'images,
- les probabilités d'appartenance des pixels à la première image.

Il obtient ainsi les germes et leurs mouvements. Il utilise ensuite une méthode de « clustering 3D » pour résoudre l'équation d'énergie. Celle-ci prend en entrée :

- les germes de chaque objet,
- le mouvement estimé de chaque germe.

Le « clustering 3D » est un algorithme itératif. On calcule d'abord la mosaïque $M_k(j)$ associée à chaque objet k en chaque pixel j ainsi que les probabilités $P_{i,k,t}$ pour chaque pixel i, objet k et temps t. Les cartes de probabilités sont mises à jour jusqu'à ce qu'elles soient stables. On obtient ainsi les probabilités d'affectation à un objet pour chaque pixel .

5.5 Conclusion

Ce chapitre décrit les grandes familles de méthode de segmentation. Dans un premier temps, nous avons vu les méthodes qui essaient de regrouper les régions/objets via des critères de similarité à

priorité spatiale, temporelle et/ou les deux. Mais ces techniques ne sont pas basées long terme, c'est-à-dire, qu'elles n'exploitent pas directement la redondance temporelle au sein des séquences vidéo. Dans la suite du chapitre, nous avons étudié les méthodes qui prennent en compte directement cet aspect temporel long terme et qui cherchent à obtenir des tubes spatio-temporels. Mais, si on essaye de fusionner ces tubes pour obtenir des objets vidéos, les résultats ne sont pas très convaincants. La dernière partie de ce chapitre a présenté un modèle d'objet vidéo basé sur un mouvement long terme et une texture mosaïque par objet. Les probabilités d'appartenance des pixels aux objets sont mis en concurrence afin de les affecter correctement.

Conclusion

Ce rapport bibliographique présente les différentes études menées au cours des six premiers mois. En effet, le premier chapitre décrit le fonctionnement du codeur H.264/AVC et permet de connaître les possibilités de codage offertes par ce nouvel algorithme. Toutes les techniques de prédiction intégrées au sein du codeur permettent de réduire le débit pour une qualité équivalente mais cela est réalisé au détriment de la complexité qui pose un réel problème dans le cas d'une architecture temps réel.

Dans le deuxième chapitre, nous présentons toutes les techniques d'optimisation du codeur H.264/AVC que nous avons rencontrées dans la littérature et qui permettent une réduction des temps de calcul pour un débit et une qualité quasi équivalents.

Le chapitre suivant présente les travaux réalisés afin d'étendre le codeur H.264/AVC à la graduabilité, que ce soit spatiale, temporelle et en qualité (SNR). Le filtrage temporel utilise la prédiction du codeur H.264/AVC pour obtenir les images hautes fréquences. Pour les images basses fréquences, une légère modification est cependant nécessaire. Les couches de qualité sont obtenues par un codage en trois passes qui permet de pouvoir tronquer le flux de données et d'obtenir des couches d'amélioration très fines. Le filtrage spatial est cependant réalisé en amont de l'architecture, la plus faible résolution spatiale avec un niveau de qualité acceptable est la couche de base de la vidéo graduable et peut être lue par un décodeur H.264/AVC classique. Les informations de codage obtenues pour les résolutions spatiales inférieures servent de prédiction pour les niveaux supérieurs.

Le filtrage temporel avec compensation de mouvement présenté dans le quatrième chapitre est le point de départ pour l'outil de pré-traitement que nous devons réaliser. Celui-ci devant nous permettre d'identifier les changements dans la séquence vidéo et de définir une stratégie de codage. Nous avons présenté plusieurs types de filtres (Haar, 5/3 et 1/3), notre choix se portant sur le schéma 1/3. En effet, le schéma 1/3 ne contenant pas d'étape de mise à jour, on peut obtenir directement les images hautes fréquences des derniers niveaux temporels sans aucun calcul au préalable. Les images basses fréquences sont quant à elles obtenues par un simple sous-échantillonnage de la séquence vidéo originale et ne contiennent pas d'artéfacts visuels.

