Optimisation du filtrage temporel

Tout au long de ce chapitre, nous présentons différentes stratégies d'optimisation de la transformée temporelle mise en jeu dans le schéma de codage vidéo $t+2D$. Nous visons tout d'abord l'amélioration de l'efficacité objective du codeur vidéo. En se basant sur le filtre temporel $5/3$ décrit dans le chapitre précédent, nous présentons tout d'abord dans la section 4.1 un algorithme quasi-optimal de choix des champs de mouvement mis en jeu dans cette transformée. La mise en place de cet algorithme conduit alors à des résultats expérimentaux qui montrent un gain significatif par rapport à la stratégie adoptée dans la section précédente, validant ainsi l'approche retenue.

L'amélioration subjective de la qualité de codage est aussi une priorité dans la construction d'une transformée temporelle. En particulier, nous rapportons dans la section 4.2 la présence d'artefacts fantômes dans les séquences vidéo décodées à bas débit, rappelant des zones ou objets précédemment observés. La présence de ces artefacts est fortuite et est mal traduite par une mesure objective de la qualité comme le PSNR. Après avoir décrit et analysé les raisons de la présence de ces artefacts, nous présentons alors un nouveau filtre temporel, basé sur la transformée $5/3$ et construit dans l'optique de ne pas générer de tels artefacts. Les résultats expérimentaux observés après la mise en place de cette transformée sont visuellement convaincants. Nous montrons ensuite comment l'algorithme précédent de choix optimal des champs de mouvement peut être appliqué dans le cas de cette transformée. Nous observons alors un gain supplémentaire de l'efficacité de codage objective en terme de PSNR.

Les transformées temporelles classiquement utilisées dans les schémas $t+2D$ possèdent un inconvénient qui n'a pas encore été mentionné : elles introduisent un retard non négligeable à l'encodage et au décodage des séquences visuelles. Cette latence est souvent trop importante pour permettre leur utilisation dans des applications en temps réel comme la vidéoconférence ou la vidéosurveillance. Nous présentons dans la section 4.3 une étude détaillée sur les retards introduits par différents filtres temporels et sur leurs causes. Nous proposons alors une stratégie de modification générale de la transformée temporelle, permettant de modérer voire d'annuler les retards qu'elle introduit et conduisant seulement à des pertes minimes en terme de débit-distorsion. Un exemple est donné dans le cas de la transformée $5/3$ et est illustré par des simulations expérimentales. Les résultats sont convaincants et concluent une solution offrant un large éventail de compromis entre délai et efficacité de codage, en fonction des besoins de l'application.

Nous avons pour l'instant seulement envisagé des transformées temporelles issues ou dérivées de l'ondelette de Haar et de l'ondelette $5/3$, de supports relativement courts. Au vu de l'amélioration constatée au chapitre précédent lors du passage du filtre de Haar au filtre $5/3$, on peut s'interroger sur le bénéfice apporté par un filtre basé sur une autre ondelette à support plus long. La construction d'une transformée temporelle avec une prédiction à plus long terme laisse ainsi entrevoir une meilleure décorrélation temporelle

des images. A cette fin, nous présentons dans la section 4.4 un filtre temporel basé sur l'ondelette de Daubechies-4. Ses performances ne sont cependant pas à la hauteur de son originalité et après avoir montré quelques résultats expérimentaux, nous expliquons les raisons de ses performances modestes.

4.1 Optimisation des vecteurs impliqués dans la prédiction

Les sous-bandes temporelles de détail constituent la majeure partie du flux binaire et une façon simple d'améliorer l'efficacité globale du codeur vidéo est de diminuer la complexité de ces images. On cherche ici à améliorer l'opérateur de prédiction mis en jeu dans la transformée temporelle 5/3 en optimisant les champs de vecteurs utilisés. La stratégie proposée dans cette section a conduit à la publication d'un article de conférence [105], repris dans un article de revue [106] plus général sur l'utilisation du schéma lifting compensé en mouvement en codage vidéo scalable.

4.1.1 Présentation du problème

On rappelle la transformée temporelle 5/3 utilisée par le schéma de codage présenté dans la section 3.1.3 qui s'exprime sous la forme lifting suivante :

$$h_t^0(\boldsymbol{n}) = x_{2t+1}(\boldsymbol{n}) - \frac{1}{2}\big(\mathcal{C}(x_{2t}, \boldsymbol{v}_{2t+1}^+)(\boldsymbol{n}) + \mathcal{C}(x_{2t+2}, \boldsymbol{v}_{2t+1}^-)(\boldsymbol{n})\big) \tag{4.1}$$

$$l_t^0(\boldsymbol{n}) = x_{2t}(\boldsymbol{n}) + \gamma\,\mathcal{C}^{-1}(h_{t-1}, \boldsymbol{v}_{2t-1}^-)(\boldsymbol{n}) + \delta\,\mathcal{C}^{-1}(h_t, \boldsymbol{v}_{2t+1}^+)(\boldsymbol{n}) \tag{4.2}$$

$$h_t(\boldsymbol{n}) = 1/\sqrt{2}\; h_t^0(\boldsymbol{n}) \tag{4.3}$$

$$l_t(\boldsymbol{n}) = \sqrt{2}\; l_t^0(\boldsymbol{n}) \tag{4.4}$$

$$\text{avec} \begin{cases} \gamma = \delta = 1/4 & \text{si } \boldsymbol{n} \text{ est connecté des deux côtés} \\ \gamma = 1/2 \text{ et } \delta = 0 & \text{si } \boldsymbol{n} \text{ est connecté seulement à gauche} \\ \gamma = 0 \text{ et } \delta = 1/2 & \text{si } \boldsymbol{n} \text{ est connecté seulement à droite} \\ \gamma = 0 \text{ et } \delta = 0 & \text{si } \boldsymbol{n} \text{ n'est pas connecté} \end{cases}$$

Nous nous intéressons ici uniquement à l'optimisation de la prédiction dans le but de diminuer la complexité des images de détail h_t. Pour simplifier les notations, nous omettons le coefficient de normalisation $1/\sqrt{2}$ dans nos raisonnements. En développant l'opérateur de compensation de mouvement $\mathcal{C}$, on peut alors réécrire l'équation (4.1) et détailler les coefficients des sous-bandes de détail :

$$h_t(\boldsymbol{n}) = x_{2t+1}(\boldsymbol{n}) - \frac{1}{2}\Big(x_{2t}\big(\boldsymbol{n} - \boldsymbol{v}_{2t+1}^+(\boldsymbol{n})\big) + x_{2t+2}\big(\boldsymbol{n} - \boldsymbol{v}_{2t+1}^-(\boldsymbol{n})\big)\Big) \tag{4.5}$$

On considère ici le problème de l'estimation bidirectionnelle des champs de mouvement avant $\boldsymbol{v}_{2t+1}^+$ et arrière $\boldsymbol{v}_{2t+1}^-$ impliqués dans la prédiction, de manière à minimiser la distorsion des images de détail h_t. Sous la réserve d'un choix judicieux d'une mesure de distorsion, on espère ainsi minimiser le coût de codage des images h_t. Il est par exemple raisonnable de penser que le choix de la norme ℓ_2 et donc la minimisation de l'énergie des images de détail h_t conduise à une réduction de leur coût de codage. On notera que cette approche a été poursuivie ultérieurement par Cagnazzo [27] dans un cas plus simple.

Compte tenu de la structure en blocs des champs de mouvement $\boldsymbol{v}_{2t+1}^+$ et $\boldsymbol{v}_{2t+1}^-$ due à la méthode choisie pour leur estimation, on choisit de minimiser la distorsion des images

de détail h_t, bloc par bloc. En nous concentrant sur la minimisation d'un bloc $\mathcal{B}$ courant appartenant à l'image x_{2t+1}, nous choisissons d'omettre l'indice spatial $\boldsymbol{n}$ pour alléger les notations et écrivons alors $\boldsymbol{v}^+ = \boldsymbol{v}^+_{2t+1}(\boldsymbol{n})$ et $\boldsymbol{v}^- = \boldsymbol{v}^-_{2t+1}(\boldsymbol{n})$. La minimisation de la distorsion des images de détail h_t revient ainsi à un problème de recherche d'optimum à deux paramètres. Elle peut se faire sous une contrainte de débit liée au coût des vecteurs $\lambda\big(R(\boldsymbol{v}^+) + R(\boldsymbol{v}^-)\big)$ ou non (en prenant $\lambda = 0$), et conduit à la minimisation du critère J général suivant :

$$J(\boldsymbol{v}^+, \boldsymbol{v}^-) = \sum_{\boldsymbol{n}\in\mathcal{B}} d\big(h_t(\boldsymbol{n})\big) + \lambda R(\boldsymbol{v}^+) + \lambda R(\boldsymbol{v}^-) \tag{4.6}$$

où $\mathcal{B}$ est un bloc de l'image courante x_{2t+1} à prédire, d une mesure de distorsion usuelle (erreur absolue ℓ_1, norme quadratique ℓ_2 , etc...) et R le coût de codage d'un vecteur. En minimisant la distorsion de tous les blocs des images de détail h_t comme illustré sur la Fig. 4.1, on espère ainsi minimiser leur complexité et donc faciliter leur codage.

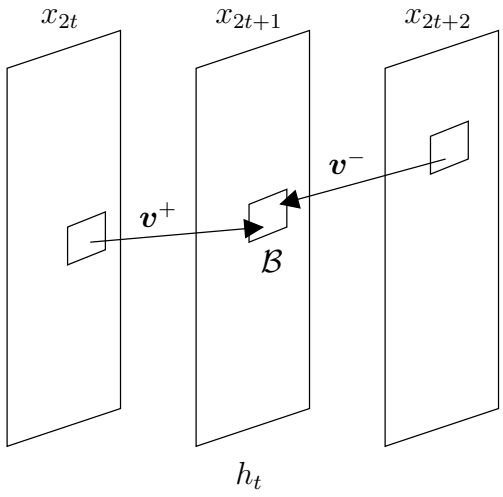


FIG. 4.1 – Opérateur de prédiction mis en jeu dans la transformée 5/3 et minimisation de la distorsion du bloc $\mathcal{B}$.

Estimation indépendante de $\boldsymbol{v}^+$ et $\boldsymbol{v}^-$

Dans la transformée temporelle 5/3 proposée dans la section 3.1 du chapitre précédent, les champs de mouvement avant $\boldsymbol{v}^+$ et arrière $\boldsymbol{v}^-$ sont estimés de façon indépendante. Le champ de mouvement avant $\boldsymbol{v}^+$ est ainsi calculé par la procédure d'appariement de blocs (*block-matching*) HVSBM, décrite en section 2.2.4, où chaque bloc de l'image courante x_{2t+1} est mis en correspondance avec un bloc de l'image de référence x_{2t}, de façon à minimiser le coût $D + \lambda R$ où D est une mesure de distorsion usuelle : la SAD (*Sum of Absolute Differences*). Le champ de mouvement arrière $\boldsymbol{v}^-$ est calculé de la même façon et indépendamment de $\boldsymbol{v}^+$ mais en prenant x_{2t+2} comme image de référence. Les vecteurs mouvements $\boldsymbol{v}^+$ et $\boldsymbol{v}^-$ vérifient donc :

$$\boldsymbol{v}^+ = \arg\min_{\boldsymbol{v}} \sum_{\boldsymbol{n}\in\mathcal{B}} \Big[d\big(x_{2t+1}(\boldsymbol{n}) - x_{2t}(\boldsymbol{n} - \boldsymbol{v}(\boldsymbol{n}))\big)\Big] + \lambda R(\boldsymbol{v}) \tag{4.7}$$

$$\boldsymbol{v}^- = \arg\min_{\boldsymbol{v}} \sum_{\boldsymbol{n}\in\mathcal{B}} \Big[d\big(x_{2t+1}(\boldsymbol{n}) - x_{2t+2}(\boldsymbol{n} - \boldsymbol{v}(\boldsymbol{n}))\big)\Big] + \lambda R(\boldsymbol{v}) \tag{4.8}$$

Pour la simple raison que les vecteurs $\boldsymbol{v}^+$ et $\boldsymbol{v}^-$ ont été estimés de manière indépendante, ils n'ont aucune raison à priori de minimiser le critère J précédemment défini et ne peuvent donc minimiser l'énergie du bloc $\mathcal{B}$ de l'image de détail h_t.

On souhaite ainsi estimer conjointement ce couple de vecteurs de façon à minimiser le critère J. Cependant, la minimisation directe de ce problème d'optimisation à deux paramètres est difficile et sa complexité quadratique est largement prohibitive. Nous proposons dans la section suivante une solution quasi-optimale, permettant de trouver un couple de vecteurs $\boldsymbol{v}^+$ et $\boldsymbol{v}^-$ constituant un minimum local de J .

4.1.2 Prédiction itérative bidirectionnelle jointe

La minimisation de J peut se faire par une suite de minimisations alternées du champ avant $\boldsymbol{v}^+$ et du champ arrière $\boldsymbol{v}^-$, en prenant compte des champs précédemment estimés. Nous avons ainsi présenté un algorithme itératif [105], capable de minimiser J et convergeant vers un minimum local. Un des intérêts de cet algorithme réside dans le fait qu'il ne nécessite pas la construction d'un nouvel estimateur de mouvements et repose sur un opérateur d'appariement de blocs quelconque que nous notons BM. Pour un bloc B courant et une image de référence x donnée, BM est défini comme un opérateur capable de fournir un vecteur $\boldsymbol{v}$=BM($\mathcal{B}$,x), pointant vers un bloc de l'image de référence et minimisant le coût $D + \lambda R$ associé au bloc $\mathcal{B}$.

Notre algorithme itératif de prédiction bidirectionnelle jointe permet de trouver les vecteurs $\boldsymbol{v}^+$ et $\boldsymbol{v}^-$ optimaux au sens de J et donc de minimiser le coût du bloc $\mathcal{B}$. Il est dit itératif car il repose sur la construction d'une suite de couples de vecteurs $\{\boldsymbol{v}_i^+, \boldsymbol{v}_i^-\}_{i\in\mathbb{N}}$, conduisant à la convergence du critère $\{J(\boldsymbol{v}_i^+, \boldsymbol{v}_i^-)\}_{i\in\mathbb{N}}$ vers un minimum local. L'algorithme s'énonce de la façon suivante :

Initialisation

Le vecteur avant $\boldsymbol{v}_0^+$ est obtenu par un appariement de blocs classique entre le bloc $\mathcal{B}$ de l'image courante x_{2t+1} et l'image de référence x_{2t}. On a alors $\boldsymbol{v}_0^+ = \text{BM}(\mathcal{B}, x_{2t})$.

Itération i, pour $i \geq 1$

- Le vecteur arrière $\boldsymbol{v}_i^-$ est obtenu par une procédure d'appariement de blocs entre un bloc virtuel $\mathcal{B}'$ et l'image de référence $x_{2t+2}/2$. Le bloc virtuel $\mathcal{B}'$ dépend du vecteur $\boldsymbol{v}_{i-1}^+$ précédent et est défini par $\mathcal{B}'(\boldsymbol{n}) = x_{2t+1}(\boldsymbol{n}) - x_{2t}\big(\boldsymbol{n} - \boldsymbol{v}_{i-1}^+(\boldsymbol{n})\big)/2$. Cette technique revient à faire une sorte d'appariement de blocs semi-compensé en mouvement. On a alors $\boldsymbol{v}_i^- = \text{BM}(\mathcal{B}', x_{2t+2}/2)$.
- De façon similaire, le vecteur avant $\boldsymbol{v}_i^+$ est obtenu par appariement de blocs entre entre le bloc virtuel $\mathcal{B}''$ semi-compensé en mouvement défini par $\mathcal{B}''(\boldsymbol{n}) = x_{2t+1}(\boldsymbol{n}) - x_{2t+2}\big(\boldsymbol{n} - \boldsymbol{v}_i^-(\boldsymbol{n})\big)/2$ et l'image $x_{2t}/2$. On a alors $\boldsymbol{v}_i^+ = \text{BM}(\mathcal{B}'', x_{2t}/2)$.

L'initialisation de l'algorithme revient ainsi à estimer d'abord le vecteur avant $\boldsymbol{v}_0^+$ par un appariement de blocs classique. Lors de la première itération, le vecteur arrière $\boldsymbol{v}_1^-$ est alors estimé grâce à la connaissance de $\boldsymbol{v}_0^+$. Ensuite, et à chaque itération, les vecteurs avant $\boldsymbol{v}_i^+$ et arrière $\boldsymbol{v}_i^-$ sont réestimés et permettent la convergence rapide du critère J vers un minimum local. On minimise ainsi le coût du bloc $\mathcal{B}$ au sens du critère J.

Chaque itération possède une complexité équivalente à deux recherches de blocs, sans compter l'initialisation. La complexité globale de l'algorithme avec n itérations est donc équivalente à $2n + 1$ procédures de recherche de blocs. Cependant, la complexité globale peut être réduite en diminuant à chaque itération le domaine de recherche des blocs. Ceci est justifié par le fait qu'il est probable que la direction du vecteur $\boldsymbol{v}_i^+$ soit proche de celle de $\boldsymbol{v}_{i-1}^+$ et que l'on peut ainsi réestimer $\boldsymbol{v}_i^+$ sur un domaine réduit. Comme dit précédemment, cet algorithme ne repose que sur l'utilisation d'une procédure d'appariement de blocs générique BM, rendant son implémentation grandement simplifiée. On remarquera enfin qu'il est possible de stopper l'algorithme au cours d'une itération, en s'arrêtant après l'estimation du vecteur arrière $\boldsymbol{v}_i^-$; on parlera alors de demi-itération.

