

Mouvement et vidéo : estimation, compression et filtrage morphologique

Nicolas Laveau

▶ **To cite this version:**

Nicolas Laveau. Mouvement et vidéo : estimation, compression et filtrage morphologique. Mathematics [math]. École Nationale Supérieure des Mines de Paris, 2005. English. NNT : . pastel-00003299

HAL Id: pastel-00003299
https://pastel.hal.science/pastel-00003299

Submitted on 24 Jan 2008

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Mouvement et Vidéo : Estimation, Compression et Filtrage morphologique

---

THÈSE

présentée à
l'École Nationale Supérieure des Mines de Paris
par

Nicolas Laveau

pour obtenir le titre de

DOCTEUR
en
MORPHOLOGIE MATHÉMATIQUE

Manuscrit
Présenté le 12 Décembre 2005 à Fontainebleau devant le jury composé de :

Dominique JEULIN	*Président*
Jan CORNELIS	*Rapporteur*
Philippe SALEMBIER	*Rapporteur*
Touradj EBRAHIMI	*Examinateur*
Frédéric FALZON	*Examinateur*
Christophe BERNARD	*Directeur de thèse*

Remerciements

À tout seigneur tout honneur, je tiens d'abord à remercier Christophe Bernard pour ces quatre années de travail : pour les nombreuses heures de travail passées ensemble, au Centre de Morphologie Mathématique d'abord, dans nos divers lieux de rencontre, pour les conseils prodigués qui ont fait de ce travail ce qu'il est, pour sa patience à m'écouter et à expliquer, pour m'avoir poussé à une certaine rigueur, scientifique, technique et enfin didactique pour la rédaction de ce document et pour tout ce qui, au-delà des aspects purement scientifiques, a permis une excellente relation entre nous.

Un grand merci à Fernand Meyer, qui m'a proposé immédiatement de venir effectuer ce travail au Centre de Morphologie Mathématique quand je lui ai mentionné mon envie d'obtenir un doctorat, ainsi qu'à Dominique Jeulin, qui apporte au suivi des doctorants sa rigueur et sa curiosité. Je tiens à remercier Philippe Salembier et Jan Cornelis d'avoir bien voulu examiner ce document et me donner leurs avis et leurs conseils pour son amélioration. Touradj Ebrahimi a volontiers accepté de faire partie de mon jury et je lui en suis redevable. Il en est de même pour Frédéric Falzon qui, de plus, m'a fourni un des thèmes d'étude de cette thèse. C'est un grand honneur pour moi d'avoir un tel jury.

Mes souvenirs vont aux différents partenaires du projet européen MASCOT, notamment Philippe Salembier, Peter Schelkens, Aljosha Smolic, Béatrice Pesquet-Popescu et surtout Henk Heijmans, irréprochable scientifiquement et humainement. J'ai pu parfois bénéficier de leurs conseils éclairés.

Au Centre de Morphologie Mathématique, je tiens aussi à remercier Étienne Decencière pour sa disponibilité pour discuter de toutes sortes de problèmes scientifiques.

Mais ces quatre ans à Fontainebleau n'ont pas été faits que de réflexions scientifiques. Je dois donc souligner l'importance de Catherine Moysan et de Marc Waroquier, pièces maîtresses du CMM (ex pour Marc), sans lesquels les chercheurs seraient (à mon avis) peu de choses. En particulier, le sourire de Catherine, son humour et sa gentillesse m'ont mis un peu de baume au coeur dans les journées de doute qui ont émaillé mon travail. Pour l'informatique, je dois remercier Michel Bilodeau, Beatriz Marcotegui (xlim3d) ainsi que Raffi Enficiaud et Romain Lerallut (Morphée) pour avoir accepté de longues discussions où j'ai pu montrer mon talent à couper les cheveux en quatre et Petr Dokladal pour ses nombreux prêts de tournevis. Pour ma présentation de soutenance, je tiens à signaler l'aide et les conseils généreusement fournis par Aurélien Cord, Raffi, Romain et Thomas Retornaz.

Les journées de recherche à Fontainebleau sont émaillées de pauses, le temps pour les neurones de refroidir. Je garde une grande reconnaissance à tous ceux qui sont passés régulièrement dans mon bureau pour empêcher les miens de griller : Gabriel Fricout, Raffi, Caroline Bernard-Michel, Romain, Christophe Aug, Pierre Mahé, Thomas R. sans oublier Mathilde Boehm et Jesús Angulo qui ont partagé mon bureau. Fontainebleau étant loin de Paris par les transports en commun, j'ai eu l'opportunité de discuter de

nombreuses heures avec Caroline, Alexandre Égreteau, François Schaub : ce furent de précieuses heures. Je salue aussi Timothée, Thibauld, Jaromir, Costin, Thomas W., Eva, Franck, Martial, Hugues et François P. (et tout ceux que j'ai malencontreusement oubliés mais qui, je l'espère, ne m'en tiendront pas rigueur).

Last but not least, un très grand merci à Alexandre Hamburger et Marta Di Pietro pour leur soutien inconditionnel et à mes parents qui, en plus, ont lu, relu et encore relu ce manuscrit pour y traquer les fautes.

Je dédie ce travail à mes grands-parents, Julia et Charles, Andrée et Marcel, partis trop tôt pour que j'aie la chance de les apprécier.

Abstract

The PhD work developped in this document deals with the treatment of video sequences. This includes video compression for most of this thesis, but also spatio-temporal filtering and video segmentation. One of the recurrent analysis tools for each of these applications is motion measurement, that is the description of temporal coherence in a video sequence.

A video compression system generally includes the three following components :

- motion estimation
- temporal and spatial transforms
- coefficient quantization and encoding

We focus on each of these components successively.

In a first time, we try to adapt a motion estimation scheme by optical flow projection of a complex-valued wavelet basis to a video compression use. The resulting field being dense and noise-sensitive, we introduce in the resolution a regularizing element in order to reduce its coding cost. In spite of a clear improvement brought by our modifications, the motion estimation scheme by projection on a wavelet basis is not competitive in comparison to block-matching which is the reference algorithm for video compression. It illustrates that the choice and the design of a scheme is tighly linked to the use for which it is intended : motion estimation schemes optimized for applications so diverse as video compression, filtering and segmentation or even 3D scene analysis are unlikely to be the same. In these experiments, we have used a motion measurement scheme which tries to optimize a criterion formally equivalent to a matching criterion for video compression on theoretical conditions which are not met in general. Such an approach is thus clearly sub-optimal.

Fortified of this observation, we have then developped another motion estimation scheme which relies on a piecewise bilinear motion field parametrization and which this time directly minimize the mean square error which is our evaluation criterion. We prove that it is possible to obtain good results when motion parameters are sparse.

In video coding with temporal prediction, we need to encode heterogenous data such as motion fields or error prediction pictures. We have worked on rate allocation among error frames and more moderately between an error frame and a motion field. We have adapted a rate planification model introduced by Mallat and Falzon which was initially designed for still images and which is currently used in flow compression of satellite pictures. This approach proves to be better than others more classicaly used in video compression.

To be able to perform a transform coding of motion fields and error frames, we tried to design new non-linear subband transform. With this intention, we have used the lifting

scheme which insures the formal invertibility of the achievable transforms, whether these are linear or not. We have designed two new non-linear decompositions.

The first one aims at reducing an artifact commonly called Gibbs' effect. This first decomposition consists in using a Deslauriers-Dubuc' predictor modified so as to reduce these artifacts. Our modification allows to reduce the ringing effect around the discontinuities at the moderate cost in terms of representation efficiency in the regular sections of the signal. The formulation avoids the filter-switching mechanism which is quite commonly used in this kind of approaches by using continuous operators such as min or max, so as to insure the transform continuity and thus its stability after quantization.

The second one tries to improve the motion field wavelet decomposition by using the information each of its components gives on the other one. Indeed, our intuition leads us to believe that discontinuities are occurring at the same positions in both of its components. We take advantage of this fact to choose the prediction and update filters.

In the two cases, the designed methods give encouraging results on synthesis signals but their efficiency is lessen by using them on real data. One of the main difficulties is to design an update step in the lifting scheme. Moreover, the most efficient linear scheme is a 4-step scheme for which it is difficult to design a correspondant non-linear step since its properties are not easily read in the indivual steps of the lifting scheme.

Lastly, we have transposed ideas from video compression to design morphological filtering operating on video sequences, which integrate the motion estimation by using structuring elements following the motion. The application of these ideas gives encouraging results in filtering and segmentation, in particular due to the strong spatio-temporal correlation introduced in the neighbourhoods : this approach leads to more stable segmentations since it imposes a much stronger correlation between region borders than temporally iterative schemes. We discuss then the possibilities of using sub-pixel accurate motion fields.