Le dernier chapitre décrit les méthodes de segmentation et de suivi d'objets. Celles-ci étant généralement réalisées sur deux images successives, elles exploîtent peu la corrélation temporelle de la vidéo et les résultats obtenus ne sont pas toujours exacts. Si l'on veut obtenir le « cycle de vie » d'un objet au sein d'une vidéo, il est nécessaire de réaliser une étude à long terme. On obtient alors par exemple, des tubes spatio-temporels qu'il faut ensuite fusionner pour décrire le mouvement des objets au cours de la

vidéo. L'étude des objets présents dans la vidéo doit nous permettre d'analyser au mieux les résultats obtenus via le filtrage temporel et de proposer le meilleur mode de codage pour les objets de la vidéo (image référence, continuité dans le mode de prédiction choisi entre les image successives).

Les prochains travaux à réaliser seront dans un premier temps, de définir des algorithmes efficaces pour le filtrage et la pré-analyse de flux vidéo haute définition en vue de leur encodage H.264 correspondant à la tâche 4.2. Par la suite, débutera la tâche 4.3, où il s'agira de mettre au point les algorithmes associés aux techniques de filtrage et de pré-analyse précédemment étudiés. Finalement, les derniers travaux à réaliser, seront une phase de test des algorithmes sur prototypes et sur la plate-forme d'accueil du projet et constitueront la tâche 4.4.

Glossaire

2-D LOGS	2-D logaritmic search
AVC	Advanced Video Coding
BBGDS	Block based gradient decent search
BPN	Back-Propagation neural network
CABAC	Context-Based Arithmetic Coding
CAVLC	Context-Adaptive Variable-Length Codes
CIF	Common Intermediate Format
DCT	Discrete Cosine Transform
DFD	Displaced Frame Difference
DPB	Decoded Picture Buffer
DWT	Discrete Wavelet Transform
FSS	Four step search
GOP	Group of Pictures
HDTV	High Definition Television
HEXBS	Hexagon-based Search
JVT	Joint Video Team
LUT	Look Up Table
MCTF	Motion Compensated Temporal Filtering
MPEG	Moving Picture Expert Group
MSE	Mean Square Error
NAL	Network Abstraction Layer
PBR	Possible Best Reference
POC	Picture Order Count
PSNR	Peak Signal to Noise Ratio
QCIF	Quarter Common Intermediate Format
RSST	Recursive Shortest Spanning Trees

Suite page suivante ...

suite de la page précédente

SAD	Sum of Absolute Differences
SDTV	Standard Definition Television
SSIM	Structured Similarity
TCD	Transformée en Cosinus Discret
TNT	Télévision Numérique Terrestre
TSS	three step search
UMCTF	Unconstrained Motion Compensated Temporal Filtering
UMHexagonS	Hybrid Unsymmetrical-cross Multi-Hexagon-grid Search
VLC	Variable Length Coding
VCEG	Video Coding Experts Group

Bibliographie

[Alatan *et al.*, 98] Alatan, A. A., Ounral, L., Wolborn, M., Mech, R., Tuncel, E. et Sikora, T. (1998). Image Sequence Analysis for Emerging Interactive Mutlimedia Services – The European COST 211 Framework. 8(7):802 – 813.

[Benois-Pineau, 02] Benois-Pineau, J. et Nicolas, H. (2002). A New Method for Region-Based Depth Ordering in a Video Sequence : Application to Frame Interpolation. *Journal of Visual Communication and Image Representation*, 13(3):363 – 385.

[Bjøntegaard, 02] Bjøntegaard, G. et Lillevold, K. (2002). Context-adaptive vlc coding of coefficients. Rapport technique, JVT document JVT-C028, Fairfax, USA.

[Bu *et al.*, 06] Bu, J., Lou, S., Chen, C. et Zhu, J. (2006). A Predictive Block-Size Mode Selection for Inter Frame in H.264. Toulouse, France. IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP'2006.

[Castagno, 98] Castagno, R. et Sodomaco, A. (1998). Estimation of Image Feature Reliability for an Interactive Videop Segmentation Scheme. volume 1, pages 938 – 942. IEEE International Conference on Image Processing, ICIP'1998. Chicago, Illinois, USA.