Nous nous proposons désormais de montrer les propriétés de convergence de cet algorithme et de montrer qu'il est nécessairement meilleur qu'une stratégie consistant à faire une estimation indépendante des champs de mouvement. En utilisant les propriétés de l'estimateur de mouvement choisi, il est possible de montrer lors de l'initialisation que :

$$\boldsymbol{v}_0^+ = \arg\min_{\boldsymbol{v}} \sum_{\boldsymbol{n}\in\mathcal{B}} d\Big[x_{2t+1}(\boldsymbol{n}) - x_{2t}(\boldsymbol{n}-\boldsymbol{v})\Big] + \lambda R(\boldsymbol{v}) \tag{4.9}$$

Il est de même possible de montrer qu'à chaque itération, on vérifie :

$$\boldsymbol{v}_i^- = \arg\min_{\boldsymbol{v}} \sum_{\boldsymbol{n}\in\mathcal{B}} d\Big[x_{2t+1}(\boldsymbol{n}) - \frac{x_{2t}(\boldsymbol{n}-\boldsymbol{v}_{i-1}^+) + x_{2t+2}(\boldsymbol{n}-\boldsymbol{v})}{2}\Big] + \lambda R(\boldsymbol{v})$$

$$\boldsymbol{v}_i^+ = \arg\min_{\boldsymbol{v}} \sum_{\boldsymbol{n}\in\mathcal{B}} d\Big[x_{2t+1}(\boldsymbol{n}) - \frac{x_{2t}(\boldsymbol{n}-\boldsymbol{v}) + x_{2t+2}(\boldsymbol{n}-\boldsymbol{v}_i^-)}{2}\Big] + \lambda R(\boldsymbol{v})$$

A l'itération i, le critère $J(\boldsymbol{v}_i^+, \boldsymbol{v}_i^-)$ vaut donc :

$$J(\boldsymbol{v}_i^+, \boldsymbol{v}_i^-) = \sum_{\boldsymbol{n}\in\mathcal{B}} d\Big[x_{2t+1}(\boldsymbol{n}) - \frac{x_{2t}(\boldsymbol{n}-\boldsymbol{v}_i^+) + x_{2t+2}(\boldsymbol{n}-\boldsymbol{v}_i^-)}{2}\Big] + \lambda\big(R(\boldsymbol{v}_i^+) + R(\boldsymbol{v}_i^-)\big) \tag{4.10}$$

La poursuite de l'algorithme et la réalisation d'une demi-itération suivante nous permet d'obtenir le vecteur arrière suivant $\boldsymbol{v}_{i+1}^-$, qui vérifie :

$$\boldsymbol{v}_{i+1}^- = \arg\min_{\boldsymbol{v}} \sum_{\boldsymbol{n}\in\mathcal{B}} d\Big[x_{2t+1}(\boldsymbol{n}) - \frac{x_{2t}(\boldsymbol{n}-\boldsymbol{v}_i^+) + x_{2t+2}(\boldsymbol{n}-\boldsymbol{v})}{2}\Big] + \lambda R(\boldsymbol{v}) \tag{4.11}$$

Le vecteur $\boldsymbol{v}_{i+1}^-$ est donc le résultat d'une minimisation d'un terme de distorsion du critère $J(\boldsymbol{v}_i^+, \boldsymbol{v})$. Sous l'hypothèse que la direction du vecteur $\boldsymbol{v}_{i+1}^-$ soit proche de $\boldsymbol{v}_i^-$, il est légitime de supposer que leurs coûts sont très proches voire égaux $R(\boldsymbol{v}_{i+1}^-) \simeq R(\boldsymbol{v}_i^-)$. En combinant les équations (4.10) et (4.11), il est alors possible de montrer que $\boldsymbol{v}_{i+1}^-$ constitue un choix nécessairement meilleur que $\boldsymbol{v}_i^-$ au sens du critère J et de montrer ainsi que :

$$J(\boldsymbol{v}_i^+, \boldsymbol{v}_{i+1}^-) \leq J(\boldsymbol{v}_i^+, \boldsymbol{v}_i^-) \tag{4.12}$$

De la même façon, on peut montrer que $J(\boldsymbol{v}_{i+1}^+, \boldsymbol{v}_{i+1}^-) \leq J(\boldsymbol{v}_i^+, \boldsymbol{v}_i^-)$ et prouver ainsi que la suite $\{J(\boldsymbol{v}_i^+, \boldsymbol{v}_i^-)\}_{i\in\mathbb{N}}$ est décroissante, bornée et donc convergente. De plus, il est possible de montrer que les champs de mouvement estimés avec cet algorithme et avec seulement une demi-itération sont toujours meilleurs au sens du critère J que des champs estimés indépendamment. Ceci revient à montrer que $J(\boldsymbol{v}_0^+, \boldsymbol{v}_1^-) \leq J(\boldsymbol{v}_*^+, \boldsymbol{v}_*^-)$, en

notant $\boldsymbol{v}_*^+$ et $\boldsymbol{v}_*^-$ les vecteurs obtenus de façon indépendante, comme spécifié dans la section 4.1.1. Cette relation est obtenue au moyen des équations (4.8) et (4.10), de l'inégalité triangulaire et en remarquant que $\boldsymbol{v}_*^+ = \boldsymbol{v}_0^+$.

La poursuite d'une seule demi-itération nous fournit alors un algorithme d'estimation de mouvement bidirectionnel conjoint efficace, conduisant à une prédiction théoriquement toujours meilleure qu'une approche où les vecteurs sont estimés de manière indépendante. De plus, les deux approches ont une complexité identique et équivalente à deux recherches globales de mouvement, renforçant d'autant plus l'intérêt de cet algorithme d'estimation de mouvement bidirectionnel conjoint.

Il est à noter que cet algorithme n'est pas spécifique à une taille de blocs fixe car il fait appel à une procédure d'appariement de blocs BM générique. Il peut ainsi s'adapter simplement à d'autres procédures d'appariement à taille de blocs variable, comme l'algorithme HVSBM décrit en section 2.2.4 et utilisé dans notre schéma de codage.

On remarquera enfin que des travaux similaires ont été proposés indépendamment dans [162] dans le cadre d'un codeur hybride vidéo MPEG-2. Cependant, la méthode retenue par les auteurs possède une complexité plus élevée que notre algorithme car elle nécessite une initialisation indépendante des champs $\boldsymbol{v}_0^+$ *et* $\boldsymbol{v}_0^-$.

4.1.3 Prédiction bidirectionnelle à vecteur de mouvement unique

La transformée temporelle de Haar est mono-directionnelle mais ne nécessite le codage que d'un seul champ de mouvement. Au contraire, la transformée $5/3$ est bidirectionnelle et utilise deux champs de mouvement, lui permettant ainsi d'effectuer une prédiction temporelle de meilleure qualité. Cependant, cet avantage a un coût car il nécessite le codage d'un champ de mouvement supplémentaire. Nous avons pu ainsi observer dans la section 3.2 du chapitre précédent qu'à bas débit, la transformée de Haar possède une efficacité de codage supérieure à la transformée $5/3$, pénalisée par le surcoût de codage engendré par son deuxième champ de mouvement.

Nous souhaitons construire une transformée temporelle capable de concilier une prédiction bidirectionnelle tout en n'utilisant qu'un *seul* champ de mouvement. En faisant l'hypothèse d'un mouvement apparent souple et uniforme entre trois images consécutives, il est possible de construire une telle transformée en se basant sur le filtre temporel $5/3$. L'idée réside dans l'utilisation d'un champ de mouvement arrière obtenu par une simple opposition de signe du champ avant, conduisant ainsi à une transformée dont la prédiction s'écrit :

$$h_t(\boldsymbol{n}) = x_{2t+1}(\boldsymbol{n}) - \frac{1}{2}\Big(x_{2t}\big(\boldsymbol{n} - \boldsymbol{v}_{2t+1}(\boldsymbol{n})\big) + x_{2t+2}\big(\boldsymbol{n} + \boldsymbol{v}_{2t+1}(\boldsymbol{n})\big)\Big) \tag{4.13}$$

L'estimation du champ de mouvement $\boldsymbol{v}_{2t+1}$ minimisant l'énergie d'un bloc $\mathcal{B}$ de h_t et donc la minimisation du critère J devient ici un problème mono-dimensionnel plus simple à résoudre. La construction d'un nouvel estimateur de mouvement est cependant nécessaire.

Des travaux similaires ont été poursuivis dans [156] où les auteurs construisent une transformée bidirectionnelle à un seul champ de mouvement en utilisant un estimateur de mouvement minimisant la somme des erreurs quadratiques avant et arrière.

4.1.4 Résultats expérimentaux

Prédiction itérative et décroissance de la distorsion

Dans le contexte de l'implémentation du codeur vidéo décrit dans le chapitre précédent, nous souhaitons tout d'abord vérifier expérimentalement les propriétés de l'algorithme itératif d'estimation jointe. Celui-ci a été implémenté au sein de la transformée 5/3 présentée dans la section 3.1.3 pour optimiser le choix des champs de mouvement mis en jeu dans la prédiction. Nous avons alors étudié les sous-bandes temporelles issues de la décomposition sur 4 niveaux des séquences vidéo *Stefan* et *Mobile*, sans codage spatial ni quantification. Plusieurs simulations ont été effectuées en faisant varier le nombre d'itérations de l'algorithme de prédiction bidirectionnelle et en le comparant avec l'approche classique où les champs de mouvement sont estimés de façon indépendante. Les tableaux Tab. 4.1, 4.2, 4.3 et 4.4 montrent les résultats obtenus en présentant la norme ℓ_1 et l'énergie moyenne (norme ℓ_2) observées sur les sous-bandes temporelles de détail, calculées sur la composante Y à différents niveaux temporels.

Norme ℓ_1	Indépendante	0.5 it	1 it	1.5 it	2 it
Niveau 1	4.70	4.21	4.04	4.03	4.01
Niveau 2	7.78	6.99	6.72	6.71	6.67
Niveau 3	11.94	10.65	10.22	10.24	10.16
Niveau 4	17.46	15.60	15.01	15.03	14.92

TAB. 4.1 – Norme ℓ_1 des images de détail de la décomposition temporelle de la séquence *Stefan* CIF 30 Hz obtenue en utilisant une estimation bidirectionnelle indépendante et en utilisant l'algorithme itératif proposé.

Énergie moyenne	Indépendante	0.5 it	1 it	1.5 it	2 it
Niveau 1	79.33	62.67	57.04	56.95	56.22
Niveau 2	194.54	154.21	141.57	141.59	139.76
Niveau 3	433.38	339.09	310.11	312.98	308.29
Niveau 4	893.81	705.19	649.74	653.49	642.21

TAB. 4.2 – Énergie moyenne des images de détail de la décomposition temporelle de la séquence *Stefan* CIF 30 Hz obtenue en utilisant une estimation bidirectionnelle indépendante et en utilisant l'algorithme itératif proposé.

Norme ℓ_1	Indépendante	0.5 it	1 it
Level 1	2.76	2.49	2.39
Level 2	5.18	4.66	4.47
Level 3	8.87	8.17	7.89
Level 4	14.66	13.97	13.48

TAB. 4.3 – Norme ℓ_1 des images de détail de la décomposition temporelle de la séquence *Mobile* CIF 30 Hz obtenue en utilisant une estimation bidirectionnelle indépendante et en utilisant l'algorithme itératif proposé.

Énergie moyenne	Indépendante	0.5 it	1 it
Level 1	31.80	23.32	20.81
Level 2	98.65	75.32	68.46
Level 3	257.48	209.46	193.39
Level 4	652.75	572.28	529.50

TAB. 4.4 – Énergie moyenne des images de détail de la décomposition temporelle de la séquence *Mobile* CIF 30 Hz obtenue en utilisant une estimation bidirectionnelle indépendante et en utilisant l'algorithme itératif proposé.

Conformément à nos attentes, on remarque que l'algorithme proposé avec une demi-itération conduit à des énergies inférieures de près de 20 % par rapport à celles obtenues avec une estimation indépendante, pour une complexité équivalente. De plus, on observe une décroissance nette des normes ℓ_1 et ℓ_2 des trames de détail à chaque itération, atteignant jusqu'à 35% après 2 itérations. Ces observations sont en accord avec les propriétés théoriques de décroissance (4.12) énoncées à la fin de la section 4.1.2.

Efficacité de codage avec le codec MC-EZBC

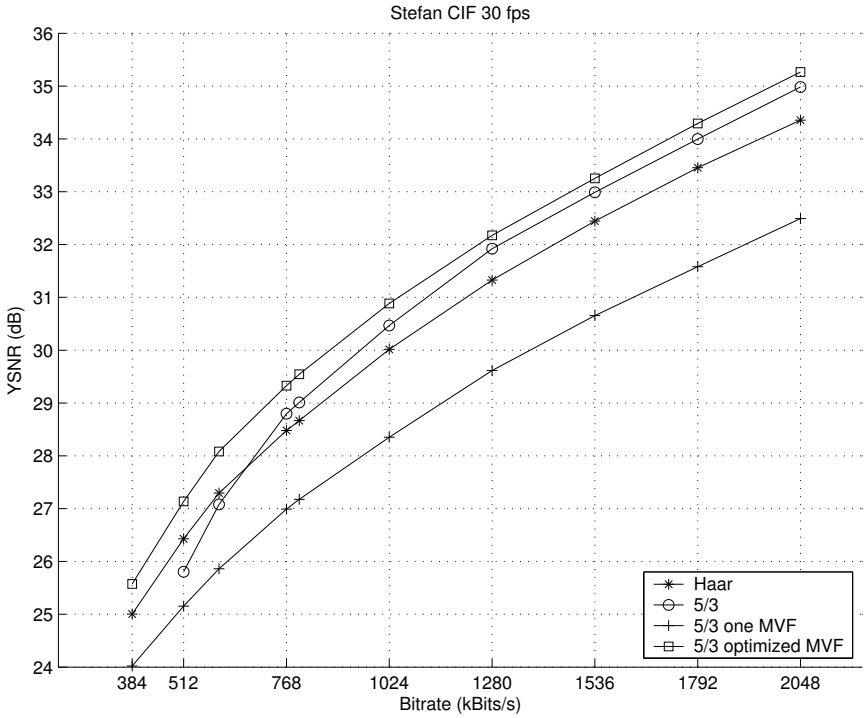
Afin d'évaluer le gain qu'apportent les deux méthodes de prédiction présentées, nous avons réalisé des simulations complètes de codage vidéo en utilisant le schéma de codage original présenté dans le chapitre précédent. Nous avons considéré les séquences vidéo couleur *Stefan*, *Foreman*, *Mobile* et *Tempête* au format CIF 30 Hz, choisies pour la variété de mouvements et de textures qu'elles offrent. Les séquences ont été décomposées sur 4 niveaux temporels et les champs de mouvement ont été estimés au 1/8ème de pixel près.

Les séquences vidéos ont été encodées entièrement, signifiant que le bitstream contient les composantes de luminance Y et de chrominances U et V de chaque image, les champs de mouvements et les informations d'en-tête. L'efficacité de codage est exprimée en terme de YSNR ou Y-PSNR, défini comme la moyenne des PSNR de la composante Y des images décodées. Les Fig. 4.2, 4.3, 4.4 et 4.5 présentent les résultats de codage obtenus en comparant les transformées temporelles suivantes :

- Transformée de Haar
- Transformée 5/3
- Transformée 5/3 à vecteur de mouvement unique, notée *5/3 one MVF*
- Transformée 5/3 avec prédiction bidirectionnelle jointe, notée *5/3 optimized MVF*

Les simulations utilisant la transformée 5/3 avec prédiction bidirectionnelle jointe des champs de mouvement ont été réalisées avec une itération, en utilisant comme mesure de distorsion d la norme ℓ_1 ou SAD. Cette transformée a donc une complexité équivalente à 3 étapes de recherche de mouvements. Par comparaison, les transformées de Haar et 5/3 à vecteur de mouvement unique possèdent une complexité équivalente à une seule étape de recherche. Pour sa part, la transformée 5/3 classique possède une complexité équivalente à 2 procédures de recherche de mouvement.

Comparons tout d'abord la transformée de Haar et la transformée 5/3. Comme précédemment, nous remarquons que la transformée 5/3 est bien plus efficace que celle de Haar à moyen et haut débits, où elle surpasse cette dernière d'environ 1 dB. Ceci peut être expliqué par une meilleure prédiction temporelle due à l'estimation bidirectionnelle du mouvement mais aussi par une mise à jour bidirectionnelle durant le calcul de la sous-

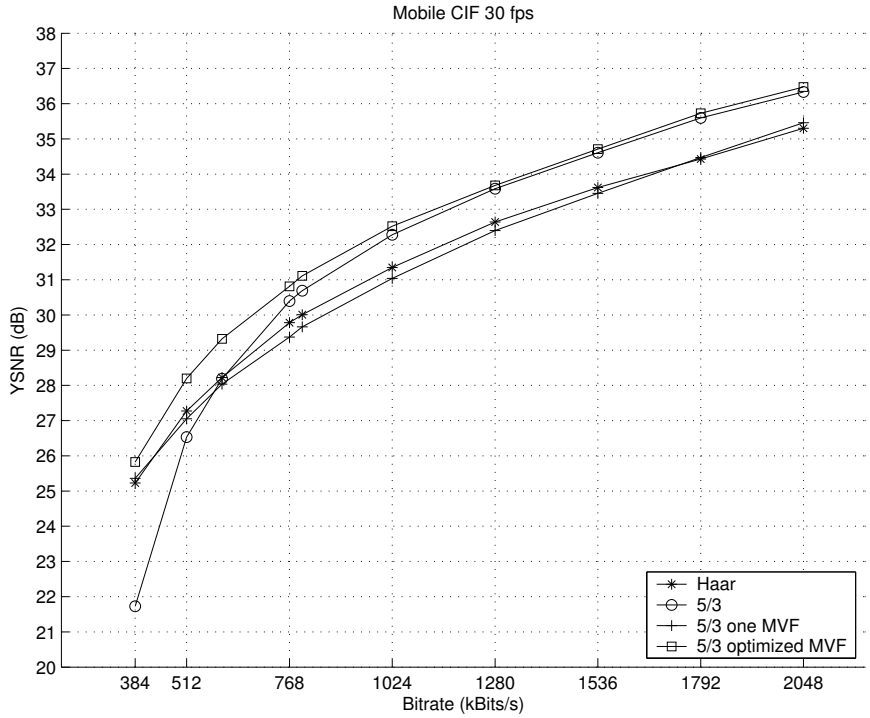


YSNR (in dB)	512 kbs	768 kbs	1024 kbs	1536 kbs	2048 kbs
Haar	26.42	28.47	30.01	32.44	34.35
5/3	25.80	28.79	30.46	32.98	34.98
5/3 one MVF	25.15	26.99	28.35	30.65	32.49
5/3 optimized MVF	27.13	29.32	30.88	33.25	35.26

FIG. 4.2 – Courbes et tableaux de débit-distorsion obtenus pour différents filtres temporels et différents débits sur la séquence *Stefan* CIF 30 Hz.

bande d'approximation. Cependant, la différence de gain est moindre sur des séquences comme *Foreman* où les mouvements complexes rotatoires de la tête du personnage réduisent les bénéfices apportés par les opérateurs bidirectionnels. Cependant, la transformée de Haar est plus efficace à faible débit car elle ne nécessite qu'un seul champ de mouvement là où la transformée 5/3 en nécessite deux. Comme ces champs sont codés sans perte, ils sont incompressibles et fixent ainsi une limite au débit minimal à laquelle une séquence vidéo peut être encodée. Le schéma 5/3 nécessite alors un débit minimal nécessairement plus important que celui de Haar et n'est donc généralement pas le plus efficace dans les très bas débits.

La transformée à vecteur de mouvement unique est un compromis entre la transformée de Haar et la transformée 5/3 : elle ne nécessite qu'un seul champ de mouvement et bénéficie cependant d'opérateurs bidirectionnels. Il est raisonnable de penser qu'elle compense les désavantages des transformées de Haar et 5/3. Ceci est confirmé expérimentalement sur la séquence *Tempête* où l'on observe des résultats supérieurs à la transformée de Haar dans les bas débits et des résultats similaires au filtre 5/3 dans les moyen et haut débits. Ceci reste vrai sur *Mobile* dans les bas débits mais pas dans les débits supérieurs où la transformée 5/3 se montre plus efficace. Cependant, sur les séquences possédant une forte activité de mouvement comme *Stefan* et *Foreman*, les résultats ne sont

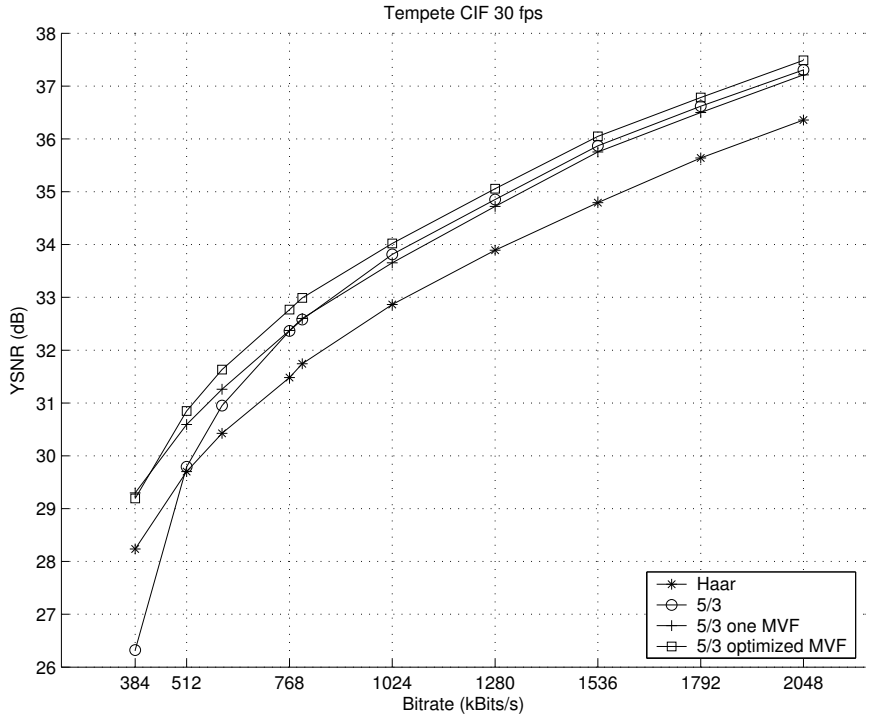


YSNR (in dB)	512 kbs	768 kbs	1024 kbs	1536 kbs	2048 kbs
Haar	27.27	29.78	31.35	33.61	35.30
5/3	26.53	30.39	32.27	34.60	36.33
5/3 one MVF	27.05	29.37	31.03	33.45	35.45
5/3 optimized MVF	28.19	30.81	32.52	34.70	36.47

FIG. 4.3 – Courbes et tableaux de débit-distorsion obtenus pour différents filtres temporels et différents débits sur la séquence *Mobile* CIF 30 Hz.

pas encourageants et sont moins bons à tous les débits que les autres transformées temporelles. En effet, ces séquences contiennent des mouvements rapides et complexes qui ne satisfont pas la contrainte d'un mouvement apparent souple et uniforme, attendue par la transformée à vecteur de mouvement unique. Il en résulte une plus grande erreur de prédiction temporelle, desservant ainsi l'efficacité de codage de la transformée 5/3 à vecteur de mouvement unique.