Keywords Video compression, motion estimation, quantization, rate allocation, wavelet transform, lifting scheme, mathematical morphology, spatio-temporal segmentation.

Résumé

Le travail de thèse développé dans ce mémoire porte sur le traitement des séquences vidéos. Ceci inclut la compression pour une grande partie de la thèse, mais également le filtrage spatio-temporel et la segmentation vidéo. Un des outils d'analyse récurrent dans chacune de ces applications est la mesure du mouvement, c'est-à-dire la description de la cohérence temporelle d'une séquence vidéo.

Un système de compression vidéo comprend généralement les trois composantes suivantes :

- estimation du mouvement,
- transformations temporelle et spatiale,
- quantification et codage des coefficients.

Nous nous intéressons successivement à chacune de ces composantes.

Dans un premier temps, nous essayons d'adapter une méthode d'estimation par projection du flot optique sur une base d'ondelettes à valeur complexe à la compression vidéo. Le champ obtenu étant dense et sensible au bruit, nous introduisons dans la résolution un élément de régularisation afin de diminuer son coût de codage. En dépit d'une nette amélioration apportée par nos modifications, la technique d'estimation par projection sur une base d'ondelettes n'est pas compétitive face au block-matching qui constitue l'algorithme de référence pour la compression vidéo. Cela illustre bien le fait que le choix et la conception d'une méthode sont étroitement liés à l'usage qui en est fait : des méthodes d'estimation de mouvement optimisées pour des applications aussi diverses que la compression vidéo, le filtrage et la segmentation, ou encore l'analyse de scènes 3D ont peu de chances d'être les mêmes. Dans ces expériences, nous avons utilisé une méthode visant à satisfaire un critère qui est équivalent à un critère d'appariement optimal pour la compression sous des conditions théoriques qui ne sont en général pas vérifiées. Une telle approche est donc visiblement sous-optimale.

Forts de cette observation, nous avons ensuite développé une méthode de mesure de mouvement qui repose sur une paramétrisation du champ bilinéaire par morceaux, et qui minimise cette fois directement l'erreur quadratique moyenne qui est notre critère d'évaluation. Nous montrons qu'il est possible d'obtenir de bons résultats quand les paramètres du champ sont épars.

Un codage vidéo avec une prédiction temporelle suppose de coder des données aussi hétérogènes que des champs de mouvement ou des images d'erreur. Nous avons travaillé sur l'allocation de débit entre images d'erreur et de manière moins approfondie entre image d'erreur et champ de mouvement. Nous avons adapté un modèle de planification de débit introduit par Mallat et Falzon qui a été initialement conçu pour des images statiques et qui est actuellement utilisé pour la compression au flot d'images satellitales.

Cette approche se révèle meilleure que des approches plus classiquement utilisées en compression vidéo.

Pour pouvoir effectuer un codage par transformée du champ et des images d'erreur, nous avons cherché à concevoir de nouvelles transformations en sous-bandes non-linéaires. Pour cela, nous avons utilisé le schéma de *lifting*, qui garantit l'inversibilité formelle des transformations qu'il peut réaliser, que celles-ci soient linéaires ou non. Nous avons construit deux nouvelles décompositions non-linéaires.

La première vise à réduire un artéfact communément appelé effet de Gibbs. Cette première décomposition consiste à utiliser un prédicteur de Deslauriers-Dubuc modifié de manière à réduire ces artéfacts. La modification introduite permet effectivement de réduire les oscillations autour de discontinuités en échange d'un surcoût modeste en terme d'efficacité de représentation dans les sections régulières du signal représenté. La formulation évite le mécanisme de transition d'un filtre à l'autre relativement habituel dans ce genre d'approche en recourant à des opérateurs continus de type min et max, qui permettent de garantir la continuité de la transformation et donc sa stabilité après quantification.

L'autre se propose d'améliorer la décomposition en ondelettes du champ de mouvement en utilisant l'information qu'apporte chacune de ses composantes sur l'autre. En effet, l'intuition nous incite à penser que les discontinuités sont présentes au même endroit dans chacune des composantes du mouvement. Nous nous servons de cette co-occurrence des discontinuités pour choisir le filtre de prédiction.

Dans les deux cas, les méthodes mises au point donnent des résultats positifs sur des signaux de synthèse mais perdent en efficacité sur des signaux réels. Une des grandes difficultés est de mettre au point un étage de mise-à-jour dans le schéma de *lifting*. Par ailleurs, le schéma linéaire le plus efficace est un schéma à 4 étages pour lequel il est difficile de concevoir un concurrent non-linéaire dans la mesure où ses propriétés sont difficilement lisibles sur les étages individuels du schéma de *lifting*.

Nous avons enfin transposé des idées rencontrées en compression vidéo pour définir des opérations de filtrage morphologique vidéo intégrant la mesure du mouvement, utilisant des éléments structurants qui suivent le mouvement. L'application de ces idées donne des résultants probants en filtrage et en segmentation, en particulier grâce à une forte cohérence spatio-temporelle introduite dans les voisinages : cette approche donne des résultats de segmentation plus stables puisqu'elle impose une cohérence temporelle beaucoup plus forte aux frontières des régions que les méthodes itératives en temps. Nous discutons ensuite des possibilités d'utilisation de champs de mouvement à précision sous-pixellique.

Mots-clés Compression vidéo, estimation du mouvement, quantification, allocation de débit, transformation en ondelettes, schéma de *lifting*, morphologie mathématique, segmentation spatio-temporelle.

Table des matières

Introduction **3**

I. Préliminaires **9**

1. Ondelettes **11**
1.1. Ondelettes "classiques" . . . 11
1.1.1. Fréquence locale . . . 11
1.1.2. Représentation temps-fréquence et temps-échelle . . . 12
1.1.3. Transformée en ondelettes continue . . . 14
1.1.4. Transformée en ondelettes discrète . . . 14
1.1.5. Analyses multi-résolutions . . . 15
1.1.6. L'algorithme de la transformée en ondelettes rapide . . . 19
1.1.7. Les ondelettes orthogonales . . . 19
1.1.8. Décroissance des coefficients et régularité . . . 21
1.2. Le schéma de *lifting* . . . 22
1.2.1. Principe général . . . 22
1.2.2. Exemples linéaires . . . 23
1.2.3. Intérêts du schéma de *lifting* . . . 25
1.2.4. Conception d'ondelettes non-linéaires . . . 26
1.3. Conclusion . . . 32

2. L'estimation du mouvement **33**
2.1. L'équation du flot optique . . . 33
2.2. L'estimation par appariement . . . 35
2.2.1. Différence de régions déplacées . . . 35
2.2.2. Le block-matching simple . . . 36
2.2.3. Amélioration du block-matching . . . 36
2.3. Horn et Schunk . . . 37
2.4. Techniques fréquentielles . . . 38
2.5. Lucas et Kanade . . . 39
2.6. Approches multi-échelles . . . 39
2.7. Flot optique sur base d'ondelettes . . . 40
2.8. Conclusion . . . 41

II. Estimation **43**

3. Compression vidéo avec mesure du mouvement en ondelettes 45
3.1. Schéma de codage de référence . . . 45
3.1.1. Quantification . . . 45
3.1.2. Rapport signal sur bruit . . . 46
3.1.3. Entropie . . . 47
3.1.4. Modèle simplifié de codage . . . 47
3.2. Intérêt a priori de la régularisation . . . 48
3.2.1. Sensibilité au bruit . . . 48
3.2.2. Réduction de l'entropie du champ de mouvement . . . 49
3.2.3. Précautions à prendre . . . 50
3.3. Méthode de régularisation . . . 51
3.3.1. Reformulation du problème de facon générale . . . 51
3.3.2. Application au cas du champ de vecteurs . . . 54
3.3.3. Détail du protocole de tests et résultats . . . 55
3.4. Conclusion . . . 62

4. Estimation du mouvement par détermination de nœuds d'une grille 65
4.1. Représentation du mouvement par déformation d'une grille . . . 65
4.1.1. Formulation du problème . . . 65
4.1.2. Méthode de résolution . . . 67
4.1.3. Interpolation de l'image et de sa dérivée . . . 69
4.1.4. Résolution multi-échelles . . . 70
4.2. Résultats . . . 74
4.2.1. Comparaison avec le block-matching . . . 74
4.2.2. Résultats au sein du codeur MC-EZBC . . . 82
4.3. Conclusion . . . 84