[CfE, 03] CfE (2003). *Call for Evidence on Scalable Video Coding Advances.* ISO/IEC JTC1/SC29/WG11 MPEG2003/N5559, Pattaya, Thaïlande.

[CfP, 03] CfP (2003). *Call for Proposals on Scalable Video Coding Advances.* ISO/IEC JTC1/SC29/WG11 MPEG2003/N6193, Waikoloa, Hawaii.

[Chaumont, 03] Chaumont, M. (2003). *Représentation en objets vidéo pour un codage vidéo progressif et concurrentiel des séquences d'images.* Thèse de doctorat, Mathématiques, Informatique, Signal, Electronique et Télécommunications (MATISSE).

[Chen *et al.*, 05] Chen, C., Mo, L., Bu, J., Lou, S. et Yang, Z. (2005). A Novel Fast Predictive Mode Decision Algorithm For H.264. Sydney, Australie. IEEE International Symposium on Signal Processing and Its Applications, ISSPA'05.

[Chen *et al.*, 04] Chen, M.-J., Chiang, Y.-Y., Li, H.-J. et Chi, M.-C. (2004). Efficient Multi-Frame Motion estimation Algorithms for MPEG-4 AVC/JVT/H.264. Proceedings of the 2004 International Symposium on Circuits and Systems.

[Chen *et al.*, 03a] Chen, Z., Xu, J. et He, Y. (2003a). Simplifications on fast motion estimation. 8ème meeting : Genèves, Suisse. Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG (ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q.6) JVT-Hxxx.

[Chen *et al.*, 02] Chen, Z., Zhou, P. et He, Y. (2002). Fast Integer Pel and Fractional Pel Motion Estimation for JVT. 6ème meeting : Awaji, Island, JP. Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG (ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q.6) JVT-F017.

[Chen *et al.*, 03b] Chen, Z., Zhou, P. et He, Y. (2003b). Fast Motion Estimation for JVT. 7ème meeting : Pattaya II, Thailande. Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG (ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q.6) JVT-G016.

[Choi *et al.*, 05] Choi, B.-D., Hwang, M.-C., Cho, J.-K., Kim, J.-S., Kim, J.-H. et Ko, S.-J. (2005). Realtime H.264 Encoding System using Fast Motion Estimation and Mode Decision. *Lecture Notes in Computer Science*, 3824:174 – 183.

[Choi, 99] Choi, S.-J. et Woods, J. W. (1999). Motion-Compensated 3-D Subband coding of video. 8(2):155 – 167.

[Daubechies, 98] Daubechies, I. et Sweldens, W. (1998). Factoring Wavelet Transforms into Lifting Steps. *J. Fourier Anal. Appl.*, 4(3):245 – 267.

[Flierl *et al.*, 03] Flierl, M., Girod, B. N. et Taubman, D. (2003). Generalized B Pictures and the Draft H.264/AVC Video Compression Standard. 13:587 – 597.

[Girod, 00] Girod, B. (2000). Efficiency analysis of multihypothesis motion-compensated prediction for video coding. volume 9, pages 173 – 183. IEEE Transactions on Image Processing.

[Golomb, 66] Golomb, S. W. (1966). Run Length Coding. IT-12:399 – 401.

[Huang *et al.*, 06] Huang, W.-B., Lin, Y.-L., Cheng, H.-W., Su, A. W. et Kuo, Y.-H. (2006). Two-Stage Mode Selection of H.264/AVC Video Encoding with Rate Distortion Optimization. Toulouse, France. IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP'2006.

[Huang *et al.*, 03] Huang, Y.-W., Hsieh, B.-Y., Wang, T.-C., Chien, S.-Y., Ma, S.-Y., Shen, C.-F. et Chen, L.-G. (2003). Analysis and reduction of reference frames for motion estimation in MPEG-4/AVC/JVT/H.264. volume 3, pages 145 – 148, Hong Kong. IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP'2003.

[ISO/IEC, 03] ISO/IEC (2003). ISO/IEC 14496-10 et ITU-T Rec. H.264, 2003. Advanced Video Coding.

[Jain, 81] Jain, J. et Jain, A. (1981). Displacement measurement and its application in interframe image coding. *IEEE Transactions on Communications*, COM-29:1799 – 1806.