Nous comparons maintenant la transformée temporelle 5/3 avec prédiction itérative bidirectionnelle jointe par rapport aux autres transformées. Il apparaît clairement qu'elle donne *systématiquement* les meilleurs résultats sur toutes les séquences et à tous les débits, comparé aux autres transformées. Nous observons ainsi des gains moyens d'environ 0.5 dB avec des pointes à plus de 1.3 dB sur les séquences *Stefan* et *Mobile*. Ceci montre que l'algorithme d'estimation *jointe* des champs de mouvement avant et arrière améliore significativement la prédiction temporelle à moyen et haut débit. De plus, l'algorithme augmente la cohérence des champs de mouvement, expliquant ainsi le gain visible dans les bas débits, où une majeure partie du budget de codage est consacré aux vecteurs de mouvement. La transformée temporelle 5/3 avec prédiction jointe des champs de mouvement apparaît donc compétitive même à bas débit, comparée à la transformée de Haar. En effet, dans l'implémentation actuelle de l'algorithme itératif, une étape d'optimisation



YSNR (in dB)	512 kbs	768 kbs	1024 kbs	1536 kbs	2048 kbs
Haar	29.70	31.48	32.86	34.79	36.35
5/3	29.79	32.36	33.81	35.86	37.30
5/3 one MVF	29.29	32.37	33.65	35.75	37.21
5/3 optimized MVF	30.84	32.76	34.01	36.04	37.48

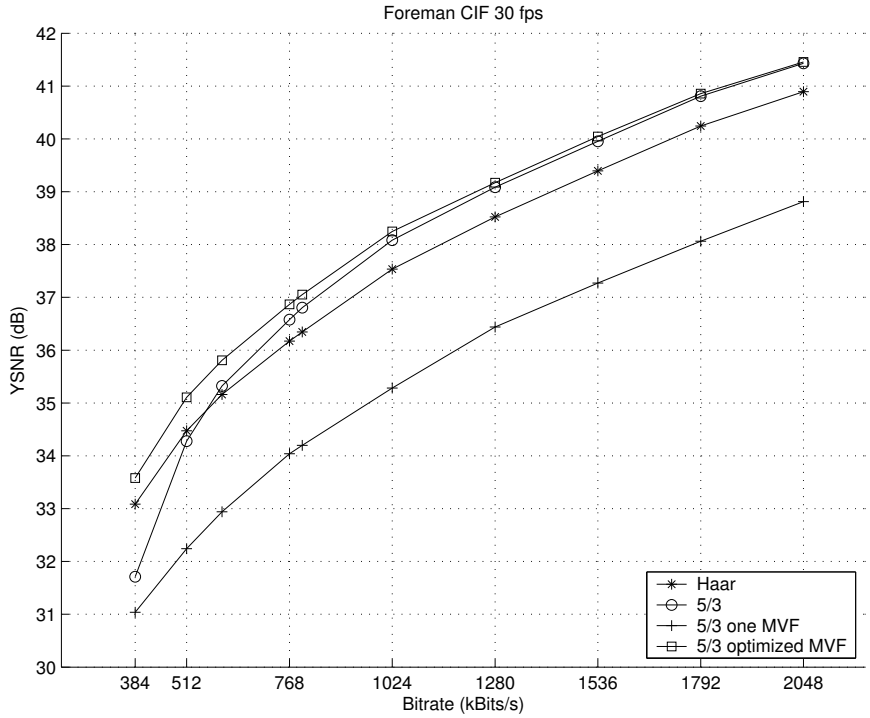
FIG. 4.4 – Courbes et tableaux de débit-distorsion obtenus pour différents filtres temporels et différents débits sur la séquence *Tempête* CIF 30 Hz.

débit-distorsion du champ de mouvement comme décrite dans la section 2.2.4 est réalisée à chaque itération par élagage. Les deux champs de mouvement ainsi obtenus sont alors nettement plus réguliers que ceux obtenus par une approche indépendante et une partie des gains de codage peut être expliqué par ce phénomène. Cette transformée possède ainsi une meilleure prédiction temporelle *et* nécessite la même quantité d'information que la transformée de Haar pour encoder ses champs de mouvement. Ceci explique pourquoi elle donne de meilleurs résultats que les autres transformées, même à bas débit.

Performance de décorrélation temporelle

Comme vu dans la section 2.2.5, le codage efficace des champs de mouvement avec une pleine exploitation de leurs dépendances spatio-temporelles est un sujet ouvert et actif, abordé par de nombreux travaux [150, 152]. Il n'est ainsi pas simple d'apprécier la performance de décorrélation opérée par une transformée temporelle à partir de sa seule efficacité de codage, du fait de la corrélation entre cette dernière et l'efficacité de codage des champs de mouvement.

Afin d'apprécier l'efficacité de la décorrélation temporelle opérée par les transformées 5/3 classique et 5/3 avec estimation jointe des champs de mouvement, nous avons procédé à des simulations de codage en excluant du budget d'encodage le débit alloué au



YSNR (in dB)	512 kbs	768 kbs	1024 kbs	1536 kbs	2048 kbs
Haar	34.47	36.17	37.53	39.39	40.89
5/3	34.27	36.57	38.08	39.95	41.43
5/3 one MVF	32.24	34.03	35.28	37.27	38.81
5/3 optimized MVF	35.10	36.86	38.24	40.04	41.45

FIG. 4.5 – Courbes et tableaux de débit-distorsion obtenus pour différents filtres temporels et différents débits sur la séquence *Foreman* CIF 30 Hz.

codage des champs de mouvement. Nous nous plaçons ainsi dans une situation idéale où les meilleurs champs de mouvement sont utilisés pour la compensation de mouvement et où leur coût de codage est nul ou négligeable. Dans le codec MC-EZBC, de tels champs de mouvement sont obtenus par la suppression de l'étape d'élagage. Ils contiennent un seul vecteur de mouvement pour chaque bloc de 4×4 pixels et sont donc presque denses. Sous ces hypothèses et en négligeant le coût de codage de ces champs, nous obtenons les résultats de codage sur la séquence *Mobile* illustrés par le Tab. 4.5.

YSNR (en dB)	512 kbs	768 kbs	1024 kbs	1536 kbs	2048 kbs
5/3	32.76	34.12	35.13	36.89	38.28
5/3 optimized MVF	33.00	34.38	35.43	37.22	38.67

TAB. 4.5 – Mesures de distorsion obtenues pour différents filtres temporels sur la séquence *Mobile* CIF 30 Hz, sans considérer le coût des champs de mouvement. La compensation de mouvement a été effectuée avec des champs quasiment denses.

Nous observons un gain en PSNR d'environ 0.3-0.4 dB sur tous les débits, en faveur de la transformée avec optimisation jointe des champs de mouvement. Ceci est comparable avec les résultats d'efficacité de codage réel à haut débit, observés précédemment lorsque

le coût de codage des champs de mouvement était inclus dans le débit total. Dans les bas débits, la différence d'efficacité de codage entre les deux transformées était nettement plus importante que 0.3 dB. Ces résultats justifient une fois de plus la propension de la transformée 5/3 optimisée jointe à lisser les champs de mouvement et à rendre leur codage moins coûteux. Ceci conclue et montre que l'algorithme itératif d'optimisation des champs de mouvement influence l'efficacité de codage en assurant à la fois une meilleure décorrélation temporelle tout en réduisant le débit nécessaire à l'encodage des champs.

Efficacité de codage avec le codec Vidwav

L'algorithme de recherche jointe itérative a de plus été intégré sur le codeur vidéo MPEG-Vidwav [97], qui utilise l'estimateur de mouvement mis en œuvre dans le codec H.264 . La mise en place de l'algorithme a cependant été effectuée seulement sur les blocs de taille 16×16 pixels, sachant que ces derniers constituent plus de 70% des décisions de modes de prédiction temporelle. De plus, l'algorithme de recherche jointe n'est utilisé qu'avec une demi-itération, n'augmentant pas ainsi la complexité du filtre temporel comparé à une transformée 5/3 classique. Les simulations expérimentales ont été conduites sur les séquences *Mobile* et *Soccer* en utilisant les conditions de scalabilité spatiales, temporelles et en débit définies dans le descriptif [25] des activités exploratoires du groupe de travail MPEG-Vidwav de Palma.

Les résultats de simulations obtenus sur les séquences *Mobile* et *Soccer* sont présentés dans les Tab. 4.6 et 4.7. Nous observons des gains en PSNR faibles d'environ 0.05 dB, loin des gains d'environ 0.7 dB obtenus avec le schéma de codage MC-EZBC. Ceci peut s'expliquer par le fait que le codec Vidwav utilise l'algorithme d'estimation de mouvement à modes de prédiction du codec H.264. En effet, cet algorithme choisit pour chaque bloc le meilleur mode de prédiction de façon à minimiser le coût du bloc. Or le mode de prédiction le plus observé lors de nos simulations est le mode bidirectionnel 16×16 utilisant un couple de vecteurs mouvements déduits des vecteurs des blocs voisins. Bien que notre algorithme réduise le coût d'un bloc, il nécessite cependant l'encodage systématique de nouveaux vecteurs. Il ne rivalise alors que rarement avec ce mode où les vecteurs ne sont pas encodés, expliquant les gains faibles observés.

YSNR (en dB)	QCIF 15 Hz 96 kbs	QCIF 15 Hz 128 kbs	CIF 15 Hz 256 kbs	CIF 30 Hz 384 kbs
5/3	28.93	30.82	28.14	29.30
5/3 optimisé	28.96	30.88	28.18	29.35

TAB. 4.6 – Mesures de distorsion obtenues avec le codec Vidwav en utilisant ou non l'algorithme de recherche bidirectionnelle optimal sur la séquence *Mobile* CIF 30 Hz.

YSNR (en dB)	QCIF 15 Hz 96 kbs	QCIF 15 Hz 128 kbs	CIF 30 Hz 256 kbs	CIF 30 Hz 384 kbs	4CIF 60 Hz 3072 kbs
5/3	31.67	35.65	31.78	35.00	36.57
5/3 optimisé	31.69	35.67	31.83	35.06	36.63

TAB. 4.7 – Mesures de distorsion obtenues avec le codec Vidwav en utilisant ou non l'algorithme de recherche bidirectionnelle optimal sur la séquence *Soccer* 4CIF 60 Hz.

De plus, nous avons souhaité évaluer de façon objective les performances du codec Vidwav muni de l'algorithme d'estimation jointe comparé au codec SVC, en cours de normalisation par le groupe ITU/MPEG JVT (*Joint Video Team*). A cette fin, nous avons choisi un extrait de la séquence vidéo haute définition *Vintage Car* de résolution 704×896 à 30 Hz que nous avons encodé avec le codec Vidwav muni de l'algorithme d'estimation jointe et avec le codec SVC JSVM 2.0. En suivant un scénario de scalabilité spatiale, temporelle et en qualité imposé, nous obtenons les résultats de codage suivants, exprimés en terme de PSNR moyen calculé sur les images décodées et présentés dans le Tab. 4.8.

YSNR (en dB)	176×224 15 Hz 96 kbs	352×448 30 Hz 384 kbs	704×896 30 Hz 1024 kbs
Vidwav + Joint	31.19	32.14	33.84
SVC JSVM 2.0	33.12	33.30	32.70

TAB. 4.8 – Mesures de distorsion obtenues pour plusieurs points de scalabilité en utilisant le codec Vidwav muni de l'algorithme de recherche jointe bidirectionnelle et le codec SVC JSVM 2.0 sur la séquence *Vintage Car* 704×896 à 30 Hz.

Le codec Vidwav muni de l'algorithme d'estimation jointe affiche de bonnes performances à la résolution nominale de la séquence vidéo où il surpasse le schéma de codage SVC d'environ 1.1 dB. Cependant, il offre une efficacité de codage moins bonne dans des résolutions inférieures. Les résultats obtenus en utilisant une itération complète n'offrent qu'une amélioration de 0.01 dB et n'ont pas été présentés. Afin d'étudier les raisons de la contre-performance observée à bas débit, nous avons tracé sur les Figs. 4.6 et 4.7 l'évolution du PSNR des images reconstruites aux débits respectifs de 1024 kbs et 384 kbs.

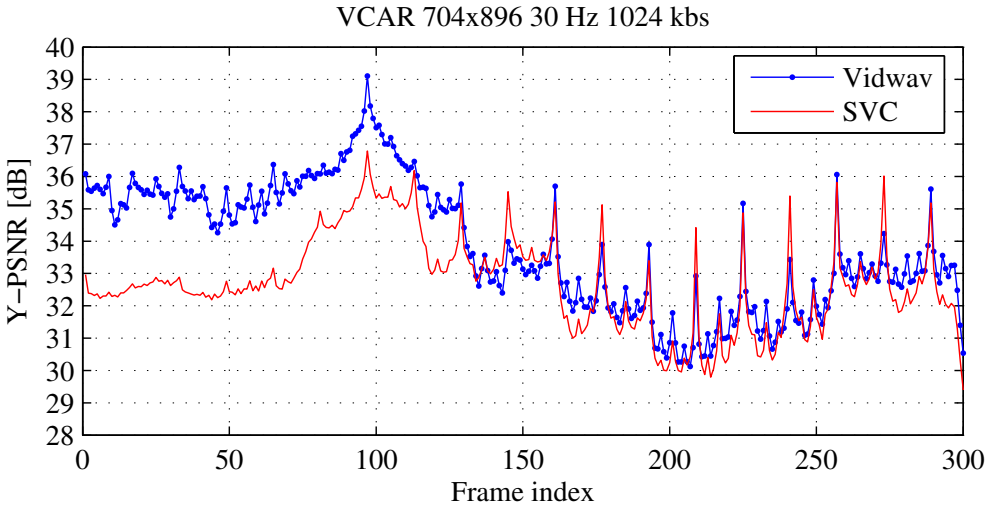


FIG. 4.6 – Comparaison de l'évolution du PSNR des images reconstruites de la séquence *Vintage Car* décodée à la résolution 704×896 à 30 Hz avec un débit de 1024 kbs, avec les codecs Vidwav et SVC.

On observe un comportement similaire dans l'évolution du PSNR des séquences décodées aux deux résolutions spatiales : le codec Vidwav surpasse le codec SVC d'environ 3 dB sur le premier tiers de la séquence. Sur les deux tiers restant et dans le cas de la résolution 704×896, le codec Vidwav offre un PSNR proche de celui du codec SVC. Cependant,

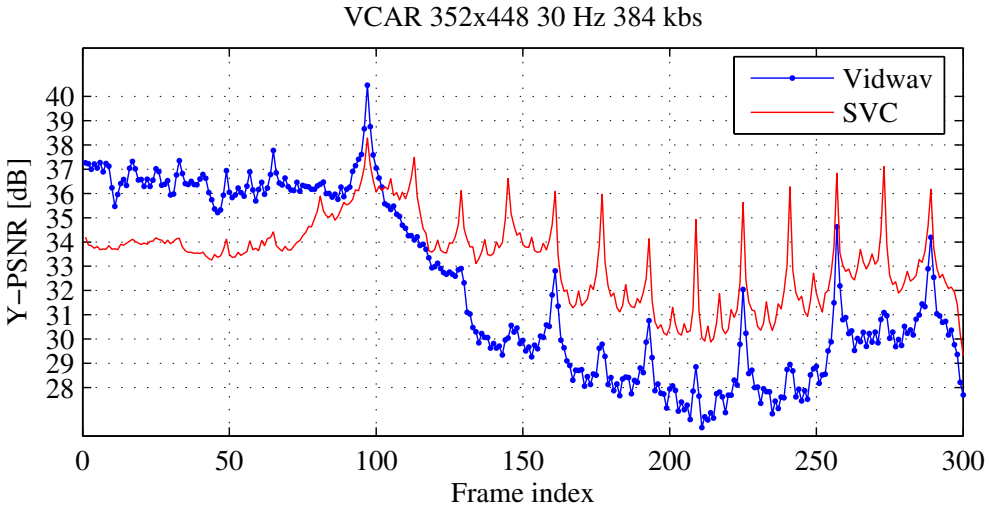


FIG. 4.7 – Comparaison de l'évolution du PSNR des images reconstruites de la séquence *Vintage Car* décodée à la résolution 352×448 à 30 Hz avec un débit de 384 kbs, avec les codecs Vidwav et SVC.

dans le cas de la résolution 352×448, le PSNR obtenu avec le codec Vidwav reste nettement en deçà, avec une baisse d'environ 4 dB. Le mouvement faible et uniforme présent dans le premier tiers de la séquence *Vintage Car* et la meilleure aptitude du codec SVC à gérer des champs de mouvement scalables peuvent justifier cet écart de PSNR. Cela explique ainsi la moins bonne efficacité du codec Vidwav observée à la résolution 352×448.

Enfin, nous avons illustré sur la Fig. 4.8 des images reconstruites par les deux codecs vidéos de la séquence *Vintage Car* à la résolution 704×896 avec un débit de 1024 kbs. On observe clairement plus de détails sur l'image reconstruite avec le codec Vidwav muni de l'estimation jointe. Le gravier, le feuillage de fond et les contours de la voiture y sont plus nets ; l'image présente ainsi un piqué supérieur à celle obtenue par le codec SVC.

4.1.5 Conclusion

L'étude de l'opérateur de prédiction impliqué dans la transformée temporelle 5/3 a permis d'élaborer deux stratégies pour améliorer son efficacité de décorrélation. Tout d'abord, la constatation que les champs de vecteurs bidirectionnels n'étaient pas choisis de façon à minimiser l'erreur de prédiction temporelle nous a conduit à construire un algorithme d'estimation conjointe du mouvement. Cet algorithme possède de nombreux avantages. Tout d'abord, il apporte des gains substantiels en terme de PSNR allant de 0.5 dB à plus de 2 dB sur une large gamme de débits et de séquences vidéos. Nous avons ainsi montré qu'il surpasse systématiquement le filtre temporel 5/3 et le filtre de Haar, même à bas débit. De plus, c'est un algorithme itératif qui possède une complexité variable, en fonction des besoins d'une application. Son utilisation avec une demi-itération possède ainsi une complexité équivalente au filtre temporel 5/3 et conduit à une meilleure efficacité de codage. Enfin, il est simple à mettre en œuvre et ne nécessite pas la construction d'un nouvel estimateur de mouvements.