III. Compression 87

5. Contrôle de débit et compression vidéo hybride 89
5.1. Le rôle du contrôle de débit . . . 89
5.2. Compression par codage avec quantification . . . 90
5.2.1. Compression et taux de distortion . . . 90
5.2.2. Quantification haute-résolution . . . 91
5.3. Des algorithmes de contrôle de débit pour un standard de compression bien connu . . . 93
5.3.1. TMN.8 (H.263) . . . 93
5.3.2. He et Kim (H.263) . . . 98
5.3.3. Algorithme de Mallat et Falzon . . . 101
5.4. Mise en oeuvre . . . 105
5.4.1. Choix du codeur . . . 105
5.4.2. Algorithme de He-Kim . . . 105
5.4.3. Algorithme de Mallat et Falzon . . . 106

5.4.4. Complexité de calcul . . . 107
5.4.5. Conclusion . . . 109
5.5. Résultats et Analyse . . . 109
5.5.1. TMN.8 . . . 109
5.5.2. Tests effectués . . . 109
5.5.3. He-Kim . . . 112
5.5.4. Mallat-Falzon . . . 114
5.5.5. Conclusion . . . 120
5.6. Perspectives . . . 120
5.6.1. Adaptation de l'algorithme dans H264 . . . 120
5.6.2. Conclusion . . . 122

6. Transformation jointe des deux composantes du champ de vecteurs 123
6.1. Entropie et information mutuelle . . . 123
6.1.1. Quelques définitions . . . 123
6.1.2. Information d'une composante de mouvement sur l'autre . . . 125
6.2. Schéma de *lifting* conjoint . . . 127
6.2.1. *lifting* joint pour données entières . . . 127
6.2.2. *lifting* joint sur des réels . . . 131
6.2.3. Asymétries du *lifting* joint . . . 132
6.3. Résultats . . . 132
6.3.1. Critère d'évaluation . . . 133
6.3.2. Test sur un champ synthétique . . . 133
6.3.3. Tests sur les données réelles . . . 134
6.4. Conclusion . . . 136

7. Ondelettes sans rebonds 137
7.1. Motivation . . . 137
7.2. Prédiction encadrée . . . 138
7.2.1. Prédicteur de Deslauriers-Dubuc . . . 138
7.2.2. Encadrement d'ordre 0 . . . 138
7.2.3. Encadrement d'ordre 1 . . . 140
7.2.4. Analyse du comportement des ondelettes sans rebonds . . . 140
7.2.5. Formulation par médian d'intervalles . . . 143
7.2.6. Résultats . . . 144
7.3. Ondelettes sans rebonds en 2 dimensions . . . 149
7.3.1. Conception . . . 149
7.3.2. Résultats . . . 152
7.4. Conclusion . . . 153

IV. Filtrage morphologique 155

8. Morphologie Mathématique et Vidéo **157**
8.1. Éléments structurants suivant le mouvement (ESM) 158
8.1.1. Des idées venant d'autres domaines 158
8.1.2. Définition des éléments structurants suivant le mouvement 159
8.1.3. Quelques considérations pratiques 160
8.2. Modèles de mouvement adaptés aux ESM 168
8.2.1. Mouvement par objet 168
8.2.2. Étude des discontinuités 169
8.3. Implantation 169
8.4. Expériences 170
8.5. ESM et champs de mouvement à précision sous-pixellique 171
8.5.1. Interpolation et Morphologie 171
8.5.2. Interpolation directe des voisins 179
8.5.3. Sur-échantillonnage de l'image 182
8.5.4. Utilisation de la partie sous-pixellique du mouvement pour déformer avec soin 184
8.6. Conclusion 185

Conclusion **189**

Bibliographie **192**

Table des figures

1.1. Pavages temps–fréquence associés aux représentations de Dirac et de Fourier 13
1.2. Pavages du plan temps–fréquence pour la représentation temps–fréquence et la représentation temps–échelle 13
1.3. Transformée en ondelettes rapides 20
1.4. Schéma de principe du *lifting*. 23
1.5. Schéma de principe du *lifting* correspondant aux ondelettes biorthogonales 7/9 25
1.6. Spectre des ondelettes 2D classiques 28
1.7. Grilles en quinconce : position des échantillons 29
1.8. Grilles en quinconce : représentation compacte 29
1.9. Schéma de principe du *lifting*, avec étape de prédiction non-linéaire. 31
1.10. Schéma de principe du *lifting*, avec étape de mise à jour en premier, puis avec une étape de prédiction non-linéaire. 32

2.1. Illustration du problème d'ouverture 34
2.2. Illustration du problème d'aliasage 35
2.3. Illustration de la mesure du flot optique par technique fréquentielle 38

3.1. Incertitudes selon la texture 49
3.2. Recherche de solutions plus cohérentes au sein des ellipses d'incertitudes . 50
3.3. Schéma de principe de la simulation de codage 56
3.4. PSNR en fonction de $\mathcal{H}_{total}$, comparaison de l'influence de n_{spl} 57
3.5. PSNR en fonction de $\mathcal{H}_{total}$, comparaison des couples $(\lambda/\lambda_0, n_{reg})$ 58
3.6. PSNR en fonction de $\mathcal{H}_{total}$, comparaison de l'influence de n_{reg} pour $\lambda/\lambda_0 = 0.25$. 59
3.7. PSNR en fonction de $\mathcal{H}_{total}$, comparaison de l'influence de n_{reg} pour $\lambda/\lambda_0 = 0.125$. 59
3.8. $\mathcal{H}_{err}$ en fonction de $\mathcal{H}_v$ 60
3.9. $\mathcal{H}_{total}$ en fonction de $\mathcal{H}_v$ 61
3.10. PSNR en fonction de $\mathcal{H}_{total}$, comparaison de l'influence de Δ_v 61

4.1. Différentes fonctions ϕ_{00} possibles. 66
4.2. Grilles de points $\mathbf{x_{ij}}$ associées aux fonctions ϕ_{00} 66
4.3. Spectres de filtrage. 73
4.4. PSNR au cours du temps, *Foreman* 6 échelles 76
4.5. PSNR au cours du temps, *Mobile* 6 échelles 77
4.6. PSNR au cours du temps, *Bus*, 6 échelles 78

4.7. PSNR au cours du temps, *Paris*, 6 échelles 79
4.8. PSNR au cours du temps, *Football* 6 échelles 80
4.9. PSNR au cours du temps, *Foreman* 4 échelles 81
4.10. Repérage des différents types de pixels pour le MCTF 82
4.11. Comparaison MC-EZBC+HVSBM contre MC-EZBC+BBL 85
4.12. Comparaison MC-EZBC+HVSBM contre MC-EZBC+BBL 85
4.13. Comparaison MC-EZBC+HVSBM contre MC-EZBC+BBL 86

5.1. Schéma du tampon 95
5.2. Courbe $R(\vartheta, \rho)$ pour A : $\rho = 0.96$, B : $\rho = 0.97$, C : $\rho = 0.98$ 113
5.3. Remplissage du tampon - *Foreman*@48kbps 118
5.4. Remplissage du tampon - *Salesman*@32kbps 119
5.5. Modification de la courbe de probabilités d'apparition des valeurs du gradient du vecteur de mouvement pour diminuer l'entropie du champ. 121

6.1. Co-histogrammes de champs de mouvement 126
6.2. Comparaison des composantes du gradient d'un champ de mouvement . . 128
6.3. Composantes du champ de mouvement synthétique 133

7.1. Interpolation linéaire et interpolation de Deslaurier-Dubuc d'ordre 4. . . . 137
7.2. Interpolation de Deslauriers-Dubuc d'ordre 4 139
7.3. Exemple de reprojection : $P_{3,0}$ 139
7.4. Exemple de reprojection : $P_{3,1}$ 141
7.5. Test des ondelettes sans rebonds 1D sur un créneau 145
7.6. Test des ondelettes sans rebonds 1D sur un signal continu par morceau . . 146
7.7. Test des ondelettes sans rebonds 1D sur un créneau 147
7.8. Test des ondelettes sans rebonds 1D sur un signal continu par morceau . . 148
7.9. Test des ondelettes sans rebonds 1D sur données réelles 149
7.10. Schéma de *lifting* 2D linéaire 150
7.11. Schéma de *lifting* 2D linéaire modifié 150
7.12. Schéma de *lifting* 2D linéaire symétrisé (au sein du schéma) 151
7.13. Schéma de *lifting* 2D linéaire symétrisé (par changement d'opérateurs) . . 151
7.14. Test des ondelettes sans rebonds 2D 153