[Koga *et al.*, 81] Koga, T., Iinuma, K., Hirano, A., Lijima, Y. et Ishiguro, T. (1981). Motion compensated interframe coding for video conferencing. pages G5.3.1 – G5.3.5, New Orleans, LA. In Proceedings Nat. Telecommunications Conf. 81.

[Le Gall, 88] Le Gall, D. et Tabatabai, A. (1988). Subband coding of digital images using symmetric kernel filters and arithmetic coding techniques. *Proceedings of the International Conference on Acoustics, Speech Signal Processing*, pages 761 – 764. New York, USA.

[Li *et al.*, 06] Li, L., Li, S., Ishiwata, S., Matsui, M., Ikenaga, T. et Goto, S. (2006). Multiple Reference Frame Motion Estimation with Fast Reference Search and Decision for H.264/AVC. Atlanta, GA, USA. IEEE International Conference on Image Processing, ICIP'2006. Review.

[Li *et al.*, 04] Li, X., Li, E. Q. et Chen, Y.-K. (2004). Fast Multi-Frame Motion Estimation Algorithm With Adaptive search Strategies In H.264. volume 3, pages 369 – 372, Montreal, Quebec, Canada. IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP'2004.

[List *et al.*, 03] List, P., Joch, A., Lainema, J., Bjøntegaard, G. et Karczewicz, M. (2003). Adaptive deblocking filter. *IEEE Transactions on Circuits and Systems for Video Technology*, 13:614 – 619.

[Liu, 96] Liu, L. et Feig, E. (1996). A block-based gradient descent search algorithm for block motion estimation in video coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 6(4):419 – 422.

[Liu *et al.*, 06] Liu, Z., Song, Y., Ikenaga, T. et Goto, S. (2006). Low-Pass Filter Based VLSI Oriented Variable Block Size Motion Estimation Algorithm for H.264. Toulouse, France. IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP'2006.

[Ma, 03] Ma, K.-K. et Qiu, G. (2003). Unequal-Arm Adaptive Rood Pattern Search for Fast Block-Matching Motion Estimation in the JVT/H.26L. Barcelone, Espagne. IEEE International Conference on Image Processing, ICIP'2003.

[Mai *et al.*, 06] Mai, Z.-Y., Yang, C.-L., Kuang, K.-Z. et Po, L.-M. (2006). A Novel Motion Estimation Method Based on Structural Similarity for H.264 Inter Prediction. Toulouse, France. IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP'2006.

[Mallat, 89] Mallat, S. G. (1989). A theory for multiresolution signal decomposition - The wavelet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11:674–693.

[Malvar *et al.*, 03] Malvar, H., Hallapuro, A., Karczewicz, M. et Kerofsky, L. (2003). Low-Complexity transform and quantization in H.264/AVC. *IEEE Transactions on Circuits and Systems for Video Technology*, 13:598 – 603.

[Mansouri *et al.*, 00] Mansouri, A.-R., Olivier, A. et Konrad, J. (2000). Topology-Independent Region Tracking with Level Sets. Vancouver, Canada. IEEE International Conference on Image Processing, ICIP'2000.

[Marpe *et al.*, 01] Marpe, D., Blättermann, G. et Wiegand, T. (2001). Adaptive codes for h.26l.

[Marpe *et al.*, 03] Marpe, D., Schwarz, H. et Wiegand, T. (2003). Context-Based Adaptive Binary Arithmetic Coding in the H.264/AVC Video Compression Standard. 13(7):620 – 636. Publication Award of ITG.

[Marquant *et al.*, 00] Marquant, G., Pateux, S. et Labit, C. (2000). Mesh and "crack lines" : Application to object-based motion estimation and higher scalability. *In IEEE International Conference on Image Processing ICIP 2000*, volume 2, pages 554–557, Vancouver, BC, Canada.

[Marquès, 98] Marquès, F. et Llach, J. (1998). Tracking of generic objects for video object generation. Chicago, USA. IEEE International Conference on Image Processing, ICIP'1998.