En souhaitant concilier une prédiction bidirectionnelle tout en n'utilisant qu'un seul champ de mouvement, nous avons de plus construit une transformée 5/3 utilisant deux

FIG. 4.8 – Reconstruction d'une image issue du codage de la séquence *Vintage Car* de résolution 704×896 à 30 Hz pour un débit de 1024 kbs avec le codec Vidwav muni de l'estimation jointe (gauche) et avec le codec SVC JSVM 2.0 (droite).

champs de mouvement avant et arrière opposés. Cependant, bien que cette transformée offre une bonne efficacité de codage en présence d'un mouvement apparent uniforme et à bas débits, elle ne rivalise pas avec la polyvalence de la transformée 5/3 avec estimation conjointe du mouvement.

4.2 Transformée temporelle 5/3 de sens uniforme

La section précédente montre comment construire une transformée temporelle en choisissant les champs de mouvement de façon à minimiser la distorsion des sous-bandes temporelles de détail. On observe alors une augmentation *objective* de l'efficacité de codage en terme de PSNR. Cependant, bien que ce dernier soit une mesure correcte de la qualité objective, il traduit parfois mal l'apparition d'artefacts, d'effets d'anneaux (*ringing*) ou de discontinuités très visibles qui peuvent apparaîtrent dans les séquences vidéos décompressées, sans que le PSNR en soit affecté.

En particulier, un inconvénient majeur des transformations temporelles de type Haar ou 5/3 est leur propension à introduire des artefacts fantômes dans les sous-bandes d'approximation. Ces artefacts nuisent à l'efficacité globale du schéma de codage et dégradent la qualité visuelle des images décodées à bas débit. Nous nous proposons dans la section 4.2.1 d'étudier les causes de ces artefacts et commentons quelques propositions faites dans la littérature pour y remédier. Nous introduisons alors dans la section 4.2.2 la transformée temporelle 5/3 uniforme, basée sur le filtre 5/3 classique qui, par construction même, ne créé pas de tels artefacts. On montre expérimentalement que cette transformée

temporelle améliore nettement la qualité visuelle des sous-bandes d'approximation et augmente l'efficacité globale de codage. Enfin, ces résultats encourageants ont conduit à la publication d'un article de conférence [99].

Dans la continuation de nos méthodes développées dans la section précédente, nous décrivons en section 4.2.3 un algorithme de calcul optimal des champs de vecteurs mis en jeux dans la transformée 5/3 uniforme. Nous observons alors expérimentalement une nouvelle amélioration des performances et relatons nos travaux dans [98].

4.2.1 Artefacts fantômes et mise à jour

Notre schéma de codage muni de la transformée temporelle 5/3 a tendance à produire des artefacts très visibles dans certaines séquences à bas débit ou dans les images d'approximation. Ces artefacts ressemblent à des réminiscences locales d'objets présents dans des images antérieures ou postérieures à l'image courante et sont nommés pour cette raison, artefacts fantômes (*ghosting artefacts*). Ils sont particulièrement visibles sur les sous-bandes temporelles d'approximation de la séquence *Stefan*, illustrées en Fig. 4.9. On y voit ainsi clairement la présence de plusieurs pieds et de plusieurs balles de tennis.

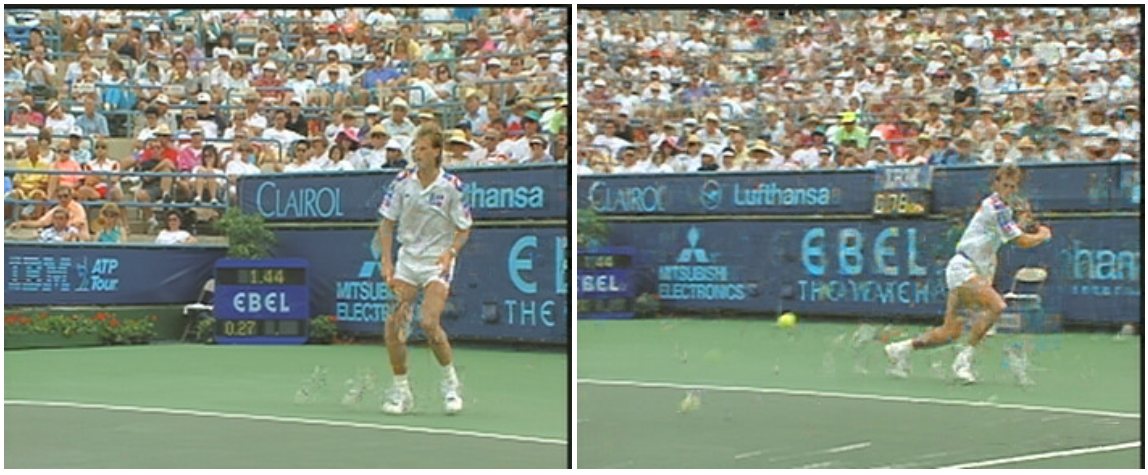


FIG. 4.9 – Présence d'artefacts fantômes sur les sous-bandes d'approximation issues du troisième niveau de la décomposition temporelle 5/3 de la séquence CIF 30 Hz *Stefan*.

Ces artefacts fantômes sont gênants pour plusieurs raisons. Tout d'abord, ils créent des discontinuités locales qui perturbent visuellement les images et induisent l'apparition de grands coefficients d'ondelettes. Ils augmentent ainsi le coût de codage des images et entraînent une diminution globale des performances du codec vidéo. De plus, ces artefacts dégradent la qualité visuelle des images d'approximation et nuisent ainsi à la scalabilité temporelle du schéma. Enfin, ils se propagent dans les niveaux temporels suivants et diminuent l'efficacité de la prédiction temporelle effectuée dans les étages supérieurs.

La présence de ces artefacts est due à l'étape de mise à jour du filtre temporel 5/3. En effet, comme vu dans la section 3.1.3, la pseudo-inversion de l'opérateur de compensation de mouvement $\mathcal{C}$ crée durant cette étape une mosaïque hétérogène de zones non-connectées, simplement connectées ou connectées de façon multiple. Le filtrage passe-bas subséquent engendre alors des zones de caractéristiques visuelles différentes, créant les artefacts.

Plusieurs approches ont été proposées pour limiter la présence des artefacts fantômes. Après avoir observé leur existence, Reichel [117] introduit par exemple une transformée de Haar non compensée en mouvement qui décide dynamiquement pour chaque pixel si il est transformé ou non, en fonction d'un critère de seuil basé sur la valeur des coefficients de détail quantifié du niveau temporel précédent. Cette solution donne des résultats visuellement probants mais n'offre pas une bonne efficacité de codage. De plus, le recours fait à un quantificateur dans la boucle d'encodage rend ce schéma de codage vidéo non scalable. Une approche plus efficace a été aussi proposée par Song [133] où les auteurs présentent une étape de mise à jour adaptative en utilisant un critère de seuil sur les coefficients, basé sur un modèle numérique simple de la vision humaine.

Enfin, on remarquera que la présence de zones non filtrées est intimement liée à la non-inversibilité des champs de mouvement durant l'étape de mise à jour. Ce problème a tout d'abord été observé par Secker et Taubman [126] où les auteurs préconisent l'utilisation d'un modèle de mouvement non basé sur des blocs mais sur une grille triangulaire déformable, presque toujours inversible (sauf en cas de retournement de maille). Ce type de modèle est cependant coûteux à encoder et difficile à estimer. Konrad [69] a étudié le problème de façon plus générale et montre l'existence d'une transformée temporelle de Haar *transverse* où le mouvement ne nécessite pas d'inversion durant l'étape de mise à jour. Cependant, cette approche n'est pas généralisable dans le cas du filtre 5/3. Poursuivant ces travaux, André [13] promeut alors l'utilisation du filtre temporel LS (2,0), obtenu par suppression de l'étape de mise à jour de la transformée temporelle 5/3 et permettant ainsi d'éliminer les artefacts fantômes de façon draconienne. Cependant, cette amputation peut aussi engendrer une baisse significative de l'efficacité de codage, pouvant atteindre 1 dB lors de son utilisation sur des séquences fluides comme *Mobile*. Des résultats expérimentaux détaillés illustrant les performances du filtre temporel 5/3 sans mise à jour sont ainsi présentés dans la section 4.2.4.

4.2.2 Transformée temporelle 5/3 de sens de mouvement uniforme

Avant de décrire les détails de la construction de la transformée 5/3 de sens de mouvement uniforme, nous nous proposons tout d'abord d'étudier quelques propriétés de connectivité de la transformée temporelle 5/3.

Transformée temporelle 5/3 classique et connectivité

Rappelons les équations de la transformée temporelle 5/3, dont les opérateurs sont illustrés par la Fig. 4.10 :

$$h_t^0(\boldsymbol{n}) = x_{2t+1}(\boldsymbol{n}) - \frac{1}{2}\big(\mathcal{C}(x_{2t}, \boldsymbol{v}_{2t+1}^+)(\boldsymbol{n}) + \mathcal{C}(x_{2t+2}, \boldsymbol{v}_{2t+1}^-)(\boldsymbol{n})\big) \tag{4.14}$$

$$l_t^0(\boldsymbol{n}) = x_{2t}(\boldsymbol{n}) + \gamma\,\mathcal{C}^{-1}(h_{t-1}, \boldsymbol{v}_{2t-1}^-)(\boldsymbol{n}) + \delta\,\mathcal{C}^{-1}(h_t, \boldsymbol{v}_{2t+1}^+)(\boldsymbol{n}) \tag{4.15}$$

$$h_t(\boldsymbol{n}) = 1/\sqrt{2}\; h_t^0(\boldsymbol{n}) \tag{4.16}$$

$$l_t(\boldsymbol{n}) = \sqrt{2}\; l_t^0(\boldsymbol{n}) \tag{4.17}$$

$$\text{avec} \begin{cases} \gamma = \delta = 1/4 & \text{si } \boldsymbol{n} \text{ est connecté des deux côtés} \\ \gamma = 1/2 \text{ et } \delta = 0 & \text{si } \boldsymbol{n} \text{ est connecté seulement à gauche} \\ \gamma = 0 \text{ et } \delta = 1/2 & \text{si } \boldsymbol{n} \text{ est connecté seulement à droite} \\ \gamma = 0 \text{ et } \delta = 0 & \text{si } \boldsymbol{n} \text{ n'est pas connecté} \end{cases}$$

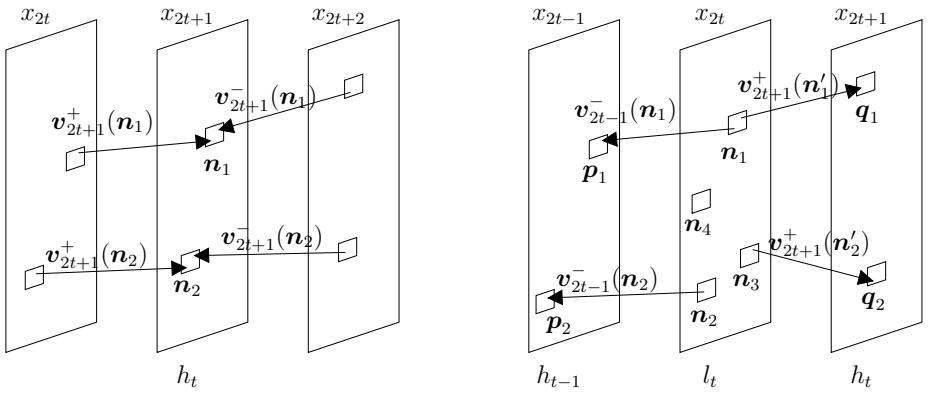


FIG. 4.10 – Opérateur de prédiction (gauche) et de mise à jour (droite) mis en jeu dans la transformée temporelle 5/3.

Lors de la prédiction décrite par l'équation (4.14), chaque pixel de l'image x_{2t+1} est toujours connecté à un seul pixel de l'image précédente x_{2t} et à un seul autre de l'image suivante x_{2t+2}. Chaque pixel de x_{2t+1} sera ainsi toujours prédit bidirectionnellement. C'est une propriété importante qui améliore généralement l'efficacité de codage, en particulier en présence d'un mouvement fluide. C'est aussi une des raisons pour laquelle la transformée 5/3 assure une meilleure décorrélation temporelle que la transformée de Haar. Cependant, lorsqu'une zone de x_{2t+1} ne peut être prédite bidirectionnellement, par exemple lors d'une occlusion ou d'une coupure de plan (*scene cut*), cette propriété n'est pas souhaitable car un des pixels prédicteur sera incorrect.

Lors de l'étape de mise à jour décrite par l'équation (4.15), les possibilités de connectivité sont nettement plus grandes. Ainsi, chaque pixel de l'image x_{2t} peut être connecté à zéro, un ou plusieurs pixels de l'image précédente x_{2t-1} et à zéro, un ou plusieurs pixels de l'image suivante x_{2t+1}. En fonction de l'état de connectivité d'un pixel et comme précisé dans la section 3.1.3, il sera alors filtré bidirectionnellement s'il est connecté dans l'image précédente et dans l'image suivante, monodirectionnellement si il est connecté dans une seule de ces images et ne sera pas filtré si il n'est pas connecté.

Le tableau 4.9 résume dans le cas de la transformée temporelle 5/3 les relations entre l'état de connexion d'un pixel et le filtrage qu'il subit lors des étapes de prédiction et de mise à jour. Il se lit comme ceci : lors de l'étape de prédiction, tous les pixels sont connectés bidirectionnellement et sont prédits par le prédicteur 5/3. Il ne peut y avoir de pixels non-connectés ou simplement connectés lors de cette étape. De plus, lors de la mise à jour, les pixels non-connectés ne sont pas filtrés, les pixels connectés sur une seule image sont filtrés par un filtre de type Haar et les pixels connectés bidirectionnellement sont filtrés par un filtre 5/3.

État de connexion d'un pixel	0	1	2
Prédiction P	N/A	N/A	Prédiction 5/3
Mise à jour U	Pas de filtrage	Filtrage Haar	Filtrage 5/3

TAB. 4.9 – Relations entre l'état de connexion d'un pixel non connecté (0), connecté sur une seule des deux images (1) et connecté sur les deux images (2) et le filtrage qu'il subira dans la transformée temporelle 5/3 classique.

Comme vu précédemment, l'opération de mise à jour induit la création de zones non-filtrées et filtrées au sein des sous-bandes temporelles d'approximation l_t. Ces zones partagent des caractéristiques différentes et leur mélange créé une mosaïque de régions inhomogènes, causant l'apparition des artefacts fantômes mentionnés plus haut.

Transformée temporelle 5/3 uniforme

Nous souhaitons donc construire une transformée temporelle où chaque pixel soit *toujours* connecté au moins à un autre, lors des étapes de prédiction et de mise à jour. Ceci est possible en introduisant deux autres champs de mouvement lors de la mise à jour mais cette solution ne se relève pas rentable, à cause du surcoût engendré par le codage des champs supplémentaires.

Une façon simple pour parvenir à cette propriété consiste à modifier la transformée temporelle 5/3 classique en utilisant deux champs de mouvement orientés dans la même direction, comme illustré par la Fig. 4.11. Cette modification nous conduit à la construction d'une nouvelle transformée, nommée transformée temporelle 5/3 uniforme, en raison du sens uniforme des champs de mouvement qu'elle met en jeu.

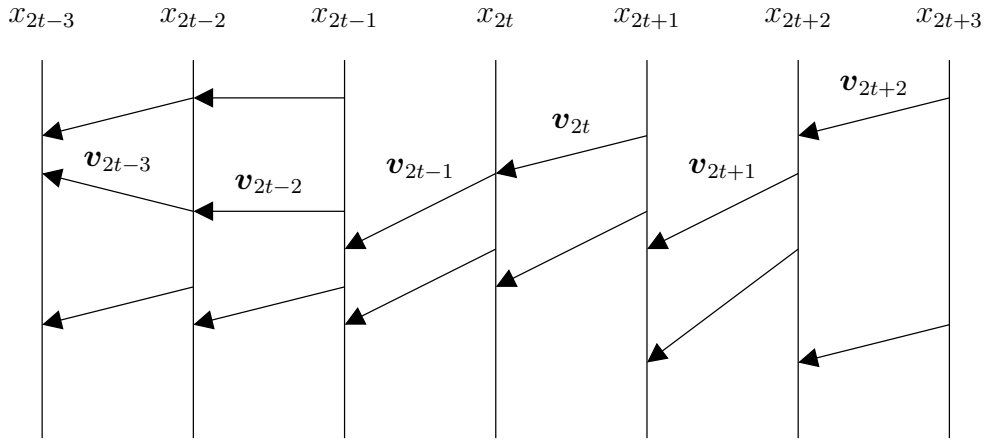


FIG. 4.11 – Orientation du mouvement dans la transformée 5/3 uniforme.

La notation indicée $+$ et $-$ n'étant plus nécessaire pour indiquer le sens de prédiction des champs de mouvement $\boldsymbol{v}$, ces derniers sont désormais notés $\boldsymbol{v}_t$ où t peut prendre des valeurs paires ou impaires. On définit ainsi un champ de vecteur mouvement arrière $\boldsymbol{v}_t$, prédisant chaque image x_t à partir de l'image suivante x_{t+1}. Le choix de la direction arrière est arbitraire et aurait pu être fait dans la direction avant.

La transformée temporelle 5/3 uniforme peut alors s'exprimer sous forme lifting par les opérateurs de prédiction et de mise à jour décrits par les Figs. 4.12 et 4.13. Elle est alors régie par les équations suivantes :

$$h_t(\boldsymbol{n}) = x_{2t+1}(\boldsymbol{n}) + \alpha\,\mathcal{C}^{-1}(x_{2t}, \boldsymbol{v}_{2t})(\boldsymbol{n}) + \beta\,\mathcal{C}(x_{2t+2}, \boldsymbol{v}_{2t+1})(\boldsymbol{n}) \tag{4.18}$$

$$\text{avec} \begin{cases} \alpha = \beta = -1/2 & \text{si } \boldsymbol{n} \text{ est connecté des deux côtés} \\ \alpha = 0 \text{ et } \beta = 1 & \text{si } \boldsymbol{n} \text{ n'est connecté qu'à gauche} \end{cases}$$

$$l_t(\boldsymbol{n}) = x_{2t}(\boldsymbol{n}) + \delta\,\mathcal{C}^{-1}(h_{t-1}, \boldsymbol{v}_{2t-1})(\boldsymbol{n}) + \gamma\,\mathcal{C}(h_t, \boldsymbol{v}_{2t})(\boldsymbol{n}) \tag{4.19}$$

$$\text{avec} \begin{cases} \delta = \gamma = 1/4 & \text{si } \boldsymbol{n} \text{ est connecté des deux côtés} \\ \delta = 0 \text{ et } \gamma = 1/2 & \text{si } \boldsymbol{n} \text{ n'est connecté qu'à gauche} \end{cases}$$

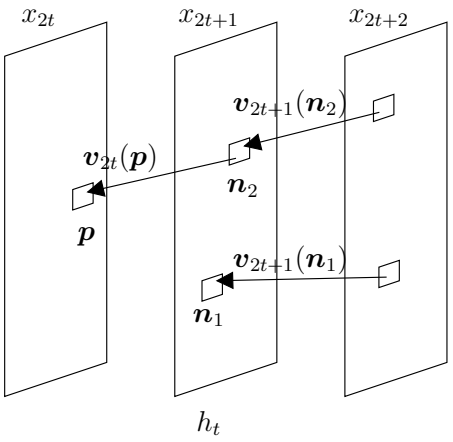


FIG. 4.12 – Opérateur de prédiction mis en jeu dans la transformée 5/3 uniforme. Le pixel $\boldsymbol{n}_1$ est simplement connecté tandis que le pixel $\boldsymbol{n}_2$ est connecté des deux côtés.