8.1. Construction d'un voisinage classique 160
8.2. Construction des voisinages issus des ESM 161
8.3. Illustration du problème de convergence pour les ESM de type S-T 162
8.4. Exemple de solutions de gestion des filtres médians pour les ESM 164
8.5. Illustration du problème de la transformation en tout-ou-rien 166
8.6. *Foreman* (trames 3-4) : érosion de taille 1. 172
8.7. *Mobile* (trames 3-4) : ouverture de taille 3. 173
8.8. *Mobile* (trames 5-10, détail) : fermeture de taille 3. 174
8.9. *Foreman* (trames 1-9) : segmentation avec marqueurs — marqueurs utilisés pour la segmentation. 175

8.10. *Foreman* (trames 1-9) : segmentation avec marqueurs — ligne de partage des eaux utilisant un élément structurant classique 176
8.11. *Foreman* (trames 1-9) : segmentation avec marqueurs — ligne de partage des eaux utilisant un ESM 177
8.12. Illustration du problème de la création de nouveaux extrema 179
8.13. Contre-exemple pour la dilatation adjointe mal-définie 181
8.14. Non-existence de la dilatation adjointe 183
8.15. Digitalisation d'une forme sous-jacente à différentes échelles. 184

Liste des tableaux

3.1. Liste des séquences de test 57
3.2. PSNR : block-matching contre méthode de Bernard régularisée 62

5.1. Table de paramètres pour l'algorithme de He-Kim 110
5.2. Table de paramètres pour l'algorithme de Mallat-Falzon adapté 111
5.3. Algorithme d'He-Kim : PSNR 112
5.4. Algorithme d'He-Kim : Images non-codées 112
5.5. Algorithme d'He-Kim : Débit réel 113
5.6. Algorithme de Mallat-Falzon ($r_0^* = 6.7$) : PSNR 114
5.7. Algorithme de Mallat-Falzon ($r_0^* = 6.7$) : Images non-codées 114
5.8. Algorithme de Mallat-Falzon ($r_0^* = 6.7$) : Débit réel 115
5.9. Algorithme de Mallat-Falzon ($r_0^* = 8.0$) : PSNR 115
5.10. Algorithme de Mallat-Falzon ($r_0^* = 8.0$) : Images non-codées 115
5.11. Algorithme de Mallat-Falzon ($r_0^* = 8.0$) : Débit réel 116

6.1. Choix de l'opérateur de mise-à-jour en fonction des (α_l) 131
6.2. Comparaison du choix des composantes et des directions privilégiées . . . 134
6.3. Gain de codage pour l'utilisation du *lifting* joint 135

Notations

Objets mathématiques

$\mathbf{x}$	point $\mathbf{x}$
$\mathbf{v}$	vecteur $\mathbf{v}$
M	matrice M
$\boldsymbol{s}$	signal $\boldsymbol{s}$

Matrices

Id	matrice identité
$\mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$	matrice diagonale remplie avec les valeurs $\lambda_1, \lambda_2, \ldots, \lambda_n$
${}^t\mathbf{v}$	vecteur transposé du vecteur $\mathbf{v}$
${}^tM$	matrice transposée de la matrice M
$\mathrm{vp}\, M$	valeurs propres de la matrice M
$\mathrm{tr}\, M$	trace de la matrice M

Fonctions

$\mathcal{F}$	transformée de Fourier
$\mathcal{W}$	transformée en ondelettes
$\mathrm{H}f$	matrice hessienne de la fonction f
∇f	gradient de la fonction f.
$\tilde{z}$	objet mathématique dual de z
$\mathbf{1}_E$	fonction indicatrice sur l'ensemble E
$\mathrm{card}\, A$	cardinal de l'ensemble A
δ	fonction de Dirac
$\widehat{f}$	transformée de Fourier de la fonction f
$f \star g$	convolution de f et de g
$\overline{w}$	conjugué du complexe w

Variables aléatoires

$\mathrm{med}(X)$	valeur médiane de l'ensemble X
$\mathrm{E}\, X$	espérance de la variable aléatoire X
$\mathrm{arrondi}(t)$	arrondi du réel t à l'entier le plus proche
$\mathrm{arrondi}(\mathbf{x})$	du point $\mathbf{x}$ au point de coordonnées entières le plus proche.
$\mathcal{H}(X)$	entropie de la variable aléatoire X
$\mathcal{H}_d(X)$	entropie différentielle de la variable aléatoire X
$\mathcal{I}(X;I)$	information mutuelle des variables aléatoires X et Y

Ensembles

$\mathbf{R}$	ensemble des réels
$\mathbf{Z}$	ensemble des entiers relatifs
$\mathbf{N}$	ensemble des entiers naturels
$\ell_2(\mathbf{Z})$	ensemble des suites de carrés sommables
$L_2(\mathbf{R})$	ensemble des fonctions de carré intégrable

Je suis comme un roseau dans la rivière
Si le courant m'emporte
Je crois que je le suivrais

Poème japonais

Introduction

Introduction

Le travail de thèse présenté dans ce document s'intéresse principalement aux problèmatiques de la compression vidéo, mais aussi à la manière dont certaines idées de ce domaine peuvent irriguer les recherches en morphologie mathématique.

Tour d'horizon de la compression vidéo Les principaux standards utilisés actuellement industriellement (H.263 pour la vidéophonie, MPEG-2 pour les DVD, MPEG-4-AVC/H.264 pour les futures chaînes haute-définition de la télévision numérique terreste (TNT)) s'appuient sur le codage hybride : de façon récursive, chaque image est prédite à partir d'une ou plusieurs images déjà transmises ; l'erreur de prédiction par rapport à l'image à comprimer subit une décomposition spatiale (souvent en cosinus discret), puis les coefficients ainsi obtenus sont codés de manière à tirer profit de leurs statistiques, avec des méthodes telles que le *run-length coding*, le codage de Huffman ou le codage arithmétique.

Plus récemment, la recherche s'est focalisée sur les codeurs vidéos progressifs. Ceux-ci codent une seule fois la séquence vidéo à haute définition. Le flux produit est conçu pour être aisément tronqué et s'adapter ainsi au débit disponible pour chaque utilisateur à chaque instant, moyennant une perte soit en qualité (progressivité de distorsion), soit en définition d'image (progressivité spatiale) soit encore en nombre d'images composant la séquence par unité de temps (progressivité temporelle). La plupart des codeurs vidéos progressifs s'articulent autour du filtrage temporel compensé selon le mouvement, qui utilise le schéma de *lifting* (voir partie 1.2).

Ce filtrage peut être effectué avant une décomposition spatiale : c'est le cas, par exemple, dans le codeur LIMAT, développé par Secker et Taubman [43] ou dans le codeur MC-EZBC de Woods et al. [10, 22, 9]. À l'inverse, d'autres codeurs réalisent d'abord la décomposition spatiale, puis appliquent seulement ensuite un filtrage temporel compensé en mouvement entre les sous-bandes respectives, comme le codeur IBMCTF développé par l'équipe de VUB [3].

Une fois les décompositions spatiale et temporelle effectuées, les coefficients sont encodés par un schéma liant les différentes sous-bandes entre elles et effectuant un codage plan de bits par plan de bits : les bits de poids forts sont codés en premier pour l'ensemble des coefficients, puis les bits de poids de plus en plus faibles. Ces bits passent au travers d'un codeur arithmétique binaire à contextes adaptatifs, codeur qui tente de tirer au mieux parti des statistiques des différents types de bits qu'il reçoit, selon les bits déjà codés dans des contextes identiques. Ces schémas sont déjà utilisés en compression d'image fixe, par exemple dans le standard JPEG-2000 [30], ou dans le codeur SPIHT de Saïd et Pearlman [41].

Plus généralement, nous pouvons distinguer trois temps dans la compression vidéo :

- une phase d'analyse de la séquence afin de déterminer le mouvement que celle-ci contient et pouvoir ainsi correctement tirer parti ensuite de la corrélation temporelle de la séquence vidéo.
- une phase de décompositions, spatiale et temporelle, c'est-à-dire de changement de représentation. L'objectif de cette phase est de rassembler l'énergie de la séquence sur quelques coefficients et obtenir une représentation de celle-ci la plus creuse possible. Des transformations sont effectuées sur les données (la séquence elle-même) aussi bien que sur les méta-données utilisées pour la compression (le champ de mouvement) : transformation en ondelettes, décomposition en cosinus discrets, prédiction par les voisins déjà codés, etc.
- une phase de codage proprement dite, qui vise à exprimer les coefficients le plus efficacement possible, à partir d'hypothèses sur leurs statistiques. C'est dans cette phase qu'est introduite la distorsion permettant de diminuer drastiquement le coût du codage des coefficients. L'objectif est bien évidemment de représenter les coefficients de la manière la plus compacte possible, tout en préservant au mieux la qualité de la vidéo.

Ce travail de thèse s'intéresse successivement à chacun de ces aspects et tente de proposer une sorte de boîte à outils pour améliorer ces différentes phases.