[Megret, 02] Megret, R. et DeMenthon, D. (2002). A Survey of Spatio-Temporal Grouping Techniques. Language And Media Processing (LAMP), Submitted to Computer Vision and Image Understanding (CVIU), an archival journal published by Academic Press.

[Mehrseresht, 06] Mehrseresht, N. et Taubman, D. (2006). An Efficient Content-Adaptive Motion-Compensated 3-D DWT With Enhanced Spatial and Temporal Scalability. 15(6):1397 – 1412.

[Ohm, 94] Ohm, J.-R. (1994). Three-Dimensional Subband Coding with Motion Compensation. 3:559 – 589.

[Ohm *et al.*, 04] Ohm, J.-R., van der Schaar, M. et Woods, J. W. (2004). Interframe Wavelet Coding : Motion Picture Representation for Universal Scalability.

[Ozbek, 05] Ozbek, N. et Tekalp, A. M. (2005). Fast H.264/AVC Video Encoding with Multiple Frame References. pages 597 – 600, Gênes, Italie. IEEE International Conference on Image Processing, ICIP'2005.

[Pau, 04] Pau, G. et Pesquet-Popescu, B. (2004). Uniform Motion-Compensated 5/3 Filterbank For Subband Video Coding. Singapore, Republic of Singapore. IEEE International Conference on Image Processing, ICIP'2004.

[Pesquet-Popescu, 01] Pesquet-Popescu, B. et Bottreau, V. (2001). Three-Dimensional Lifting Schemes for Motion Compensated Video Compression. volume 3, pages 1793 – 1796, Salt Lake City, UT, USA. IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'2001.

[Po, 96] Po, L. M. et Ma, W. C. (1996). A novel four-step search algorithm for fast block motion estimation. *IEEE Transactions on Circuits and Systems for Video Technology*, 6(3):313 – 317.

[Porikli, 04] Porikli, F. et Wang, Y. (2004). Automatic Video Object Segmentation Using Volume Growing and Hierarchical Clustering. 3:442 – 453.

[Rabbani, 02] Rabbani, M. et Joshi, R. (2002). An overview of the JPEG2000 still image compression standard. *Signal Processing : Image Communication*, 17:3 – 48.

[Reichel *et al.*, 05] Reichel, J., Schwarz, H. et Wien, M. (2005). *Scalable Video Coding - Working Draft 1.* Hong Kong, CN.

[Richardson, 03] Richardson, I. E. G. (2003). *H.264 and MPEG-4 video compression : Video Coding for Next-Generation Multimedia.* Chippenham.

[Schäfer *et al.*, 05] Schäfer, R., Schwarz, H., Marpe, D., Schierl, T. et Wiegand, T. (2005). MCTF and Scalability Extension of H.264/AVC and its Application to Video Transmission, STorage and Surveillance. Bejing, Chine.

[Schwarz *et al.*, 05] Schwarz, H., Marpe, D., Schierl, T. et Wiegand, T. (2005). Combined scalability support for the scalable extension of H.264/AVC. Amsterdam, Pays-Bas.

[Schwarz *et al.*, 04] Schwarz, H., Marpe, D. et Wiegand, T. (2004). MCTF and Scalability Extension of H.264/AVC. San Francisco, CA, USA.

[Su *et al.*, 06] Su, R., Liu, G. et Zhang, T. (2006). Fast Mode Decision Algorithm for Intra Prediction in H.264/AVC. Toulouse, France. IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP'2006.

[Su, 04] Su, Y. et Sun, M.-T. (2004). Fast Multiple Reference Frame Motion Estimation for H.264. *In Proceedings of the 2004 IEEE International Conference on Multimedia and Expo, ICME 2004*, pages 695–698, Taipei, Taiwan.

[Sweldens, 95] Sweldens, W. (1995). The lifting scheme : a new philosophy in biorthogonal wavelet constructions. *Proc. SPIE 2569*, pages 68 – 79.

[Sweldens, 97] Sweldens, W. (1997). The lifting scheme : a construction of second generation wavelets. *Siam J. Math. Anal.*, 29(2):511 – 546.

[Tillier *et al.*, 03] Tillier, C., Pesquet-Popescu, B., Zhang, Y. et Heijmans, H. (2003). Scalable Video Compression with Temporal Lifting Using 5/3 Filters. pages 55 – 58, St Malo France. Proceedings of the Picture Coding Symposium.