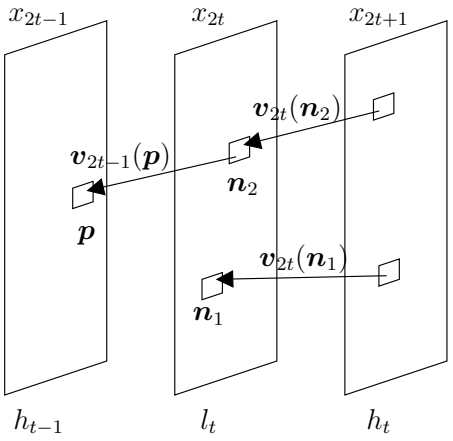


FIG. 4.13 – Opérateur de mise à jour utilisé dans la transformée 5/3 uniforme. Le pixel $\boldsymbol{n}_1$ est simplement connecté tandis que le pixel $\boldsymbol{n}_2$ est connecté des deux côtés.

Cette transformation possède une propriété importante : chaque pixel est toujours connecté à un autre dans l'image précédente, durant l'étape de prédiction et durant la mise à jour. Contrairement à la transformée temporelle 5/3 classique, cette caractéristique assure ainsi ne jamais avoir de zone découverte durant l'étape de mise à jour et permet alors à chaque pixel de toujours bénéficier au moins d'un filtrage temporel passe-bas mono-directionnel.

Lors de la prédiction décrite par l'équation (4.18), chaque pixel simplement connecté est prédit monodirectionnellement par un filtre de Haar et chaque pixel connecté bidirectionnellement est prédit par un filtre de type 5/3. Ainsi, comparé au filtre temporel 5/3 classique, la prédiction n'est pas toujours bidirectionnelle. Ceci est souvent un avan-

tage car le manque de connectivité dans une direction est souvent lié à l'occultation ou à l'apparition d'un objet, qui ne peuvent ainsi être prédit que dans une direction.

L'étape de mise à jour décrite par l'équation (4.19) est très similaire à celle de la prédiction. Chaque pixel subit ainsi un filtrage passe-bas mono ou bidirectionnel et il n'y a donc pas de pixels non-filtrés comme dans le filtre temporel 5/3 classique. Cette mise à jour est donc plus régulière et doit conduire à des images filtrées plus homogènes que celle obtenues avec le filtre temporel 5/3 classique. Comme évoqué précédemment dans la section 4.2.1, cette propriété doit contribuer à réduire la source majeure de création d'artefacts fantômes dans la transformée temporelle.

Le tableau 4.10 dresse les propriétés de connectivité de la transformé temporelle 5/3 uniforme. Mis en correspondance avec celui de la transformée 5/3 classique présenté en Fig. 4.9, il montre l'intérêt principal de la transformée 5/3 uniforme : tous les pixels sont au moins prédits et mis à jour par un pixel de l'image précédente. Ceci permet ainsi de résoudre élégamment le problème de gestion des occlusions lors de la prédiction et des zones non-connectées lors de la mise à jour.

Connectivité	0	1	2
Prédiction P	N/A	Prédiction Haar	Prédiction 5/3
Mise à jour U	N/A	Filtrage Haar	Filtrage 5/3

TAB. 4.10 – Relations entre l'état de connexion d'un pixel non connecté (0), connecté sur une seule des deux images (1) et connecté sur les deux images (2) et le filtrage qu'il subira dans la transformée temporelle 5/3 uniforme.

On notera qu'une variante de cette transformée temporelle, utilisant des champs de vecteurs orientés dans la même direction, a été étudiée indépendamment par Golwelkar et Woods [54]. Cependant, leur volonté sous-jacente semblait être avant tout de mettre en œuvre une transformée temporelle bidirectionnelle sans aborder le problème des artefacts fantômes. Ils mettent ainsi surtout en avant leur habilité à pouvoir traiter les images au fil de l'eau mais n'établissent pas de comparaison avec la transformée temporelle 5/3 classique. Enfin, les auteurs présentent des résultats expérimentaux nettement en deçà de ceux observés dans la section 4.2.4.

4.2.3 Prédiction bidirectionnelle optimale des zones découvertes

Nous abordons dans cette section le problème de l'estimation optimale des champs de mouvement impliqués dans la transformée 5/3 uniforme. En suivant la même approche que celle décrite dans la section 4.1 dans le cas de la transformée 5/3 classique, on s'intéresse à la minimisation d'un critère J basé sur les images de détail h_t, en espérant ainsi réduire leur coût de codage. Développons le critère J dans le cas de la transformée 5/3 uniforme décrite par l'équation de prédiction (4.18)et illustrée par la Fig. 4.12 :

$$
\begin{aligned}
J(\boldsymbol{v}_{2t}, \boldsymbol{v}_{2t+1}) &= \sum_{\boldsymbol{n}\in\mathcal{B}} d\Big[h_t(\boldsymbol{n})\Big] + \lambda\big(R(\boldsymbol{v}_{2t}) + R(\boldsymbol{v}_{2t+1})\big) \qquad (4.20)\\
&= \sum_{\boldsymbol{n}\in\mathcal{B}_1} d\Big[x_{2t+1}(\boldsymbol{n}) - \mathcal{C}(x_{2t+2}, \boldsymbol{v}_{2t+1})(\boldsymbol{n})\Big] + \lambda\big(R(\boldsymbol{v}_{2t}) + R(\boldsymbol{v}_{2t+1})\big)\\
&\quad + \sum_{\boldsymbol{n}\in\mathcal{B}_2} d\Big[x_{2t+1}(\boldsymbol{n}) - \frac{\mathcal{C}^{-1}(x_{2t}, \boldsymbol{v}_{2t})(\boldsymbol{n}) + \mathcal{C}(x_{2t+2}, \boldsymbol{v}_{2t+1})(\boldsymbol{n})}{2}\Big]
\end{aligned}
$$

où d est une mesure de distorsion quelconque, λ la contrainte de Lagrange, R le coût de codage d'un vecteur, $\mathcal{C}$ l'opérateur de compensation de mouvement et $\mathcal{B}$ est un bloc de l'image courante x_{2t+1} à prédire. Le bloc $\mathcal{B}$ est subdivisé en deux sous-ensembles $\mathcal{B} = \mathcal{B}_1 + \mathcal{B}_2$ où $\mathcal{B}_1$ est l'ensemble des points connectés uniquement sur l'image suivante et $\mathcal{B}_2$ l'ensemble des points connectés bidirectionnellement.

Il est possible de minimiser directement J, revenant ainsi à estimer conjointement $\boldsymbol{v}_{2t}$ et $\boldsymbol{v}_{2t+1}$. Cependant, la minimisation d'un problème à deux paramètres possède une complexité quadratique dont le coût de calcul est prohibitif. Il nous faut donc chercher une solution sous-optimale.

Nous nous proposons ici d'estimer d'abord le champ de mouvement $\boldsymbol{v}_{2t}$ puis d'estimer le champ $\boldsymbol{v}_{2t+1}$, en fonction de $\boldsymbol{v}_{2t}$ de façon à minimiser J. C'est une minimisation alternée qui peut être répétée itérativement de manière à converger vers un optimum local, de façon similaire à l'algorithme décrit dans la section 4.1.2. De plus, il est aisé de montrer que cette approche est nécessairement meilleure qu'une estimation indépendante de $\boldsymbol{v}_{2t}$ et $\boldsymbol{v}_{2t+1}$.

Algorithme

Considérons la Fig. 4.12. L'estimation de $\boldsymbol{v}_{2t}$ est faite tout d'abord par un algorithme d'appariement de blocs classique en prenant x_{2t} comme image courante et x_{2t+1} comme image de référence. La compensation de l'image x_{2t+1} par le champ $\boldsymbol{v}_{2t}$ fournit ainsi une bonne approximation de x_{2t}. De plus, en parcourant le champ $\boldsymbol{v}_{2t}$ à l'envers, on peut obtenir une approximation de x_{2t+1} par compensation inverse de l'image x_{2t}, donnée par $\mathcal{C}^{-1}(x_{2t}, \boldsymbol{v}_{2t})$ et illustré par la Fig. 4.14.

FIG. 4.14 – Compensation inverse d'une image $\mathcal{C}^{-1}(x_{2t}, \boldsymbol{v}_{2t})$ provenant de la séquence *Foreman*. Les zones noires représentent les zones découvertes.

Du fait de la non-inversibilité de $\mathcal{C}$, la compensée inverse $\mathcal{C}^{-1}(x_{2t}, \boldsymbol{v}_{2t})$ n'est pas définie partout. Elle comporte quelques zones découvertes car tous les pixels de x_{2t+1} ne sont pas reliés à x_{2t}. Cependant, ces régions ne correspondent pas nécessairement à des zones occlues par des déplacements d'objets. Au vu de la Fig. 4.14, il est ainsi raisonnable de penser que ces zones puissent être prédites par l'image suivante x_{2t+2}. De plus, chaque pixel de l'image x_{2t+1} est connecté à un pixel de l'image x_{2t+2}. On peut ainsi aisément prédire x_{2t+1} par la compensée directe $\mathcal{C}(x_{2t+2}, \boldsymbol{v}_{2t+1})$.

Nous souhaitons donc poursuivre cette idée : prédire l'image x_{2t+1} d'une part à partir de $\mathcal{C}^{-1}(x_{2t}, \boldsymbol{v}_{2t})$, malgré les zones découvertes qu'elle comporte et d'autre part à partir de

$\mathcal{C}(x_{2t+2}, \boldsymbol{v}_{2t+1})$. Ceci revient à effectuer tout d'abord une première prédiction incomplète par $\mathcal{C}^{-1}(x_{2t}, \boldsymbol{v}_{2t})$. Chaque pixel de x_{2t+1} bénéficiera alors d'une prédiction supplémentaire par $\mathcal{C}(x_{2t+2}, \boldsymbol{v}_{2t+1})$.

On cherche ainsi à estimer $\boldsymbol{v}_{2t+1}$ en souhaitant prédire $x_{2t+1} - \mathcal{C}^{-1}(x_{2t}, \boldsymbol{v}_{2t})$ à partir de $\mathcal{C}(x_{2t+2}, \boldsymbol{v}_{2t+1})$, tout en tenant compte des zones découvertes. Cependant, comment lancer une procédure d'appariement de blocs lorsque l'image courante comporte des trous ? Ceci peut être résolu grâce à l'algorithme suivant.

Estimation de $\boldsymbol{v}_{2t+1}$ sous la connaissance de $\boldsymbol{v}_{2t}$

La connaissance de $\boldsymbol{v}_{2t}$ nous permet de savoir si un pixel sera connecté avec la seule image x_{2t+2} ou avec les deux images x_{2t} et x_{2t+2}. Ainsi les pixels non-définis de $\mathcal{C}^{-1}(x_{2t}, \boldsymbol{v}_{2t})$ sont ceux connectés uniquement avec x_{2t+2} et appartiennent donc à $\mathcal{B}_1$, les autres appartenant à $\mathcal{B}_2$. Tout comme dans la section 4.1.2, on peut définir une image intermédiaire semi-compensée en mouvement a, représentant la première passe de prédiction :

$$a(\boldsymbol{n}) = \begin{cases} \frac{1}{2} x_{2t+1}(\boldsymbol{n}) & \text{si } \boldsymbol{n} \in \mathcal{B}_1 \\ x_{2t+1}(\boldsymbol{n}) - \frac{1}{2}\mathcal{C}^{-1}(x_{2t}, \boldsymbol{v}_{2t})(\boldsymbol{n}) & \text{si } \boldsymbol{n} \in \mathcal{B}_2 \end{cases} \tag{4.21}$$

Avant de pouvoir prédire l'image intermédiaire a par x_{2t+2}, il nous faut un algorithme d'estimation de mouvement tenant compte d'une pondération pour chaque pixel car les pixels de a appartenant à $\mathcal{B}_1$ ont une dynamique deux fois moindre que ceux appartenant à $\mathcal{B}_2$. Définissons alors une métrique $\mathcal{M}$ pour l'estimateur de mouvement entre deux images x et y en faisant intervenir le masque de pondération w :

$$\mathcal{M}(x, y) = \sum_{\boldsymbol{n} \in \mathcal{B}} d\Big[w(\boldsymbol{n}) \big(x(\boldsymbol{n}) - y(\boldsymbol{n}) \big) \Big] \tag{4.22}$$

$$\text{où } w(\boldsymbol{n}) = \begin{cases} 2 & \text{si } \boldsymbol{n} \in \mathcal{B}_1 \\ 1 & \text{si } \boldsymbol{n} \in \mathcal{B}_2 \end{cases} \tag{4.23}$$

On peut alors montrer que l'estimation du champ de mouvement $\boldsymbol{v}_{2t+1}$ par une procédure d'appariement de blocs munie de la métrique $\mathcal{M}$, en prenant a comme image courante et $x_{2t}/2$ comme image de référence, est équivalente à la minimisation du critère J pour le bloc $\mathcal{B}$, connaissant $\boldsymbol{v}_{2t}$. En effet, en utilisant l'opérateur d'appariement de blocs BM défini en section 4.1.2, on montre que :

$$\begin{aligned}
\boldsymbol{v}_{2t+1} &= \text{BM}_{\mathcal{M}}(a, \frac{1}{2} x_{2t}) \\
&= \arg\min_{\boldsymbol{v}_{2t+1}} \sum_{\boldsymbol{n} \in \mathcal{B}} d\Big[w(\boldsymbol{n}) \big(a(\boldsymbol{n}) - \frac{1}{2}\mathcal{C}(x_{2t}, \boldsymbol{v}_{2t+1})(\boldsymbol{n}) \big) \Big] + \lambda \big(R(\boldsymbol{v}_{2t}) + R(\boldsymbol{v}_{2t+1}) \big) \\
&= \sum_{\boldsymbol{n} \in \mathcal{B}_1} d\Big[x_{2t+1}(\boldsymbol{n}) - \mathcal{C}(x_{2t+2}, \boldsymbol{v}_{2t+1})(\boldsymbol{n}) \Big] + \lambda \big(R(\boldsymbol{v}_{2t}) + R(\boldsymbol{v}_{2t+1}) \big) \\
&\quad + \sum_{\boldsymbol{n} \in \mathcal{B}_2} d\Big[x_{2t+1}(\boldsymbol{n}) - \frac{\mathcal{C}^{-1}(x_{2t}, \boldsymbol{v}_{2t})(\boldsymbol{n}) + \mathcal{C}(x_{2t+2}, \boldsymbol{v}_{2t+1})(\boldsymbol{n})}{2} \Big] \\
&= \arg\min_{\boldsymbol{v}} J(\boldsymbol{v}_{2t}, \boldsymbol{v})
\end{aligned}$$

On obtient alors un champ optimal $\boldsymbol{v}_{2t+1}$ qui minimise l'énergie des images de détail h_t. Ne nécessitant que deux procédures de recherche de blocs, la transformée temporelle

5/3 uniforme ainsi définie a une complexité équivalente à celle d'une transformée 5/3 classique ou d'une transformée 5/3 avec une prédiction optimisée jointe à une demi-itération, telle que décrite dans la section 4.1.

Enfin, il est intéressant de remarquer que l'algorithme ci-dessus peut être étendu de façon itérative comme dans le cas de la prédiction jointe itérative, en réestimant successivement $\boldsymbol{v}_{2t}$ avec la connaissance de $\boldsymbol{v}_{2t+1}$ puis en réestimant $\boldsymbol{v}_{2t+1}$, etc. On converge ainsi vers un minimum local. Cependant, les résultats expérimentaux montrent que le gain obtenu par cette approche itérative est marginal, pour une complexité accrue. Nous nous cantonnerons donc à la transformée 5/3 uniforme précédemment définie.

4.2.4 Résultats expérimentaux

Réduction des artefacts fantômes

La série d'images présentée en Fig. 4.15 illustre les sous-bandes temporelles d'approximation issues de la décomposition de la séquence vidéo *Stefan* sur quatre niveaux temporels, obtenues avec le filtre 5/3 uniforme et le filtre 5/3 classique. Les images issues de la décomposition uniforme sont visiblement de meilleure qualité et ne présentent pas d'artefacts fantômes. On remarque ainsi que de nombreux gribouillis, présents aux abords des objets en mouvement dans le cas du filtre temporel 5/3 classique, sont absents dans le cas du filtre 5/3 uniforme. L'amélioration de l'aspect visuel des sous-bandes temporelles d'approximation permet d'augmenter ainsi la qualité de la scalabilité temporelle du schéma de codage.

Connectivité

Le Tab. 4.11 présente des résultats intéressants sur l'état de connectivité des pixels lors de l'étape de mise à jour de la transformée temporelle 5/3 uniforme et de la transformée 5/3 classique. Conformément aux propriétés de la transformée 5/3 uniforme, le pourcentage de pixels non-connectés, donc non-filtrés est nul à tous les niveaux. Par comparaison, ce taux atteint près de 10 % dans le dernier niveau temporel de la transformée 5/3 classique. De plus, on remarque que les taux de pixels simplement connectés et connectés bidirectionnellement sont plus importants dans le cas uniforme que dans le cas classique, à tous les niveaux temporels. Il en résulte une prédiction et un filtrage passe-bas de meilleure qualité.

Efficacité de codage

Afin d'évaluer son efficacité de codage, la transformée 5/3 uniforme a été mise en place au sein du codec MC-EZBC. Des simulations ont alors été conduites sur les séquences *Mobile*, *Tempête* et *City*, en utilisant une décomposition temporelle sur 5 niveaux et une estimation du mouvement au 1/8ème de pixel près. Les résultats exprimés en terme de Y-PSNR sont présentés dans les Tabs. 4.12, 4.13 et 4.14 et sont mis en comparaison avec le filtre temporel 5/3 classique, muni de la prédiction optimisée jointe itérative décrite dans la section 4.1.2 et avec le filtre temporel 5/3 classique sans mise à jour.