Estimation du mouvement L'estimation du mouvement, si elle constitue une première étape dans la compression d'une séquence vidéo, a aussi des applications en robotique et en analyse d'images. En particulier, la connaissance du champ de mouvement d'une séquence est une étape possible pour l'analyse de scènes à 3 dimensions. La connaissance du mouvement peut aussi permettre de délimiter les contours des objets de la séquence vidéo, c'est-à-dire de les segmenter.

La plupart des méthodes d'estimation de mouvement sont définies en vue d'une application précise. Les algorithmes cherchant à estimer le mouvement réel des objets s'appuient généralement sur l'équation du flot optique ou sur un modèle dérivé. Si nous exprimons une séquence vidéo en noir et blanc sous la forme d'une fonction à niveaux de gris $I(t, \mathbf{x})$, dépendant du temps t et de la position $\mathbf{p}$ dans l'image, nous pouvons exprimer la dérivée temporelle de celle-ci par l'équation :

$$\frac{d}{dt} I(t, \mathbf{x}(t)) = \frac{\partial I}{\partial t} + \mathbf{v} \cdot \nabla I,$$

où $\mathbf{v}(x)$ est le mouvement du point $\mathbf{x}(t)$. Si nous faisons l'hypothèse que l'illumination est constante dans la scène, c'est-à-dire que le niveau de gris d'un point réel projeté sur l'image est constant au cours du temps, nous obtenons l'équation du flot optique :

$$\frac{\partial I}{\partial t} + \mathbf{v} \cdot \nabla I = 0$$

Nous effectuerons un inventaire rapide et non-exhaustif des différentes grandes approches d'estimation du mouvement dans le chapitre 2.

La méthode développée par Christophe Bernard [5] utilise cette équation en la projetant sur une base d'ondelettes complexes pour obtenir un champ de mouvement presque dense. La qualité de l'estimation du mouvement réel des objets est convaincante d'après les résultats obtenus.

La qualité des résultats incite à se poser la question suivante : est-il possible d'utiliser cette méthode dans le cadre de la compression vidéo ? Plus précisément, quelles sont les adaptations utiles pour améliorer les résultats de la méthode de Bernard dans le cadre de la compression ? Nous tenterons d'apporter une réponse à cette interrogation dans le chapitre 3.

Par ailleurs, la méthode de Bernard est intrinséquement multi-échelles, puisqu'elle s'appuie sur la projection de l'équation du flot optique sur les ondelettes à différentes échelles. Cet approche multi-échelles permet de réduire les difficultés surgissant du fait de l'aliasage temporel, c'est-à-dire de ce que les fenêtres d'échantillonnages sont trop étroites par rapport à la distance des temps d'échantillonnage. Cela engendre une discontinuité entre un objet et lui-même à l'image suivante, qui gêne l'estimation correcte du mouvement. Travailler à des échelles grossières permet, en élargissant sa vue, de résoudre en partie l'ambiguité de l'aliasage et de pouvoir ainsi travailler sur de bonnes bases aux échelles plus fines.

Dans le chapitre 4, nous avons essayé d'adapter la résolution d'un champ suivant un modèle bilinéaire par morceau à cette approche multi-échelles, en apportant un soin particulier aux problèmes d'aliasage spatial qui peuvent survenir lors de l'application d'opérations de filtrage et sous-échantillonnage aux images.

Codage des coefficients Les débits des réseaux sont encore trop faibles pour permettre la transmission des séquences vidéos sans perte pour des applications où le temps réel entre en compte. Il faut donc effectuer une compression avec pertes de la séquence vidéo. La réduction de la taille des données, une fois celles-ci représentées de manière la plus creuse possible sur une nouvelle base, est effectuée par la quantification des coefficients. Celle-ci consiste à approximer les coefficients par une nombre fini de valeurs, et à les représenter dans le flux de données par un entier associé à la valeur de représentation.

Le paramètre de contrôle dont nous disposons est celui de la quantification des coefficients. Notre contrainte, elle, est le débit du canal de transmission. Il existe divers algorithmes visant à trouver la valeur adéquate du paramètre de quantification selon le débit à réaliser et selon les coefficients, appelés algorithmes d'allocation de débit. Ces algorithmes s'appuient sur des études des statistiques des coefficients, leur modélisation et l'analyse des propriétés de ces distributions lors de la quantification.

Pour le codeur de test du format H.263, l'algorithme d'allocation le plus utilisé est celui de Ribas-Corbera et Lei [39], mais d'autres algorithmes promettent des améliorations de qualité, comme celui proposé par He et Kim [18]. Ces auteurs, qui séparent le débit entre la part utilisée pour l'expression des coefficients annulés par la quantification et la part utilisée pour exprimer les autres coefficients, dits significatifs, font une analyse initiale similaire à celle effectuée par Mallat et Falzon [29] pour la compression d'images fixes.

Dans le chapitre 5, nous transposons l'analyse de Mallat et Falzon à la vidéo afin de

réaliser une comparaison des différents algorithmes.

Transformations spatiales Les bases d'ondelettes sont de plus en plus utilisées comme bases de représentation pour la compression : le standard JPEG-2000 s'appuie sur des transformations en ondelettes, de même que la plupart des codeurs vidéo progressifs. Le schéma de *lifting* développé par Sweldens [49], s'il présente des avantages en termes de calcul, permet surtout de développer des nouvelles formes d'ondelettes et en particulier d'introduire un aspect non-linéaire dans la transformée en ondelettes. Cette non-linéarité peut viser à adapter les ondelettes à de nouveaux types d'applications : c'est le cas par exemple du schéma de *lifting* entier qui assure que le résultat de l'application d'une transformée en ondelettes sur des données entières restera entier, ce qui est une propriété idéale pour effectuer une compression sans perte. L'introduction d'un aspect non-linéaire peut aussi viser à adapter la forme des ondelettes aux données qu'elles utilisent : les ondelettes *update-first* de Claypoole et Baraniuk [11] tentent d'adapter l'ordre de l'ondelette à la régularité locale du signal.

Après avoir rappelé dans le chapitre 1 la théorie générale des ondelettes puis le schéma de *lifting*, nous définirons à partir de celui-ci deux nouveaux types de schémas pour répondre à des problématiques particulières de compression :

- Si la transformée en ondelettes est couramment utilisée sur les coefficients de l'image, elle est moins couramment utilisée sur le champ de mouvement, où l'on se contente généralement d'un codage prédictif à partir des voisins déja codés. Néanmoins, si nous souhaitons augmenter le nombre de paramètres de mouvement, ou permettre des représentations aisément échelonnables du champ, la transformation en ondelettes est l'outil "naturel" de transformation. Les deux composantes du champ de mouvement sont généralement traitées comme deux images indépendantes. En faisant l'hypothèse qu'en réalité, ces deux composantes nous renseignent l'une sur l'autre, nous développerons au chapitre 6 un schéma de *lifting* conçu pour tirer parti de cette information mutuelle.
- Un des défauts des ondelettes d'ordre élevé lorsqu'elles sont utilisées en compression est l'apparition de rebonds autour des discontinuités. Ces rebonds sont problématiques aussi bien objectivement (en terme de distorsion mathématique) que subjectivement, c'est-à-dire du point de vue de la personne observant l'image dégradée. En incorporant des encadrements au schéma de *lifting*, nous définirons dans le chapitre 7 des "ondelettes sans rebonds" qui suppriment ce type d'artéfact. Nous pouvons espèrer ainsi améliorer la qualité visuelle de la séquence vidéo décompressée.

Filtrage morphologique et segmentation En compression vidéo, l'objectif de l'estimation du mouvement est de permettre d'exploiter au mieux la corrélation entre les différentes trames de la séquence vidéo et de s'affranchir ainsi des problèmes d'aliasage temporel. Les méthodes utilisées pour le filtrage temporel compensé en mouvement intègrent la compensation de mouvement dans le choix des pixels sur lequel le filtre temporel va s'appliquer. Ce faisant, elles "corrigent" le voisinage temporel du pixel sur lequel le

filtre s'applique : plutôt que de suivre simplement l'axe temporel, le voisinage d'un point donné suit le fil du mouvement de l'objet.

Il se trouve que la notion de voisinage d'un point est une des notions de base de la morphologie mathématique. Il est tentant de transposer cette notion de voisinage suivant le mouvement à ce domaine.

À l'heure actuelle, les outils de morphologie mathématique traitent généralement séparément les dimensions temporelle et spatiales dans leur traitement. Dans le cas de la segmentation d'objets au sein d'une séquence vidéo, les techniques les plus utilisées effectuent la segmentation de la première trame, puis utilisent les résultats obtenus pour obtenir des marqueurs qui serviront à initialiser la segmentation de la trame suivante [42, 36, 23, 50]. Nous pouvons rapprocher ce type de traitement du codage hybride, qui est lui aussi récursif.