[Turaga *et al.*, 05] Turaga, D. S., van der Schaar, M., Andreopoulos, Y., Munteanu, A. et Schelkens, P. (2005). Unconstrained Motion Compensated Temporal Filtreing (UMCTF) for Efficient and Flexible Interframe Wavelet Video Coding. *Signal Processing : Image Communication*, 20(1):1 – 19.

[van der Schaar, 03] van der Schaar, M. et Turaga, D. S. (2003). Unconstrained Motion Compensated Temporal Filtering (UMCTF) Framework for Wavelet Video Coding. pages 81 – 84, Santa Barbara, CA, USA.

[Vetterli, 85] Vetterli, M. (1985). Splitting a signal into subsampled channels allowing perfect reconstruction. *In Proceedings of IASTED Conference on Applied Signal Processing and Digital Filtering*, Paris.

[Vetterli, 86] Vetterli, M. (1986). Filter banks allowing perfect reconstruction. *Signal Processing*, 10(3):219–244.

[Wang, 94] Wang, J. Y. A. et Adelson, E. H. (1994). Representing Moving Images with Layers. volume 3, pages 625 – 638. IEEE Transactions on Image Processing.

[Wang *et al.*, 00] Wang, Y., Doherty, J. F. et Van Dyck, R. E. (2000). Moving Object Tracking in Video. Washington DC, USA. Proc. IEEE Applied Imagery Pattern Recognition Workshop.

[Wang *et al.*, 04] Wang, Z., Bovick, A. C., Sheikh, H. R. et Simoncelli, E. P. (2004). Image quality assesment : from error visibility to structural similarity. volume 13, pages 600 – 612. IEEE Transactions on Image Processing.

[Wedi, 03] Wedi, T. (2003). Motion Compensation in H.264/AVC. *IEEE Transactions on Circuits and Systems for Video Technology*, 13:577 – 586.

[Wiegand *et al.*, 03] Wiegand, T., Sullivan, G., Bjontegaard, G. et Luthra, A. (2003). Overview of the H.264/AVC Video Coding Standard. 13(7):560 – 576.

[Wiegand *et al.*, 99] Wiegand, T., Zhang, X. et Girod, B. (1999). Long-Term Memory Motion-Compensated Prediction. volume 9, pages 70 – 84. IEEE Transactions on Circuits and Systems for Video Technology.

[Xu *et al.*, 03] Xu, J., Yang, P. et He, Y. (2003). Modification of Fast Motion Estimation. 10ème meeting : Waikoloa, HI, USA. Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG (ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q.6) JVT-J027.

[Yu, 04a] Yu, A. C. (2004a). Efficient Block-Size Selection Algorithm for Inter-Frame Coding in H.264/MPEG-4 AVC. volume 3, pages 169 – 172, Montreal, Quebec, Canada. IEEE International Conference on Acoustic, Speech, and Signal Processing, ICASSP'2004.

[Yu, 04b] Yu, A. C. et Martin, G. R. (2004b). Advanced Block Size Selection Algorithm for Inter-Frame Coding in H.264/AVC. pages 95 – 98, Singapore, Republic of Singapore. IEEE International Conference on Image Processing, ICIP'2004.

[Yu *et al.*, 05] Yu, A. C., Martin, G. R. et Park, H. (2005). Improved Schemes for Inter-Frame Coding in the H.264/AVC Standard. volume 2, pages 902 – 905. IEEE International Conference on Image Processing, ICIP'2005.

[Yuan *et al.*, 04] Yuan, Y., Feng, D. et Zhong, Y.-Z. (2004). Three Fast Methods for Adaptive Key-frame Setting and Dynamic Frame-rate Adjusting in Video Coding. volume 1 de *ISSN :1304-4508*. International Journal of Computational Intelligence.

[Zhu *et al.*, 02] Zhu, C., Lin, X. et Chau, L.-P. (2002). Hexagon-Based Search Pattern for Fast Block Motion Estimation. *IEEE Transactions on Circuits and Systems for Video Technology*, 12(5):349 – 355.