Afin de comparer notre schéma de codage avec un codec normatif à l'état-de-l'art, nous avons ajouté les performances débit-distorsion obtenues avec le codec H.264/AVC, dont les caractéristiques sont rappelées dans la section 2.1.2. Les conditions de simulations sont celles utilisées lors de l'appel à propositions MPEG [8] : utilisation du JSVM 7.3 avec

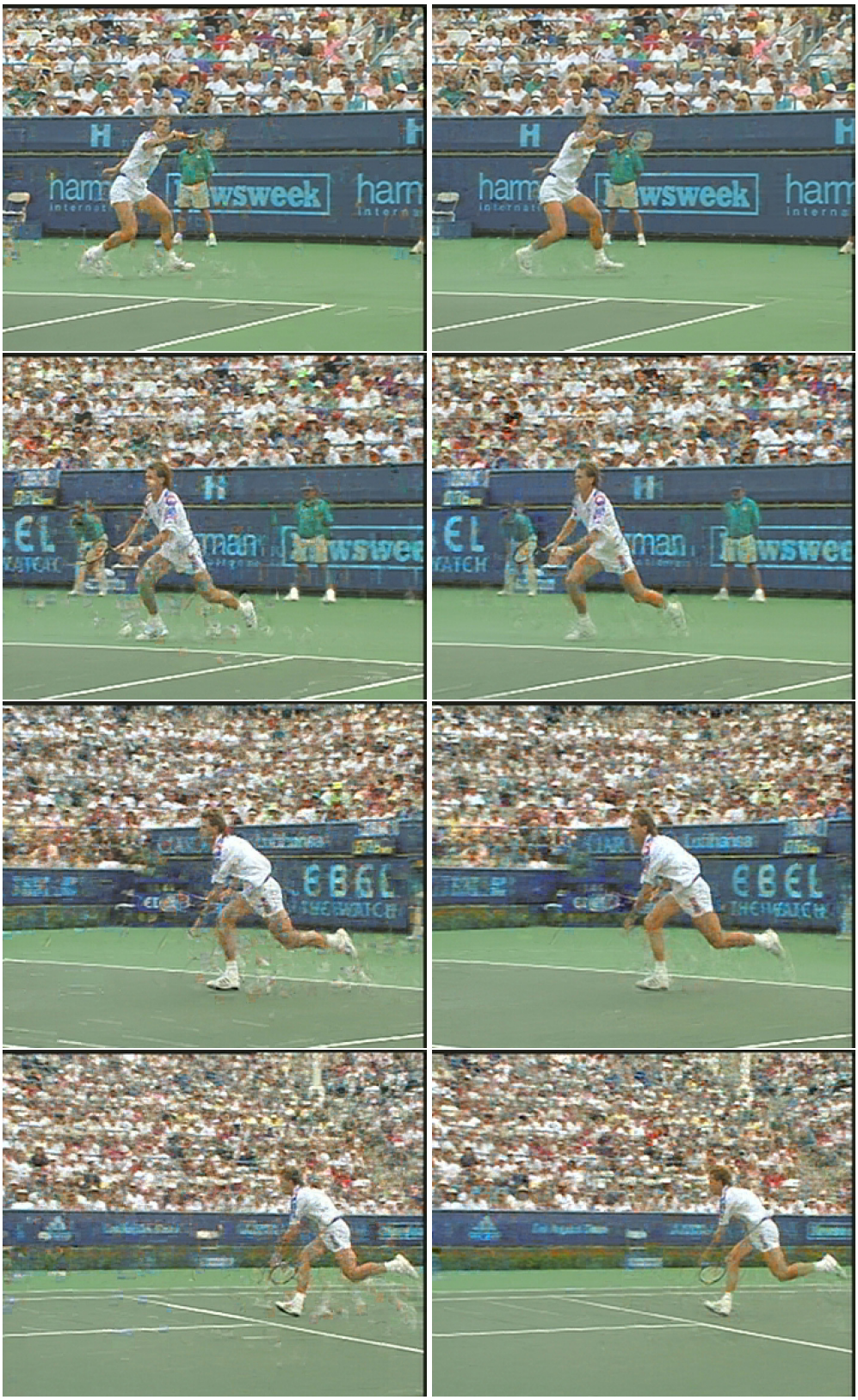


FIG. 4.15 – Images d'approximation du quatrième niveau issues de la décomposition temporelle de la séquence *Stefan* CIF 30 Hz obtenue avec le filtre 5/3 classique (à gauche) et avec le filtre 5/3 uniforme (à droite).

Transformée 5/3 classique	0	1	2
Niveau temporel 1	0.67	6.86	92.47
Niveau temporel 2	2.23	13.79	83.98
Niveau temporel 3	5.46	22.80	71.74
Niveau temporel 4	10.73	29.62	59.65

Transformée 5/3 uniforme	0	1	2
Niveau temporel 1	0.00	4.13	95.87
Niveau temporel 2	0.00	9.01	90.99
Niveau temporel 3	0.00	17.42	82.58
Niveau temporel 4	0.00	26.67	73.33

TAB. 4.11 – Pourcentage de pixels non-connectés (0), simplement connectés (1) et connectés bidirectionnellement (2) durant l'étape de mise à jour de la transformée 5/3 uniforme et de la 5/3 classique, à différents niveaux temporels. Ces résultats ont été obtenus sur la décomposition temporelle de la séquence *Foreman* CIF 30 Hz sur 4 niveaux.

optimisation débit-distorsion, codage CABAC, contrôle de débit et utilisation de 5 images de référence.

YSNR (en dB)	512 kbs	768 kbs	1024 kbs	1536 kbs
Filtre 5/3 uniforme	30.23	32.39	33.85	36.07
Filtre 5/3 classique optimisé	29.64	31.80	33.22	35.08
Filtre 5/3 sans mise à jour	28.82	31.04	32.55	34.86
H.264/AVC	29.90	31.88	33.68	35.27

TAB. 4.12 – Courbes de débit-distorsion obtenues pour différents filtres temporels et différents débits sur la séquence *Mobile* CIF 30 Hz.

YSNR (en dB)	512 kbs	768 kbs	1024 kbs	1536 kbs
Filtre 5/3 uniforme	32.04	33.82	35.03	36.99
Filtre 5/3 classique optimisé	31.44	33.23	34.41	36.36
Filtre 5/3 sans mise à jour	30.62	32.46	33.75	35.73
H.264/AVC	32.31	34.01	35.15	37.07

TAB. 4.13 – Courbes de débit-distorsion obtenues pour différents filtres temporels et différents débits sur la séquence *Tempête* CIF 30 Hz.

YSNR (en dB)	3000 kbs	6000 kbs
Filtre 5/3 uniforme	36.70	38.43
Filtre 5/3 classique optimisé	36.59	38.25
Filtre 5/3 sans mise à jour	35.59	37.14
H.264/AVC	36.20	37.80

TAB. 4.14 – Courbes de débit-distorsion obtenues pour différents filtres temporels et différents débits sur la séquence *City* 4CIF 60 Hz.

On observe les très bonnes performances du filtre 5/3 uniforme comparé au filtre 5/3 classique optimisé, avoisinant des gains de 0.5 dB en moyenne et atteignant jusqu'à 1 dB sur *Mobile* à 1536 kbs, pour une complexité équivalente. On remarque aussi les bons résultats que la transformée offre en comparaison du codec H.264/AVC, sachant que ce dernier n'offre aucune forme de scalabilité. Les simulations réalisées avec le filtre 5/3 sans mise à jour montrent de plus l'importance de cette étape en terme de PSNR dans la transformée temporelle. L'augmentation de l'efficacité objective de codage apportée par la transformée 5/3 uniforme est donc réelle, bien qu'elle n'ait pas été construite explicitement dans cette optique. Il est probable que les gains en PSNR observés soient dus à la meilleure qualité des images d'approximations obtenues au cours de la décomposition temporelle, augmentant ainsi mécaniquement l'efficacité de la prédiction temporelle dans les étages supérieurs.

4.2.5 Conclusion

En étudiant l'origine des artefacts fantômes apparaissant dans les séquences décodées à bas débit, nous avons été amenés à construire une transformée temporelle basée sur la transformée 5/3 dont les champs sont orientés dans la même direction : la transformée 5/3 uniforme. Cette construction particulière assure que tous les pixels soient connectés lors des étapes de prédiction et de mise à jour temporelle. Il en résulte un filtrage passe-haut et passe-bas plus homogène, conduisant à des sous-bandes temporelles d'approximation dépourvues d'artefacts. En suivant la même approche que celle suivie pour la transformée 5/3 classique, nous avons de plus construit un algorithme optimal d'estimation des champs de mouvement mis en jeu dans la transformée 5/3 uniforme, minimisant l'erreur de prédiction temporelle. On montre alors expérimentalement que la transformée temporelle 5/3 uniforme améliore nettement la qualité visuelle des sous-bandes d'approximation tout en augmentant l'efficacité globale de codage de façon significative.

4.3 Modération de la latence

Bien que la recherche de l'efficacité de codage soit primordiale dans la construction d'une transformée temporelle, elle ne doit pas masquer d'autres problématiques couramment rencontrées lors de la mise en situation *effective* d'un codec vidéo. En plus des contraintes matérielles sur la taille mémoire ou concernant la vitesse d'exécution à prendre en compte, il faut veiller à ce que la latence intrinsèque créée par un codec vidéo ne soit pas trop importante.

Après avoir défini précisément les notions de délais et de latence dans la section 4.3.1, nous justifions en section 4.3.2 pourquoi les différentes transformées temporelles de Haar, 5/3 et assimilées ne peuvent être utilisées dans des schémas de codage $t + 2D$ pour des applications de visioconférence ou de vidéosurveillance en temps réel, du fait de la latence trop importante qu'elles engendrent. Afin de pallier à ce problème, nous présentons en section 4.3.3 une méthode flexible et générique pour réduire la latence créée par une transformée temporelle. Des résultats expérimentaux décrits en section 4.3.4 et menés sur la transformée temporelle 5/3 montrent alors l'existence d'un compromis intéressant entre latence et efficacité de codage, dépendant des besoins de l'application visée.

Ces travaux ont conduit à la publication d'un premier article de conférence [104] où seul le délai d'encodage était considéré lors de la construction d'une transformée à délai

réduit. Un deuxième article [107] a alors complété ces travaux en envisageant tous les délais et en améliorant nettement l'efficacité de la transformée précédente.

4.3.1 Introduction, latence et délais

Dans une application de type visioconférence, le délai représente simplement le temps écoulé entre la capture d'une image côté émetteur et son affichage côté récepteur. Aussi appelée latence ou retard, le délai est dépendant de nombreux facteurs et se décompose en délai de transmission réseau, durée de traitement par le processeur, délai de paquetisation, de switching routeur... Certains de ces facteurs sont dépendants de l'architecture réseau et matérielle et peuvent être réduits (délai de transmission par le réseau, durée de traitement de calcul) tandis que d'autres sont intrinsèques à l'application et restent incompressibles.

Dans le cas de notre schéma de codage $t+2D$, les modules d'estimation de mouvement, de transformation spatiale et de codage entropique créent un délai qui n'est fonction que de la puissance de calcul du processeur. De nombreuses techniques d'optimisation logicielle, matérielle et algorithmique existent pour accélérer ces calculs mais sortent largement du cadre de ce chapitre.

Par contre, les transformées temporelles utilisées dans le schéma nécessitent généralement des images situées dans le futur ; elles sont donc non-causales et introduisent alors une certaine latence dans le codec. Cette dernière n'est pas compressible et ne dépend que de la fréquence de la vidéo et du nombre d'images situées dans le futur nécessaires à la transformation d'une image. Dans le cas des filtres temporels de type Haar ou 5/3, nous verrons que cette latence est suffisamment importante pour interdire l'utilisation d'un codec $t + 2D$ dans des applications de type visioconférence.

La problématique de la latence introduite par les filtres temporels 5/3 et 9/7 a été constatée en tout premier par Parisot [93, 94]. Dans ces articles, les auteurs insistent de plus sur l'optimisation de l'occupation mémoire nécessaire à l'implémentation matérielle de tels filtres. Ils dressent ainsi les tables relatives à la latence induite et à la mémoire requise par les filtres 5/3 et 9/7 mais ne proposent pas d'alternatives pour les réduire.

4.3.2 Analyse des délais créés par différents filtres temporels

Afin de se placer dans un cadre indépendant de tout réseau matériel et de tout contexte logiciel, nous considérons désormais que les délais de transmission et de temps de calcul sont *instantanés*. Sur la base de ces hypothèses, le délai D lié à une transformée temporelle n'est alors fonction que de la fréquence vidéo f exprimée en Hertz et d'un nombre N d'images et est noté $D = N \times f$.

Délai d'encodage

Le délai d'encodage $D_e = N_e \times f$ est la durée maximale nécessaire à la transformation d'une image courante en sa sous-bande correspondante. C'est donc le nombre maximal N_e d'images situées dans le futur nécessaire à la transformation de l'image courante. Nous avons représenté sur la Fig. 4.16 une analyse temporelle de type 5/3 sur 3 niveaux où le délai maximal N_e et les délais d'encodage de chaque image sont indiqués, exprimés en nombre d'images N. L'image d'indice 2 possède ainsi un délai d'encodage de 4, signifiant qu'elle nécessite 4 images dans le futur pour pouvoir être décomposée. Le

chemin en gras désigne le chemin de traitement lié à l'image ayant le plus grand délai d'encodage $N_e = 14$.

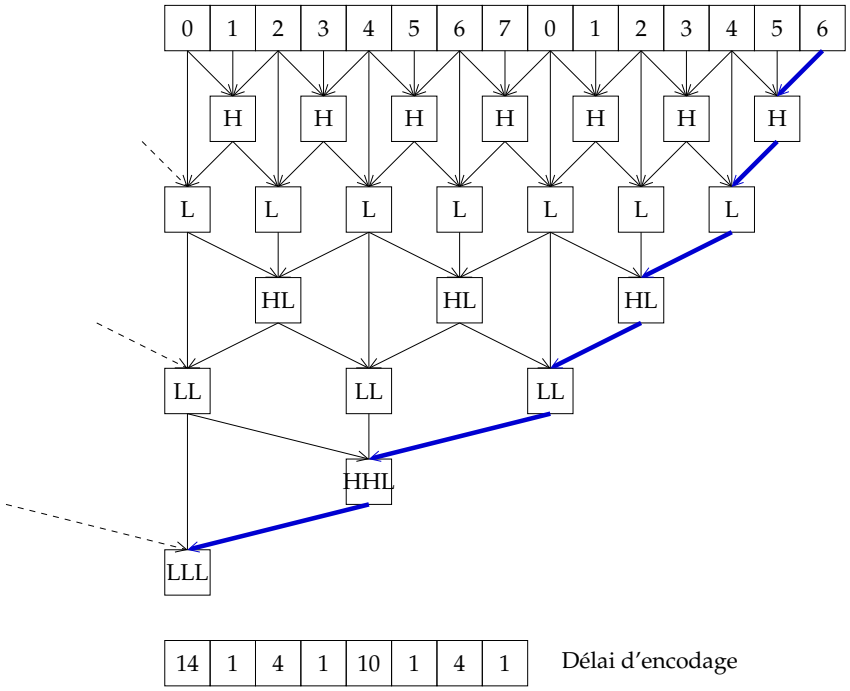


FIG. 4.16 – Délai maximal d'encodage d'une trame dans une analyse temporelle 5/3 à 3 niveaux. Le chemin en gras désigne le chemin de traitement lié à l'image ayant le plus grand délai d'encodage $N_e = 14$.

Délai de décodage et de reconstruction

Le délai de décodage $D_d = N_d \times f$ est la durée maximale nécessaire à la transformation inverse d'une sous-bande en sa trame correspondante. C'est donc le nombre maximal N_d de sous-bandes situées dans le futur nécessaire à la reconstruction de l'image courante.

Le délai de reconstruction $D_r = N_r \times f$ est la durée maximale nécessaire à la transformation d'une image courante en sa sous-bande correspondante. C'est donc le nombre maximal N_r d'images situées dans le futur nécessaire à la transformation *et* à la reconstruction de l'image courante. Le délai de reconstruction n'est pas égal à la somme du délai d'encodage D_e et du délai de décodage D_d car l'image possédant le délai maximal d'encodage n'est pas nécessairement celle qui possède le délai maximal de décodage. Il est aussi appelé délai point à point (*End-to-end delay*) dans la littérature. Ce délai a une signification précise dans le cadre d'une application de vidéoconférence en temps réel car il caractérise précisément le temps nécessaire à une image capturée du côté émetteur pour pouvoir être reconstruite par le récepteur.

Nous avons représenté sur la Fig. 4.17 une analyse temporelle de type 5/3 sur 3 niveaux où les délais maximaux N_d et N_r sont représentés. Les délais de décodage et de reconstruction de chaque image sont aussi indiqués, exprimés en nombre d'images N. L'image d'indice 0 nécessite ainsi 4 sous-bandes pour être reconstruite et donc 14 images futures vues du côté encodeur pour pouvoir être reconstruite. Le chemin en gras désigne

le chemin de traitement lié à l'image ayant le plus grand délai de décodage $N_d = 11$ et de reconstruction $N_r = 21$.

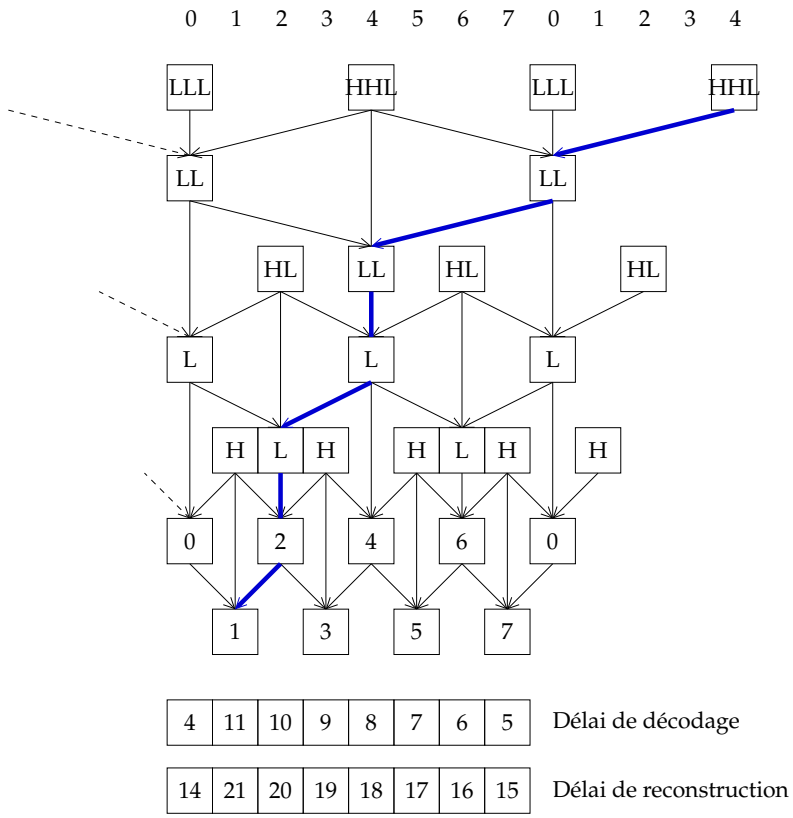


FIG. 4.17 – Délais maximaux de décodage et de reconstruction d'une trame dans une synthèse temporelle 5/3 à 3 niveaux. Le chemin en gras désigne le chemin de traitement lié à l'image ayant le plus grand délai de décodage $N_d = 11$ et de reconstruction $N_r = 21$.

En considérant une analyse temporelle sur N niveaux, il est possible de calculer les délais d'encodage, de décodage et de reconstruction introduits par divers filtres temporels. Nous nous intéressons dans les sections suivantes aux calculs de ces délais créés par les transformées temporelles les plus couramment utilisées.

Filtre temporel de Haar

Le filtre temporel de Haar est décrit par les équations suivantes :

$$h_t = x_{2t+1} - \mathcal{C}(x_{2t}, \boldsymbol{v}^+_{2t+1})$$
$$l_t = x_{2t} + \frac{1}{2}\mathcal{C}^{-1}(h_t, \boldsymbol{v}^+_{2t+1})$$

On peut montrer qu'une analyse temporelle de Haar sur N niveaux engendre alors les délais :

$$\begin{cases} N_e = 2^N - 1 \\ N_d = 2^{N-1} \\ N_r = 2^N - 1 \end{cases}$$

La suppression de l'étape de mise à jour du filtre de Haar conduit à un filtre purement causal, n'introduisant aucun délai $N_e = N_d = N_r = 0$. Cependant, comme vu précédemment dans la section. 3.2 du chapitre précédent, l'efficacité du filtre de Haar n'est pas satisfaisante. Nous verrons par la suite qu'il existe des alternatives plus intéressantes pour obtenir des filtres à délai faible ou même nul.

Filtre temporel 5/3

Nous rappelons les équations de filtrage temporel 5/3 :

$$h_t = x_{2t+1} - \frac{1}{2}\big(\mathcal{C}(x_{2t}, \boldsymbol{v}^+_{2t+1}) + \mathcal{C}(x_{2t+2}, \boldsymbol{v}^-_{2t+1})\big)$$
$$l_t = x_{2t} + \frac{1}{4}\big(\mathcal{C}^{-1}(h_{t-1}, \boldsymbol{v}^-_{2t-1}) + \mathcal{C}^{-1}(h_t, \boldsymbol{v}^+_{2t+1})\big)$$

Il est possible de montrer qu'une décomposition temporelle sur N niveaux, selon le filtre 5/3 classique ou selon sa variante 5/3 uniforme décrit dans la section 4.2, introduit les délais suivants :

$$\begin{cases} N_e = 2^{N+1} - 2 \\ N_d = 3 \times 2^{N-1} - 1 \\ N_r = 3 \times (2^N - 1) \end{cases}$$

Ainsi, l'utilisation d'une transformée 5/3 sur 4 niveaux temporels dans une application de visioconférence induirait un délai de reconstruction d'au minimum $D_r = N_r \times f = 1.5$ s ! Le seuil de confort visuel étant d'environ 300 ms, ce délai est ainsi bien trop important.