D'autres techniques consistent à segmenter l'ensemble des trames de la séquence, puis à reconnecter les différents régions obtenues à l'aide de divers outils [8, 31, 14]. Citons en particulier le travail de Vincent Agnus [2], qui utilise des éléments structurants suivant le mouvement purement temporel (dans le sens où aucun point du voisinage ne se trouve sur la même image qu'un autre) à cette fin. Un autre exemple de filtrage morphologique purement temporel peut être trouvé dans le travail de Frédéric Guichard [16].

Il existe enfin quelques travaux [52] cherchant à considérer la séquence vidéo (2D+t) comme un volume 3D, mais ceux-ci se heurtent au problème de l'aliasage temporel. Les points rassemblés au sein d'un voisinage 3D utilisé dans une séquence vidéo peuvent avoir de ce fait une faible corrélation. Dans le chapitre 8, nous proposons une méthode pour construire des éléments structurants suivant le mouvement afin de rétablir une bonne corrélation des points d'un voisinage entre eux. Ces éléments sont utilisés pour effectuer des opérations de filtrage et de segmentation dans un espace 2D+t et non plus 3D.

Première partie .

Préliminaires

Chapitre 1.

Ondelettes

Bien que leur développement soit encore relativement récent, les ondelettes font désormais partie intégrante du monde du traitement d'image : elles s'utilisent aussi bien en compression qu'en débruitage, et la représentation qu'elles fournissent sert aussi à l'étude des structures et de leurs propriétés. Ce chapitre introduit les ondelettes à partir de l'analyse temps-fréquence, puis développe le schéma de *lifting*, outil récent qui permet d'introduire aisément des aspects non-linéaires aux ondelettes.

1.1. Ondelettes "classiques"

1.1.1. Fréquence locale

L'objectif des ondelettes est de fournir un intermédiaire entre une analyse purement temporelle d'un signal, et une analyse purement fréquentielle. La représentation temporelle d'un signal f peut être vue comme son expression sur une base de distributions de Dirac :

$$f(t) = \int_{\mathbf{R}} f(u)\delta(t-u)du. \tag{1.1}$$

De manière similaire, sa représentation fréquentielle (ou transformée de Fourier) peut s'écrire comme sa décomposition sur la base de Fourier :

$$f(t) = \frac{1}{\sqrt{2\pi}} \int_{\mathbf{R}} \widehat{f}(\omega)e^{i\omega t}d\omega. \tag{1.2}$$

La première représentation ne fournit qu'une information temporelle, mais de résolution maximale : $f(t)$ est exactement l'intensité de f au moment t. À l'opposé, la seconde représentation ne donne qu'une information fréquentielle, mais de résolution maximale : $\widehat{f}(\omega)$ est exactement l'intensité de f pour la fréquence ω. Mais $f(t)$ ne permet pas de connaître le contenu fréquentiel de f, pas plus que $\widehat{f}(\omega)$ ne donne d'information sur l'intensité du signal à un instant précis.

Chacune des représentations contient autant d'information sur le signal, mais ne permet aisément qu'un seul type d'interprétation (temporel ou fréquentiel). Ceci peut s'expliquer par la nature des éléments des bases. Les distributions de Dirac $t \mapsto \delta(t-u)$ ont une résolution temporelle infiniment haute, mais n'ont aucune résolution fréquentielle. C'est l'inverse pour les exponentielles complexes $t \mapsto e^{i\omega t}$. Nous pouvons reformuler notre objectif comme l'obtention d'une base d'éléments qui auraient une bonne résolution à la

fois en temps et en fréquence. C'est pourquoi Morlet et Gabor suggérèrent l'utilisation d'une base de fonctions intermédiaires entre les distributions de Dirac et les exponentielles complexes, qui seraient donc localisées à la fois dans les deux dimensions qui nous intéressent.

Il y a cependant une limite à la qualité de localisation que nous pouvons obtenir, qui nous est donnée par l'inégalité d'Heisenberg. Soit f une fonction de carré intégrable, dont la norme L_2 vaut 1 :

$$\int |f(t)|^2 \, dt = 1.$$

Si nous définissons le centre $c(f)$ et la largeur $\Delta(f)$ d'une telle fonction comme

$$c(f) = \int t|f(t)|^2 dt,$$

$$\Delta(f) = \sqrt{\int (t - c(f))^2 \, |f(t)|^2 dt}.$$

$\Delta(f)$ représente une mesure de la dispersion de la fonction autour de son centre (l'analogie est directe avec la moyenne et la variance). L'inégalité d'Heisenberg, valable pour toute fonction f de norme L_2 égale à 1, nous indique que :

$$\Delta(f)\Delta(\widehat{f}) \geq \frac{1}{2}. \tag{1.3}$$

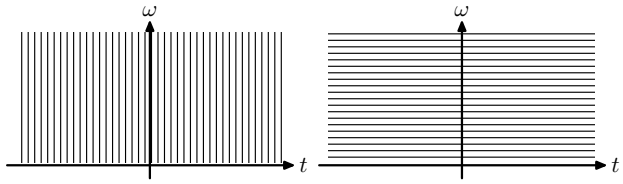
$\Delta(\widehat{f})$ correspond à la largeur fréquentielle de la fonction f. L'inégalité d'Heisenberg fait qu'il est impossible d'obtenir une fonction dont les largeurs temporelles et fréquentielles soient indépendamment arbitrairement faibles. D'autre part, nous savons que l'inégalité d'Heisenberg est une égalité pour (et seulement pour) les fonctions gaussiennes modulées et translatées :

$$G_{t_0,\omega_0,\Delta t}(t) = Ae^{-\frac{(t-t_0)^2}{2(\Delta t)^2}} e^{i\omega_0 t},$$

où A est tel que la norme de $G_{t_0,\omega_0,\Delta t}$ soit unitaire. Ces fonctions particulières sont maintenant appelées les ondelettes de Gabor.

1.1.2. Représentation temps-fréquence et temps-échelle

À chaque fonction, nous pouvons associer un rectangle temps-fréquence dans le plan (t, ω), centré en $(c(f), c(\widehat{f}))$ et de taille $(\Delta(f), \Delta(\widehat{f}))$. Ce rectangle est une représentation intuitive du support temps-fréquence de f. À une base de $L_2(\mathbf{R})$, nous pouvons associer un pavage du plan temps-fréquence. Afin d'affiner notre intuition, nous supposerons que ces rectangles en forment une partition (sans qu'il y ait en réalité de résultat théorique montrant ceci : que le plan soit nécessairement entièrement recouvert et qu'il n'y ait aucun recouvrement dans nos rectangles). Pour les distributions de Dirac, cela correspond à une infinité de lignes verticales (une ligne étant un rectangle infiniment long et étroit), tandis qu'il s'agit de lignes horizontales pour la base de Fourier, comme le représente schématiquement la figure 1.1.



(a) Dirac (b) Fourier

FIG. 1.1.: Pavages temps–fréquence associés aux représentations de Dirac et de Fourier

L'ensemble des ondelettes de Gabor forme une base redondante, et nous ne devons choisir qu'un sous-ensemble des combinaisons possibles de t_0, ω_0, Δt. Deux approches ont été proposées :

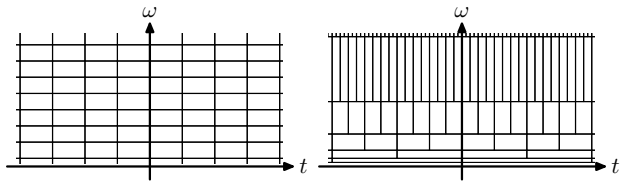
- une approche temps-fréquence, où chaque rectangle possède la même longueur et la même hauteur (voir figure 1.2(a)). Cela revient à fixer Δt, puis à faire varier t_0 et ω_0 indépendamment. Les fonctions de notre base sont alors de la forme

$$g_{t_0,\omega_0}(t) = g_0(t - t_0)e^{i\omega_0 t}$$

  où $g_0(t) = A_0 e^{-\frac{t^2}{2(\Delta t)^2}}$. La représentation induite est la transformée de Fourier à fenêtre.
- une approche temps-échelle, dans laquelle la largeur des fonctions est inversement proportionnelle à la fréquence (i.e. $\omega_0 \Delta t$ est une constante c_0) comme sur la figure 1.2(b). Dans ce cas, les fonctions de notre base deviennent :

$$g_{t_0,\Delta t}(t) = \frac{1}{\sqrt{\Delta t}} g_0\left(\frac{t - t_0}{\Delta t}\right)$$

  où cette fois $g_0 = A_0 e^{-\frac{t^2}{2(\Delta t)^2}} e^{ic_0 t}$. Nos rectangles ne sont pas tous de taille identique, mais leur aire est constante, de même que leur résolution fréquentielle relative $\frac{\Delta\omega}{\omega_0}$.