On remarque que la seule marge d'action pour réduire le délai dans une décomposition 5/3 consiste à diminuer le nombre de niveaux N de l'analyse temporelle, entraînant par conséquence une forte diminution des performances du codeur vidéo. Ceci motive ainsi la nécessité de construire un filtre avec une latence moindre.

De plus, on notera que la transformée temporelle 5/3 sans mise à jour utilisée par André [13] possède des délais plus faibles que la transformée 5/3 classique :

$$\begin{cases} N_e = 2^{N-1} \\ N_d = 2^N - 1 \\ N_r = 2^N - 1 \end{cases}$$

Comparé au filtre 5/3 classique, on observe une réduction d'un facteur 4 sur N_e, d'un facteur 3/2 sur N_d et d'un facteur 3 sur N_r. Bien qu'importante, cette diminution du délai reste toutefois insuffisante et ne permet pas d'assurer une latence compatible avec une application de type visioconférence.

En étendant le cas des filtres temporels dyadiques présentés ci-dessus, il est possible de calculer les délais introduits par les filtres temporels 3-bandes de Tillier [143, 144], dont un des intérêts réside dans leur aptitude à fournir des facteurs de scalabilité d'ordre 3.

Filtres temporels 3-bandes

Le filtre temporel 3-bandes le plus simple [143] est l'équivalent 3-bandes du filtre de Haar. Il est monodirectionnel, créé deux sous-bandes de détail et s'exprime sous la forme

lifting suivante :

$$\begin{aligned} h_t^+ &= x_{3t+1} - \mathcal{C}(x_{3t}, \boldsymbol{v}_{3t+1}^+) \\ h_t^- &= x_{3t-1} - \mathcal{C}(x_{3t}, \boldsymbol{v}_{3t-1}^+) \\ l_t &= x_{3t} + \frac{1}{4}\big(\mathcal{C}^{-1}(h_t^+, \boldsymbol{v}_{3t+1}^+) + \mathcal{C}^{-1}(h_t^-, \boldsymbol{v}_{3t-1}^+)\big) \end{aligned}$$

L'analyse temporelle sur N niveaux par un filtre Haar 3-bandes introduit les délais :

$$\begin{cases} N_e = (3^N - 1)/2 \\ N_d = (3^N - 1)/2 \\ N_r = (3^N - 1)/2 \end{cases}$$

L'introduction d'une prédiction bidirectionnelle permet alors d'obtenir un équivalent 3-bandes du filtre temporel 5/3 [144], s'exprimant par :

$$\begin{aligned} h_t^+ &= x_{3t+1} - \frac{1}{2}\big(\mathcal{C}(x_{3t}, \boldsymbol{v}_{3t+1}^+) + \mathcal{C}(x_{3t+2}, \boldsymbol{v}_{3t+1}^-)\big) \\ h_t^- &= x_{3t-1} - \frac{1}{2}\big(\mathcal{C}(x_{3t}, \boldsymbol{v}_{3t-1}^+) + \mathcal{C}(x_{3t-2}, \boldsymbol{v}_{3t-1}^-)\big) \\ l_t &= x_{3t} + \frac{1}{4}\big(\mathcal{C}^{-1}(h_t^-, \boldsymbol{v}_{3t+1}^-) + \mathcal{C}^{-1}(h_t^+, \boldsymbol{v}_{3t-1}^+)\big) \end{aligned}$$

Ce filtre engendre les délais suivants :

$$\begin{cases} N_e = 3^N - 1 \\ N_d = (3^N - 1)/2 \\ N_r = 3^N - 1 \end{cases}$$

Loin de réduire les délais de la transformée temporelle 5/3, ces filtres 3-bandes introduisent des retards nettement supérieurs à leurs homologues dyadiques pour un même nombre de niveaux temporels. Ils ne peuvent donc satisfaire nos contraintes de latence.

4.3.3 Construction d'un filtre temporel flexible à délai contraint

Plusieurs solutions comme la suppression drastique de l'étape de mise à jour [157] ou la combinaison étagée de plusieurs filtres temporels (5/3 et Haar) ont été proposées [53] mais se révèlent insuffisantes. En effet, elles permettent de réduire le délai mais ne sont pas suffisamment flexibles pour le diminuer à dessein voire l'annuler complètement.

Nous avons proposé un compromis simple [104] pour modérer ce retard à l'encodage, conduisant à un compromis entre efficacité de codage et contrainte de délai. Il consiste à supprimer la partie "en avant" des opérateurs de prédiction et de mise à jour des niveaux temporels les plus élevés et peut s'appliquer à n'importe quelle transformation temporelle. Cependant, ces travaux ne concernaient que la réduction du délai d'encodage et l'on observait une chute de performance importante en présence de contraintes de latence trop élevées. Afin de résoudre ces problèmes, nous avons poursuivi nos travaux et avons présenté [107] une transformée temporelle étagée composée de transformées élémentaires, afin d'obtenir un compromis souple entre latence et efficacité de codage.

Analyse temporelle 5/3 à délai contraint

Considérons les trois transformées élémentaires $T1$, $T2$ et $T3$ suivantes. La transformée $T1$ est la transformée $5/3$ classique ; elle possède la meilleure efficacité de codage mais introduit le plus grand retard. Elle est définie par :

$$\begin{aligned} h_t &= x_{2t+1} - \frac{1}{2}\big(\mathcal{C}(x_{2t}, \boldsymbol{v}^+_{2t+1}) + \mathcal{C}(x_{2t+2}, \boldsymbol{v}^-_{2t+1})\big) \\ l_t &= x_{2t} + \frac{1}{4}\big(\mathcal{C}^{-1}(h_{t-1}, \boldsymbol{v}^-_{2t-1}) + \mathcal{C}^{-1}(h_t, \boldsymbol{v}^+_{2t+1})\big) \end{aligned} \tag{T1}$$

Afin de réduire le retard introduit par la transformée élémentaire $T1$ et tout en gardant de bonnes propriétés de décorrélation, nous considérons une transformée $5/3$ dégénérée sans mise à jour en avant. En effet, nous avons mis en évidence dans la section 4.2.4 que la mise à jour n'a qu'une influence limitée sur l'efficacité de codage. Notons cette transformée $T2$:

$$\begin{aligned} h_t &= x_{2t+1} - \frac{1}{2}\big(\mathcal{C}(x_{2t}, \boldsymbol{v}^+_{2t+1}) + \mathcal{C}(x_{2t+2}, \boldsymbol{v}^-_{2t+1})\big) \\ l_t &= x_{2t} + \frac{1}{2}\mathcal{C}^{-1}(h_{t-1}, \boldsymbol{v}^-_{2t-1}) \end{aligned} \tag{T2}$$

La transformée $T2$ possède un retard deux fois moindre que la transformée $T1$ mais ne peut engendrer un retard nul. Pour atteindre ce but, nous devons continuer notre dégradation de la transformée $5/3$ et introduire alors une transformée sans prédiction en avant. De plus, comme le délai introduit par la prédiction en avant est plus important que celui induit par la mise à jour avant, nous devons supprimer la mise à jour avant. Le champ $\boldsymbol{v}^-_{2t+1}$ n'est alors utilisé que pour réaliser la mise à jour arrière, que nous décidons de supprimer pour économiser le coût de codage du champ et améliorer ainsi l'efficacité de codage. On obtient au final la transformée $T3$ suivante, qui est une forme dégénérée de la transformée de Haar sans mise à jour :

$$\begin{aligned} h_t &= x_{2t+1} - \mathcal{C}(x_{2t}, \boldsymbol{v}^+_{2t+1}) \\ l_t &= x_{2t} \end{aligned} \tag{T3}$$

Nous considérons alors une transformée temporelle étagée paramétrée par (P, Q) qui consiste à appliquer la transformée $T1$ sur les P premiers niveaux temporels, la transformée $T2$ sur les Q suivants et la transformée $T3$ sur les niveaux restants. Cette analyse temporelle que nous nommerons transformée (P, Q), introduit alors les délais suivants :

$$\text{Si } Q = 0 \begin{cases} N_e = 2^{P+1} - 2 \\ N_d = \lfloor 3 \times 2^{P-1} - 1 \rfloor \\ N_r = 3 \times (2^P - 1) \end{cases}$$

$$\text{Si } Q > 0 \begin{cases} N_e = 2^{P+1} + 2^{P+Q-1} - 2 \\ N_d = 2^{P+Q} - 1 \\ N_r = 2^{P+1} + 2^{P+Q} - 3 \end{cases}$$

Munis de ces relations, nous pouvons agir sur les paramètres P et Q de la transformée pour satisfaire un délai donné. Sachant qu'il peut y avoir plusieurs couples solutions satisfaisant un délai donné, nous choisirons les paramètres qui maximisent le nombre d'étapes de prédictions bidirectionnelles, c'est à dire ceux qui maximisent $P + Q$. On

pourra remarquer que la transformée $(P, Q) = (N, 0)$ correspond à une transformée 5/3 classique sur N niveaux et que la transformée $(P, Q) = (0, 0)$ est une analyse temporelle de Haar sans mise à jour, à délai nul. On notera de plus que les délais engendrés par la transformée (P, Q) sont indépendants du nombre de niveaux temporels N. Nous pouvons alors établir le Tab. 4.15 donnant les paramètres optimaux de la transformée (P, Q) permettant de satisfaire une contrainte de délai de reconstruction donnée.

Délai maximal de reconstruction	N_r	(P, Q) optimal
1500 ms	45	(4,0)
500 ms	15	(0,4)
300 ms	9	(1,2)
167 ms	5	(1,1)
100 ms	3	(0,2)
34 ms	1	(0,1)
0 ms	0	(0,0)

TAB. 4.15 – Paramètres optimaux (P, Q) pour satisfaire un délai de reconstruction pour une décomposition temporelle de 4 niveaux à la fréquence $f = 30$ Hz.

Ce tableau se lit comme suit : par exemple, pour obtenir une transformée (P, Q) ayant un délai de reconstruction maximal de 167 ms, valeur couramment utilisée dans les applications de visioconférence, il faut utiliser le couple de paramètres $(P, Q) = (1, 1)$. Ce choix correspond à une transformation étagée utilisant la transformée élémentaire $T1$ pour le premier niveau, la transformée $T2$ pour le second et enfin la transformée $T3$ pour les niveaux restants. Les structures correspondantes à la décomposition temporelle et à la reconstruction sur 3 niveaux sont illustrées par les Figs. 4.18 et 4.19 et sont à comparer avec les Figs.4.16 et 4.17, obtenues avec la transformée temporelle 5/3 classique. Par construction et comme attendu, on remarque que le délai de reconstruction n'est jamais supérieur à 5 trames.

4.3.4 Résultats expérimentaux

Nous avons construit dans la section précédente la transformée paramétrable (P, Q) à délai contraint. Cette réduction de délai a été rendue possible par l'utilisation de transformées élémentaires 5/3 dégénérées sans mise à jour avant ou mono-directionnelles, d'une efficacité moindre que la transformée 5/3. Il y a donc un compromis clair entre efficacité de codage et délai.

Afin d'apprécier les performances de la transformée temporelle (P, Q) en fonction d'une contrainte de délai, nous avons effectué des simulations de codage sur les séquences *Football* et *Tempête*. Les tableaux Tab. 4.16 et 4.17 présentent ces résultats exprimés en Y-PSNR, obtenus pour un ensemble de contraintes de délais et de débits.

Conformément aux attentes exprimées dans la section précédente, on observe que les meilleures performances sont atteintes dans le cas de la transformée 5/3 non-contrainte, à délai supérieur à 1500 ms. On note ensuite une dégradation légère en fonction du délai imposé, avec une perte de seulement 0.6 dB en présence d'un délai maximal de 150 ms. Cette dégradation est plus importante pour *Tempête* où la souplesse du mouvement pâtit de la perte des opérateurs de prédiction bidirectionnels dans les bas délais.

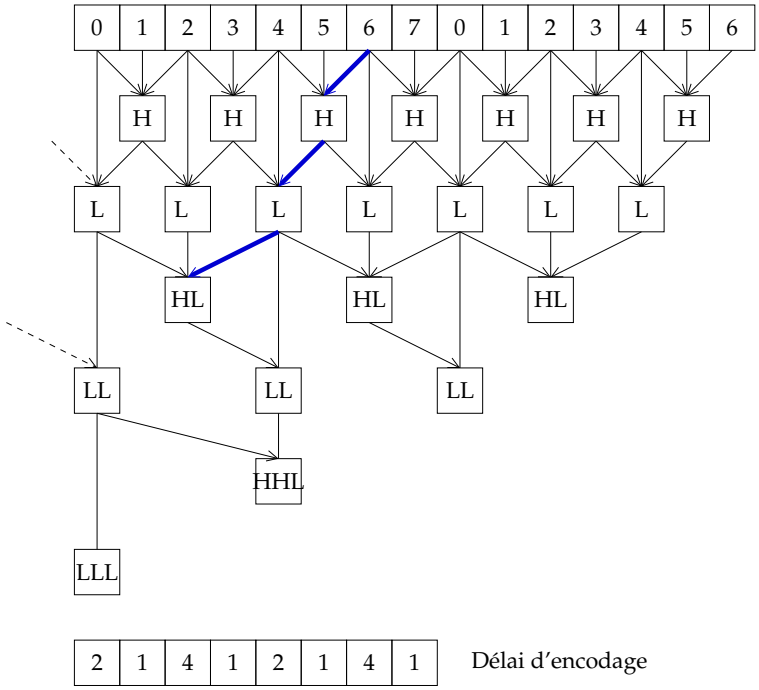


FIG. 4.18 – Délai maximal d'encodage d'une trame dans l'analyse temporelle à 3 niveaux par la transformée $(P, Q) = (1, 1)$.

Délai maximal	384 kbs	512 kbs	768 kbs	1024 kbs
1500 ms	27.15	28.49	30.32	31.79
300 ms	26.83	28.11	29.92	31.43
150 ms	26.51	27.68	29.44	30.90
35 ms	26.23	27.39	29.15	30.61
0 ms	25.94	27.08	28.79	30.30

TAB. 4.16 – Mesures de débit-distorsion obtenues avec la transformée temporelle (P, Q) pour plusieurs contraintes de délai de reconstruction et pour différents débits sur la séquence *Football* CIF 30 Hz.

Délai maximal	384 kbs	512 kbs	768 kbs	1024 kbs
1500 ms	29.72	31.00	32.77	33.95
300 ms	28.65	29.97	31.65	32.98
150 ms	27.90	29.23	30.84	32.20
35 ms	27.35	28.65	30.23	31.61
0 ms	26.91	28.11	29.64	31.02

TAB. 4.17 – Mesures de débit-distorsion obtenues avec la transformée temporelle (P, Q) pour plusieurs contraintes de délai de reconstruction et pour différents débits sur la séquence *Tempête* CIF 30 Hz.

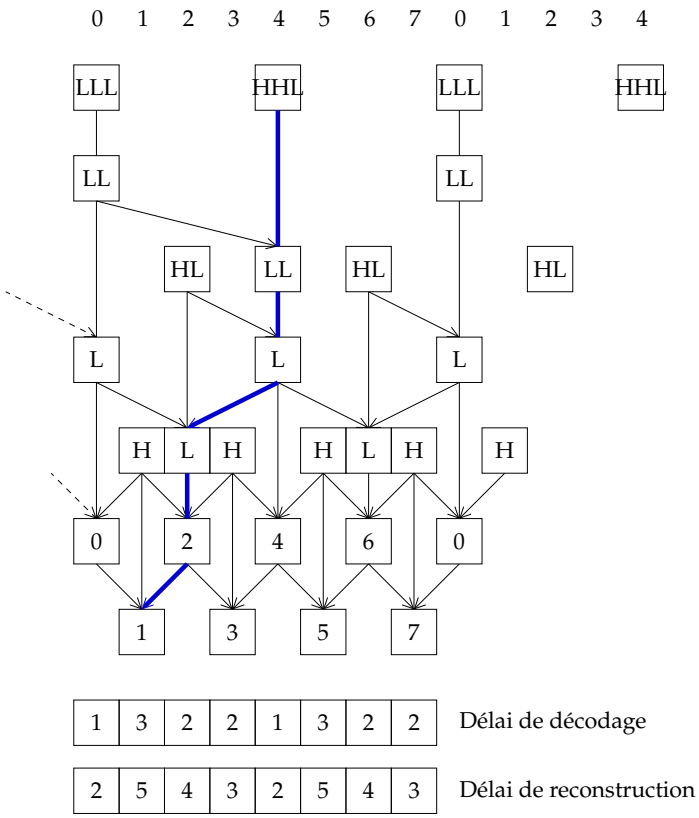


FIG. 4.19 – Délais maximaux de décodage et de reconstruction d'une trame dans la synthèse à 3 niveaux par la transformée $(P, Q) = (1, 1)$.

La transformée temporelle (P, Q) à délai contraint a de plus été implémentée conjointement par Viéron [158] au sein du codec SVC du groupe de normalisation MPEG et a donné des résultats similaires tout à fait satisfaisants. D'autres travaux ultérieurs sur la réduction du retard introduite par la transformée temporelle dans le codec SVC ont aussi été menés dans [124]. Les auteurs proposent de partitionner la décomposition d'un GOP en sous-ensembles indépendants, en contraignant chaque image de ces ensembles à ne jamais nécessiter plus de N images en avant pour être décomposée. Tout comme notre technique, cette proposition revient à couper les dépendances en avant lors de la décomposition temporelle. Cependant, elle possède l'inconvénient de créer des structures de prédiction irrégulières où, sur un même niveau temporel, les mêmes trames peuvent être prédites par une ou deux images et être mises à jour ou pas.

4.3.5 Conclusion

Nous avons présenté une transformée temporelle flexible, dont le délai est paramétrable en fonction des besoins d'une application. Les résultats expérimentaux ont montré l'existence d'un compromis intéressant entre délai et efficacité de codage. Ils ont de plus mis en évidence la dégradation modérée de l'efficacité de codage en fonction du délai, en partant du cas non-contraint au cas de délai nul. Entre ces deux extrêmes, un large éventail de compromis entre l'efficacité souhaitée et le délai maximal admissible existe, laissant le choix du filtre en fonction des besoins de l'application. Il est à noter que les mé-

thodes présentées ont été utilisées afin de réduire la latence des filtres temporels mis en jeu dans le schéma de codage $t+2D$. Cependant, des perspectives envisageables consisteraient à utiliser ces mêmes techniques pour abaisser le nombre total d'images nécessaires à la décomposition temporelle, afin de réduire la taille des mémoires tampons du codec. On pourrait par exemple considérer le nombres d'images dans le futur et *dans le passé*, nécessaires pour transformer une image courante, en raisonnant sur la taille du nombre d'images du tampon cyclique utilisé dans l'implémentation au fil de l'eau de la transformée temporelle.

4.4 Transformée Daubechies-4 compensée en mouvement

Nous avons vu dans la section 2.2.3 que les premiers codeurs vidéos scalables $t + 2D$ utilisaient une transformée temporelle de Haar par souci de simplicité. L'introduction du lifting temporel par Pesquet-Popescu [108] a alors permis l'utilisation de n'importe quelle décomposition temporelle même non-linéaire, tout en garantissant son inversibilité. De nombreuses transformées basées sur le filtre temporel $5/3$ compensé en mouvement ont pu alors être mises en œuvres et ont montré une efficacité de codage supérieure à la transformée de Haar.

Cependant, mis à part les schémas purement prédictifs UMCTF de Turaga [151], il n'existe pas dans la littérature de décomposition temporelle compensée en mouvement basée sur un filtre autre que celui de Haar ou que le filtre $5/3$. Le fait est singulier car il a été montré en codage d'image la nette supériorité de la transformée $9/7$ sur la transformée $5/3$. Est-il raisonnable de penser que l'augmentation de la taille du support d'un filtre temporel puisse améliorer son efficacité de codage ? C'est pour tenter de répondre à cette question que nous nous proposons dans cette section de mettre en œuvre une transformée temporelle compensée en mouvement basée sur l'ondelette Daubechies-4 .

4.4.1 Description et mise en œuvre

Nous souhaitons construire une transformée temporelle de support plus long que l'ondelette $5/3$ mais il n'est pas aisé de construire explicitement une structure lifting *ad-hoc* à trois étage. Nous nous proposons d'utiliser alors des structures déjà existantes, correspondant à des familles d'ondelettes connues.

Il existe au moins deux transformées en ondelettes dont la structure en lifting possède 3 étages : l'ondelette Daubechies-4 et l'ondelette biorthogonale $7/5$. La transformée en ondelettes Daubechies-4 est orthogonale, possède 2 moments nuls et un support de 4 points alors que la transformée CDF $7/5$ est biorthogonale, symétrique et possède 3 moments nuls. Poursuivant notre optique d'étudier le comportement d'une transformée temporelle de support un peu plus large que l'ondelette $5/3$, nous avons opté pour l'ondelette Daubechies-4. En effet, l'ondelette $7/5$ possède un support peut-être un peu trop grand pour une première approche et ceci peut être nuisible à la qualité de la prédiction temporelle en présence d'un mouvement trop rapide dans une séquence vidéo.

La transformation en ondelettes Daubechies-4 d'un signal mono-dimensionnel x_t peut s'exprimer sous forme lifting au moyen de trois opérateurs : un opérateur de prédiction P1, une mise à jour U et un autre opérateur de prédiction P2. Une mise à l'échelle des sous-bandes est alors effectuée par les étapes S1 et S2. La transformée de Daubechies-4

est illustrée en Fig. 4.20 et s'exprime sous forme lifting par les équations suivantes :

$$h_t^0 = x_{2t+1} - \sqrt{3}x_{2t} \tag{P1}$$

$$l_t^0 = x_{2t} + \frac{\sqrt{3}}{4}h_t^0 + \frac{\sqrt{3}-2}{4}h_{t+1}^0 \tag{U}$$

$$h_t^1 = h_t^0 + l_{t-1}^0 \tag{P2}$$

$$h_t = \frac{\sqrt{3}-1}{\sqrt{2}}h_t^1 \tag{S1}$$

$$l_t = \frac{\sqrt{3}+1}{\sqrt{2}}l_t^0 \tag{S2}$$

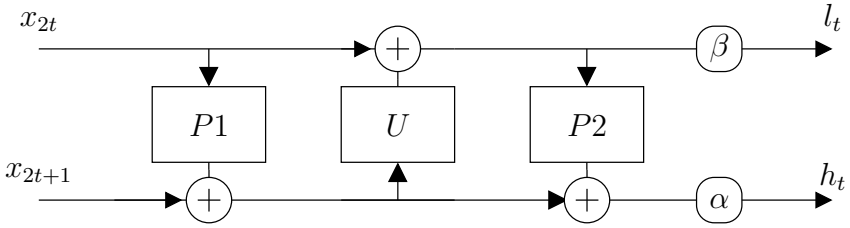


FIG. 4.20 – Structure lifting de la transformée en ondelettes de Daubechies-4

Le cadre général du lifting temporel développé par Pesquet-Popescu [108], rappelé dans la section 3.1.2, nous permet aisément de construire une transformée temporelle compensée en mouvement à partir de la formulation lifting d'une transformée monodimensionnelle. L'utilisation de l'opérateur de compensation de mouvement $\mathcal{C}$ introduit dans la section 3.1.3 nous permet alors de construire une transformée temporelle basée sur l'ondelette Daubechies-4 et s'exprimant sous la forme :

$$h_t^0 = x_{2t+1} - \sqrt{3}\,\mathcal{C}(x_{2t}, \boldsymbol{v}_0) \tag{P1}$$

$$l_t^0 = x_{2t} + \frac{\sqrt{3}}{4}\mathcal{C}(h_t^0, \boldsymbol{v}_1) + \frac{\sqrt{3}-2}{4}\mathcal{C}(h_{t+1}^0, \boldsymbol{v}_2) \tag{U}$$

$$h_t^1 = h_t^0 + \mathcal{C}(l_{t-1}^0, \boldsymbol{v}_3) \tag{P2}$$

$$h_t = \frac{\sqrt{3}-1}{\sqrt{2}}h_t^1 \tag{S1}$$

$$l_t = \frac{\sqrt{3}+1}{\sqrt{2}}l_t^0 \tag{S2}$$

La Fig. 4.21 illustre la décomposition d'un extrait de séquence vidéo sur un niveau en utilisant cette transformée temporelle. Comme on peut l'apercevoir, elle met en jeu quatre champs de vecteurs mouvement $\boldsymbol{v}_0$, $\boldsymbol{v}_1$, $\boldsymbol{v}_2$ et $\boldsymbol{v}_3$. Tout comme dans le cas des filtres de Haar ou 5/3 et comme abordé dans la section 3.1.2, il n'est pas souhaitable de les conserver tous. En effet, ils ont un coût de codage non-négligeable et il est préférable d'en estimer certains et en déduire les autres. La première étape de prédiction $P1$ nécessite un champ de mouvement avant $\boldsymbol{v}_0 = v_{2t+1}^+$, prédisant l'image x_{2t+1} par rapport à l'image x_{2t}. La mise à jour U met en jeu les deux champs $\boldsymbol{v}_1$ et $\boldsymbol{v}_2$. Le champ de mouvement $\boldsymbol{v}_1$ est l'opposé du champ $\boldsymbol{v}_0$ et peut se calculer par inversion en utilisant l'opérateur de compensation inverse $\mathcal{C}^{-1}$, comme dans le cas des filtres de Haar ou 5/3. Le champ de

mouvement arrière $\boldsymbol{v}_2$ prédit l'image x_{2t+3} par rapport à l'image x_{2t}, sur une distance de trois images. Ce champ est aussi l'opposé du champ $\boldsymbol{v}_3$, mis en jeu dans la prédiction $P2$: le champ $\boldsymbol{v}_3$ peut donc être calculé par inversion de $\boldsymbol{v}_2$.

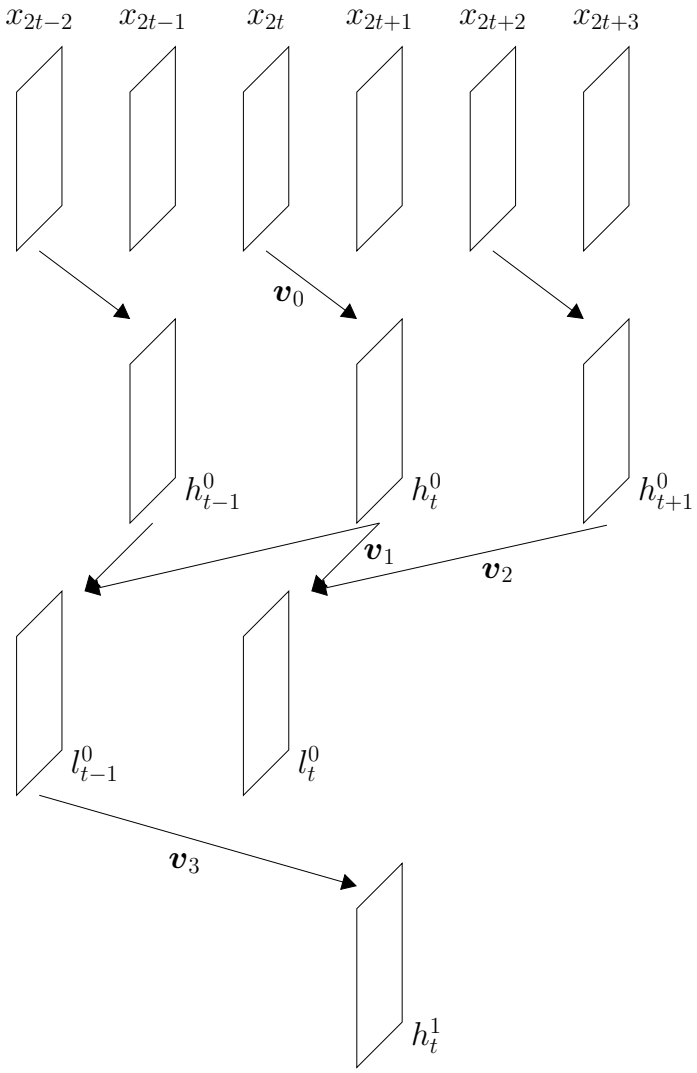


FIG. 4.21 – Décomposition temporelle en ondelettes Daubechies-4

Plusieurs stratégies sont donc possibles sur le choix des champs de mouvement mis en jeu dans la transformée de Daubechies-4. Afin de réduire la redondance de ces champs et de ne pas augmenter la complexité de notre prototype qui ne gère qu'au maximum deux champs de mouvement, nous choisissons de ne considérer que les champs avant $\boldsymbol{v}_0 = \boldsymbol{v}^1_{2t+1}$ et $\boldsymbol{v}_3 = \boldsymbol{v}^2_{2t+1}$, mis en jeu dans les prédictions $P1$ et $P2$. Seuls ces champs seront alors estimés et encodés dans le bitstream vidéo compressé. L'opérateur de compensation inverse $\mathcal{C}^{-1}$ permet d'obtenir les autres champs de mouvement $\boldsymbol{v}_1$ et $\boldsymbol{v}_2$ par inversion des champs $\boldsymbol{v}_0$ et $\boldsymbol{v}_3$, respectivement. L'opérateur de mise à jour de la transformée temporelle Daubechies-4 se réécrit alors :

$$l^0_t = x_{2t} + \frac{\sqrt{3}}{4}\mathcal{C}^{-1}(h^0_t, \boldsymbol{v}_0) + \frac{\sqrt{3}-2}{4}\mathcal{C}^{-1}(h^0_{t+1}, \boldsymbol{v}_3) \qquad \text{(U)}$$

La mise en œuvre efficace de la transformée temporelle Daubechies-4 nécessite une implémentation au fil de l'eau comme abordé dans la section 3.1.4 où les images sont décomposées et transformées à la volée. Cette implémentation repose sur un module où un buffer cyclique consomme deux nouvelles images et produit deux sous-bandes temporelles. Le module effectue alors un cycle en accomplissant les étapes décrites dans la Fig. 4.22, de façon similaire au module réalisant la transformée temporelle 5/3 de la Fig. 3.6.

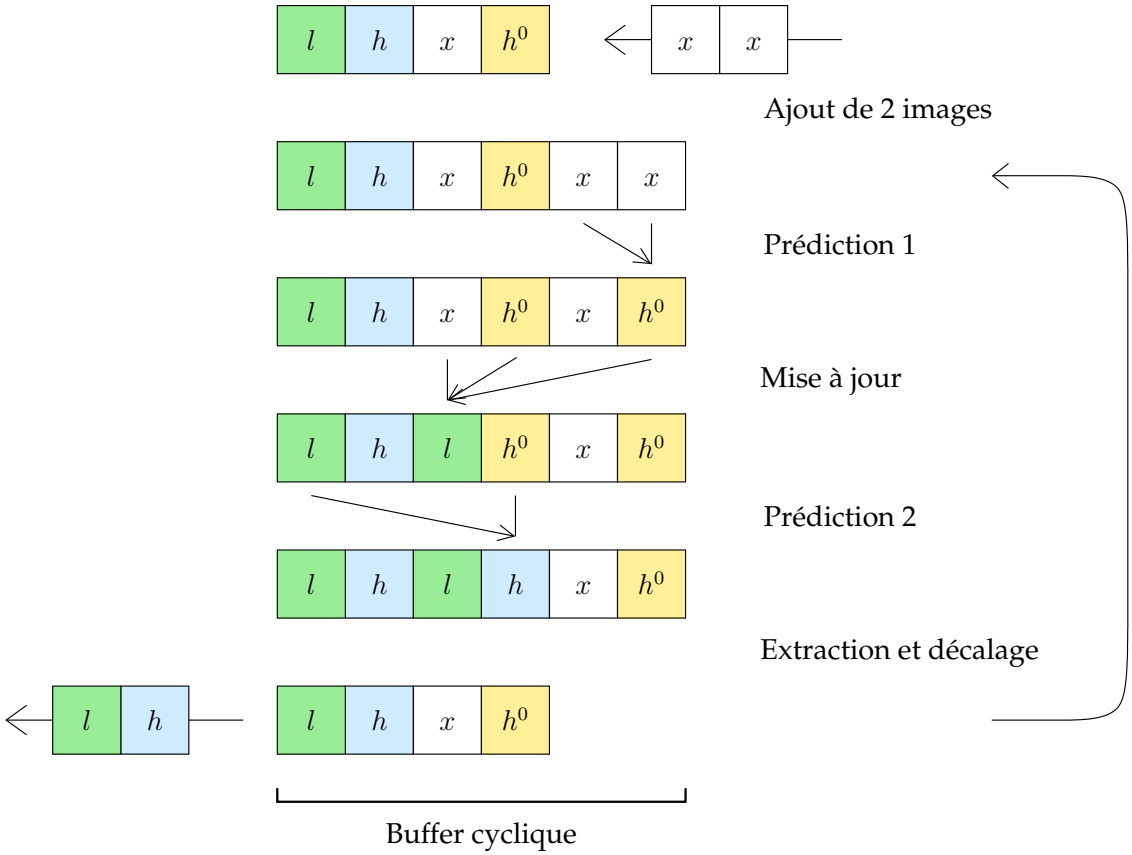


FIG. 4.22 – Schéma de fonctionnement du module de traitement au fil de l'eau de la transformée temporelle Daubechies-4. On y observe l'évolution du buffer cyclique de filtrage.

4.4.2 Résultats expérimentaux

Afin d'évaluer son efficacité de codage vidéo, nous avons intégré la transformée temporelle Daubechies-4 (D4) au sein de notre prototype sous la forme modulaire au fil de l'eau décrite précédemment. Les simulations ont été effectuées sur les séquences *Mobile* et *Foreman* CIF 30 Hz, en utilisant 3 niveaux de décompositions temporelles. Les résultats de simulation sont présentés dans les Tabs. 4.18 et 4.19, où ils sont comparés à des expérimentations de codage en conditions similaires effectuées avec les filtres temporels de Haar et 5/3.

On observe la performance médiocre réalisée par le filtre temporel D4. En effet, les résultats de codage observés restent inférieurs à ceux obtenus avec le filtre de Haar ou

YSNR (en dB)	384 kbs	512 kbs	768 kbs	1024 kbs	2048 kbs
5/3	25.30	27.32	**29.85**	**31.45**	**35.46**
Haar	**25.73**	**27.38**	29.66	31.15	35.06
D4	18.33	19.14	21.15	22.53	27.08

TAB. 4.18 – Mesures de distorsion obtenues en utilisant différents filtres temporels à différents débits sur la séquence *Mobile* CIF 30 Hz.

YSNR (en dB)	384 kbs	512 kbs	768 kbs	1024 kbs	2048 kbs
5/3	**32.76**	**33.85**	**35.27**	**36.50**	**39.61**
Haar	32.21	33.23	34.65	35.85	38.93
D4	22.09	23.02	24.19	25.45	28.74

TAB. 4.19 – Mesures de distorsion obtenues en utilisant différents filtres temporels à différents débits sur la séquence *Foreman* CIF 30 Hz.

le filtre 5/3. Il est possible que ce manque d'efficacité soit expliqué par les nombreux champs de mouvement nécessités par la transformation D4. Deux inversions de champs sont ainsi nécessaires pour effectuer l'opération de mise à jour, qui n'a pas la même signification que sa consœur dans le filtre 5/3. En effet et contrairement au filtre 5/3, l'opérateur U ajoute ici à la sous-bande l_t^0 les sous-bandes h_t^0 qui sont loin d'être des images peu énergétiques. Les erreurs de mouvement créées par la compensation inverse lors de la mise à jour ont alors une influence importante en créant de larges coefficients sur les sous-bandes de détail. Le surcoût nécessaire au codage de ces coefficients dégrade ainsi le rapport signal à bruit et diminue l'efficacité de la transformée temporelle D4. Enfin, la forte dissymétrie de l'ondelette Daubechies-4 est peut-être en cause dans le manque d'efficacité de la transformée temporelle D4. D'autres configurations de mouvement, l'utilisation d'un nombre supérieur de champs ou la mise en œuvre de la transformée 7/5 pourraient être envisagés afin d'augmenter l'efficacité de codage.

4.5 Conclusion

Nous avons présenté dans ce chapitre plusieurs stratégies d'optimisation de la transformée temporelle mise en jeu dans le schéma de codage $t + 2D$. En se basant sur sa structure lifting, nous avons poursuivi plusieurs axes de recherche visant à améliorer son efficacité de décorrélation temporelle.

Nous avons tout d'abord proposé un algorithme quasi-optimal d'estimation bidirectionnelle conjointe des champs de mouvement mis en jeu dans la transformée temporelle 5/3 compensée en mouvement. Cet algorithme est itératif, converge rapidement et permet la minimisation de la distorsion des sous-bandes temporelles de détail. Sa mise en place au sein du codec MC-EZBC conduit à un gain moyen en PSNR de plus de 1 dB par rapport à une estimation indépendante des champs de mouvement, pour une complexité équivalente.

Un inconvénient majeur de la transformée temporelle 5/3 est sa propension à créer des artefacts fantômes dans les séquences décodées à bas débits. Ces artefacts sont visuellement désagréables, complexifient le codage des images et sont liés à la présence de zones non-connectées durant l'étape de mise à jour temporelle. Afin de supprimer ces

artefacts, nous avons proposé une transformée 5/3 uniforme où les champs de mouvement sont orientés dans le même sens, empêchant la création de zones non-connectées. Comme précédemment, il est alors possible de concevoir un algorithme d'estimation bidirectionnelle conjoint des champs de mouvement mis en jeu dans la transformée 5/3 uniforme. Sa mise en œuvre expérimentale permet une réduction visible des artefacts présents dans les sous-bandes temporelles d'approximation et offre une efficacité de codage supérieure à la transformée 5/3 optimisée, surpassant même dans certains cas le codec H.264, pourtant non-scalable.

Les filtres temporels classiques introduisent un retard important dans le schéma de codage vidéo $t+2D$, prohibant leur utilisation pour des applications de visioconférence en temps réel. Après avoir étudié les causes de cette latence, nous avons présenté une transformée temporelle flexible basée sur le filtre 5/3, capable de respecter une contrainte de délai imposée. Elle consiste en la construction d'une analyse temporelle utilisant trois types de filtres temporels élémentaires. Les résultats expérimentaux montrent une faible dégradation du PSNR en fonction du délai imposé et concluent sur l'existence d'un compromis entre le délai imposé et l'efficacité de codage obtenue. Cette transformée flexible offre ainsi une large plage de possibilités, s'étalant du cas non-contraint au cas de délai nul, en fonction des besoins de l'application.

Enfin, conscients du gain important en efficacité de codage apporté par la transformée temporelle 5/3 comparée à la transformée de Haar, nous avons souhaité expérimenter un filtre compensé en mouvement plus long, basé sur l'ondelette Daubechies-4. Sa mise en œuvre au sein d'un schéma de codage $t+2D$ est facilitée par l'utilisation de la structure lifting et une implémentation au fil de l'eau. La transformée temporelle Daubechies-4 montre cependant une efficacité de codage inférieure aux filtres de Haar et 5/3, due probablement à une gestion complexe des champs de mouvement.