(a) Temps–fréquence (b) Temps–échelle

FIG. 1.2.: Pavages du plan temps–fréquence pour la représentation temps–fréquence et la représentation temps–échelle

Les ondelettes généralement utilisées en traitement d'images font partie de la seconde catégorie de fonctions.

1.1.3. Transformée en ondelettes continue

Choisissons une fonction de base ψ, que nous appellerons ondelette, qui satisfait la condition d'admissibilité

$$C_\psi = \int_{\mathbf{R}} \frac{|\widehat{\psi}(\omega)|^2}{\omega} d\omega < +\infty. \tag{1.4}$$

Nous définissons alors la transformée en ondelettes continue $\mathcal{W}f$ de la fonction f par :

$$\mathcal{W}f(t,s) = \int_{\mathbf{R}} f(\tau) \frac{1}{\sqrt{s}} \overline{\psi\left(\frac{\tau - t}{s}\right)} d\tau. \tag{1.5}$$

La tranformée inverse est :

$$f(t) = \frac{1}{C_\psi} \iint_{\mathbf{R}^2} \mathcal{W}f(\tau,\sigma) \frac{1}{\sqrt{\sigma}} \psi\left(\frac{t-\tau}{\sigma}\right) d\tau \frac{d\sigma}{\sigma^2}. \tag{1.6}$$

Cette transformation est, de façon similaire à la transformée de Fourier, quasiment isométrique :

$$\|f\|^2_{L_2(\mathbf{R})} = \frac{1}{C_\psi} \iint_{\mathbf{R}^2} |\mathcal{W}f(t,s)|^2 dt \frac{ds}{s^2}. \tag{1.7}$$

La transformée en ondelettes continue $\mathcal{W}f$ est cependant une représentation très redondante de f.

1.1.4. Transformée en ondelettes discrète

Si nous reprenons notre analogie avec les rectangles du pavage, le fait est que les rectangles se recouvrent, même s'ils ont tous des centres différents. Pour éliminer ce recouvrement, il nous faut prendre des valeurs discrètes pour les paramètres de nos fonctions de base.

Morlet a suggéré de construire des bases ou des *frames* à l'aide du modèle suivant :

$$g_{t_0,\Delta t}(t) = \frac{1}{\sqrt{\Delta t}} g\left(\frac{t - t_0}{\Delta t}\right)$$

où les valeurs de Δt sont choisies sur une grille géométrique, et où les pas de translation sont proportionnels à Δt :

$$\begin{aligned} \Delta t &= b^j \\ t_0 &= k\Delta t \end{aligned}$$

Le jeu d'échelles le plus couramment utilisé est celui des échelles dyadiques 2^{-j}. Nous obtenons alors des familles de fonctions de la forme $g_0(2^j(t-2^{-j}k)) = g_0(2^j t - k)$, où j et k sont des entiers. Si nous normalisons selon la norme L_2, ce qui est le cas le plus courant, nous obtenons des familles d'ondelettes de la forme $(\psi_{jk})_{j,k\in\mathbf{Z}}$ où $\psi_{jk}(t) = 2^{j/2}\psi(2^j t - k)$.

Mallat a établi un parallèle entre la représentation temps-fréquence de Morlet et les filtres miroirs en quadrature utilisés par Burt, Adelson et Simoncelli en compression

d'image. Pour cela, il a décrit une famille de décompositions en ondelettes qui peuvent être implémentées très efficacement avec une “transformée en ondelettes rapide”, dans laquelle l'ondelette ψ est une convolution infinie de filtres discrets. Il est possible d'obtenir deux filtres discrets m_0 et m_1 :

$$\begin{aligned} k &\mapsto m_0[k] \qquad k \in \mathbf{Z} \\ k &\mapsto m_1[k] \qquad k \in \mathbf{Z} \end{aligned}$$

dont les transformées de Fourier $\omega \mapsto \widehat{m}_0(\omega)$ et $\omega \mapsto \widehat{m}_1(\omega)$ sont 2π-périodiques. Nous supposons qu'il existe une fonction d'échelle ϕ et une ondelette ψ de $L_2(\mathbf{R})$ telles que :

$$\widehat{\phi}(\omega) = \prod_{k=1}^{+\infty} \widehat{m}_0(\frac{\omega}{2^k}) \tag{1.8a}$$

$$\widehat{\psi}(\omega) = \widehat{m}_1(\frac{\omega}{2})\widehat{\phi}(\frac{\omega}{2}) \tag{1.8b}$$

Sous certaines conditions, la famille (ψ_{jk}) est alors une base orthonormale, et la décomposition en ondelettes d'un signal échantillonné peut être réalisée en un temps restreint avec une séquence d'étapes de filtrage et de sous-échantillonnage.

Cette approche par filtres réduit significativement la complexité de l'objet à concevoir : au lieu d'avoir à choisir une fonction, nous devons choisir un jeu discret (et bien souvent fini) de coefficients pour nos filtres.

1.1.5. Analyses multi-résolutions

Toute la théorie s'appuyant sur les filtres discrets a été largement développée durant la dernière vingtaine d'années. De nombreux théorèmes reliant la nature des filtres et celle des ondelettes ont été énoncés et ont permis la définition de familles entières d'ondelettes.

1.1.5.1. Cadre théorique

Le cadre dans lequel Mallat a inscrit toute la théorie qu'il a développée s'appuie sur la définition d'analyses multi-résolutions. Une analyse multi-résolution est une suite de sous-espaces fermés de $L_2(\mathbf{R})$, notée $(V_j)_{j\in\mathbf{Z}}$, dont les propriétés sont les suivantes :

$$V_j = \left\{ \sum_{k\in\mathbf{Z}} a_k \phi_{jk} : a_k \in \mathbf{R} \right\} \tag{1.9a}$$

$$V_j \subset V_{j+1} \tag{1.9b}$$

$$\bigcap_{j\in\mathbf{Z}} V_j = \{0\} \tag{1.9c}$$

$$\overline{\bigcup_{j\in\mathbf{Z}} V_j} = L_2(\mathbf{R}). \tag{1.9d}$$

Ces équations amènent plusieurs remarques :

– L'équation (1.9a) signifie que V_j est un espace de Riesz engendré par la famille dénombrable $(\phi_{jk})_{k\in\mathbf{Z}}$. Cette définition s'appuie sur la métrique sous-jacente de l'espace fonctionnel, dans la mesure où cet espace peut être défini comme la fermeture de l'espace des combinaisons linéaires finies de ϕ_{jk}. Cela impose une contrainte sur la fonction ϕ (si les ϕ_{jk} sont les dilatés/translatés de celle-ci). En effet, pour la métrique L_2, la mise en correspondance :

$$\begin{aligned} \ell_2(\mathbf{Z}) &\to L_2(\mathbf{R}) \\ (a_k)_{k\in\mathbf{Z}} &\mapsto \sum_{k\in\mathbf{Z}} a_k \phi_{0k} \end{aligned}$$

se doit d'être continue. Cela implique qu'une fonction ϕ avec une décroissance trop lente ne peut pas être utilisée dans ce cadre.

– Notre choix des $(\phi_{jk})_{k\in\mathbf{Z}}$ nous pousse à considérer, avec l'équation (1.9a) que V_{j+1} doit être plus riche, en termes de capacité d'expression, que V_j, sans que ceci implique que le second soit un sous-ensemble du premier. C'est notre seconde hypothèse (1.9b) qui l'impose. Cela signifie aussi que $\phi \in V_1$, et donc, en utilisant les propriétés en termes d'échelle et d'invariance par translation de notre famille, que :

$$\phi(t) = 2\sum_{k\in\mathbf{Z}} m_0[k]\phi(2t-k). \tag{1.10}$$

Nous avons donc ici l'apparition de notre filtre m_0.

– La troisième hypothèse est indiquée simplement pour la clarté, dans la mesure où le seul signal de $L_2(\mathbf{R})$ quand la résolution diminue, est le signal nul, quelle que soit la forme de ϕ. La dernière hypothèse est valide dès que $\widehat{\phi}$ s'annule en 0.

Il est naturel de souhaiter exprimer la différence entre deux espaces successifs V_j et V_{j+1}. Nous contruisons pour ceci un nouvel espace de Riesz W_0 tel que :

$$V_0 \oplus W_0 = V_1. \tag{1.11}$$

Nous pouvons contruire W_0 comme l'espace engendré par les translatés d'une fonction ψ (qui deviendra notre ondelette) :

$$W_0 = \left\{ t \mapsto \sum_{k\in\mathbf{Z}} d_k \psi(t-k) : d_k \in \mathbf{Z} \right\}. \tag{1.12}$$

Cela impose que la fonction ψ soit dans l'espace V_1 et donc que :

$$\psi(t) = \sum_{k\in\mathbf{Z}} m_1[k]\phi(2t-k), \tag{1.13}$$

ce qui amène le second filtre discret m_1.

Si nous passons dans le domaine fréquentiel pour transformer les convolutions en produits, nous finissons par obtenir exactement les équations (1.8a) et (1.8b).

1.1.5.2. Bases d'ondelettes

L'équation (1.11) peut être transposée à une échelle j quelconque :

$$V_j \oplus W_j = V_{j+1}, \tag{1.14}$$

ce qui nous donne par itération :

$$V_j \oplus W_j \oplus \cdots \oplus W_{j'-1} = W_{j'} \qquad \text{si } j < j'. \tag{1.15}$$

Si nous faisons tendre j' vers $+\infty$ (et respectivement j vers $-\infty$), nous obtenons les deux décompositions :

$$L_2(\mathbf{R}) = V_j \oplus \overline{\bigoplus_{j'=j}^{+\infty} W_{j'}} \qquad \forall j \in \mathbf{Z} \tag{1.16}$$

$$L_2(\mathbf{R}) = \overline{\bigoplus_{j=-\infty}^{+\infty} W_j} \tag{1.17}$$

Si nous effectuons l'union des bases de chacun des espaces de Riesz dans ces sommes directes, nous arrivons à plusieurs bases d'ondelettes :

$$\mathcal{B}_j = \{\phi_{jk} : k \in \mathbf{Z}\} \cup \{\psi_{j'k} : j' \geq j, k \in \mathbf{Z}\} \tag{1.18}$$

$$\mathcal{B} = \{\psi_{jk} : j \in \mathbf{Z}, k \in \mathbf{Z}\} \tag{1.19}$$

1.1.5.3. La transformée en ondelettes

Cette transformée est utilisée pour les signaux échantillonnés. Pour un échantillonnage uniforme, le signal est souvent décrit alors par :

$$f = 2^{j/2} \sum_{k \in \mathbf{Z}} f[k/2^j] \phi_{jk}$$

où l'échantillon $f[k/2^j]$ est estimé par :

$$f[k/2^j] \simeq f(k/2^j).$$

Nous avons ici la représentation de f sur notre base de V_j. La transformée en ondelettes consiste à obtenir son expression sur la base correspondant à la somme directe suivante :

$$V_L \oplus W_L \oplus \cdots \oplus W_{j-1} \qquad L < j.$$

Cette transformation est récursive et consiste à remplacer, à chaque étape, la représentation d'un $V_{j'}$ donné par la représentation sur $V_{j'-1} \oplus W_{j'-1}$. Nous obtenons donc ainsi successivement des décompositions adaptées à chacune des sommes directes :

$$\begin{gathered} V_{j-1} \oplus W_{j-1} \\ V_{j-2} \oplus W_{j-2} \oplus W_{j-1} \\ \vdots \\ V_L \oplus W_L \oplus W_{L+1} \oplus \cdots \oplus W_{j-1}. \end{gathered}$$

1.1.5.4. Filtres duaux, ondelettes duales

L'étape fondamentale dans notre transformée en ondelettes est le changement de base :

$$V_{j+1} \rightarrow V_j \oplus W_j.$$

Ceci correspond à la mise en correspondance suivante :

$$\ell_2(\mathbf{Z}) \rightarrow \ell_2(\mathbf{Z}) \times \ell_2(\mathbf{Z})$$
$$(a_{j+1,k})_{k\in\mathbf{Z}} \mapsto [(a_{jk})_{k\in\mathbf{Z}}, (d_{jk})_{k\in\mathbf{Z}}].$$

Si on note A_j et D_j les fonctions 2π–périodiques dont les coefficients sont les suites discrètes $k \mapsto a_{jk}$ et $k \mapsto d_{jk}$:

$$A_j(\omega) = \sum_{k\in\mathbf{Z}} a_{jk} e^{-ik\omega}$$
$$D_j(\omega) = \sum_{k\in\mathbf{Z}} d_{jk} e^{-ik\omega}$$

l'itération de base s'écrit comme l'application d'une matrice de transfert :

$$\begin{bmatrix} A_j(2\omega) \\ D_j(2\omega) \end{bmatrix} = \begin{bmatrix} \widehat{m}_0(\omega) & \widehat{m}_0(\omega+\pi) \\ \widehat{m}_1(\omega) & \widehat{m}_1(\omega+\pi) \end{bmatrix} \begin{bmatrix} A_{j+1}(\omega) \\ A_{j+1}(\omega+\pi) \end{bmatrix} \tag{1.20}$$

Une condition nécessaire pour que cette transformation soit inversible est donc que la matrice de transfert

$$T(\omega) = \begin{bmatrix} \widehat{m}_0(\omega) & \widehat{m}_0(\omega+\pi) \\ \widehat{m}_1(\omega) & \widehat{m}_1(\omega+\pi) \end{bmatrix}$$

soit bornée sur $[0, 2\pi]$ et d'inverse borné sur $[0, 2\pi]$. Dans ce cas, on appelle matrice de transfert duale la matrice $\tilde{T}(\omega) = T(\omega)^{-1}$. Il existe deux autres fonctions 2π–périodiques $\tilde{\widehat{m}}_0$ et $\tilde{\widehat{m}}_1$ telles que $\tilde{T}(\omega)$ s'écrive :

$$\tilde{T}(\omega) = \begin{bmatrix} \tilde{\widehat{m}}_0(\omega) & \tilde{\widehat{m}}_0(\omega+\pi) \\ \tilde{\widehat{m}}_1(\omega) & \tilde{\widehat{m}}_1(\omega+\pi) \end{bmatrix}$$

Ces filtres définissent les ondelettes duales $\tilde{\phi}$ et $\tilde{\psi}$ par des relations identiques à (1.8a) et (1.8b) :

$$\tilde{\widehat{\phi}}(\omega) = \prod_{k=1}^{+\infty} \tilde{\widehat{m}}_0\left(\frac{\omega}{2^k}\right) \tag{1.21a}$$
$$\tilde{\widehat{\psi}}(\omega) = \tilde{\widehat{m}}_1\left(\frac{\omega}{2}\right) \tilde{\widehat{\phi}}\left(\frac{\omega}{2}\right) \tag{1.21b}$$

Les fonctions $\tilde{\phi}$ et $\tilde{\psi}$ sont donc des ondelettes duales dans le sens où pour tout j, on a les formules de décomposition sur $L_2(\mathbf{R})$:

$$f = \sum_{k\in\mathbf{Z}} \left\langle f, \tilde{\phi}_{jk} \right\rangle \phi_{jk} + \sum_{j'\geq j, k\in\mathbf{Z}} \left\langle f, \tilde{\psi}_{j'k} \right\rangle \psi_{j'k} \tag{1.22}$$

pour tous $j \in \mathbf{Z}$ et $f \in \ell_2(\mathbf{Z})$, et en faisant tendre j vers $-\infty$ la formule de décomposition homogène :

$$f = \sum_{j,k\in\mathbf{Z}} \left\langle f, \tilde{\psi}_{jk} \right\rangle \psi_{jk} \tag{1.23}$$

1.1.6. L'algorithme de la transformée en ondelettes rapide

Les coefficients des filtres m_0, m_1 et des filtres duaux $\tilde{m}_0$ et $\tilde{m}_1$ interviennent dans le calcul des changements de base

$$\{\phi_{jk} : k \in \mathbf{Z}\} \cup \{\psi_{jk} : k \in \mathbf{Z}\} \leftrightarrow \{\phi_{j+1,k} : k \in \mathbf{Z}\}$$

avec les formules suivantes :

– dans le sens de transformation dit « direct » (*forward wavelet transform*), on a

$$a_{jk} = 2\sum_{l\in\mathbf{Z}} \tilde{m}_0[k] a_{j+1,2l-k}$$
$$d_{jk} = 2\sum_{l\in\mathbf{Z}} \tilde{m}_1[k] a_{j+1,2l-k}$$

– et dans le sens de transformation inverse (*inverse wavelet transform*), on obtient

$$a_{j+1,k} = \frac{1}{2}\sum_{l\in\mathbf{Z}} m_0[2l-k] a_{jl} + m_1[2l-k] d_{jl}$$

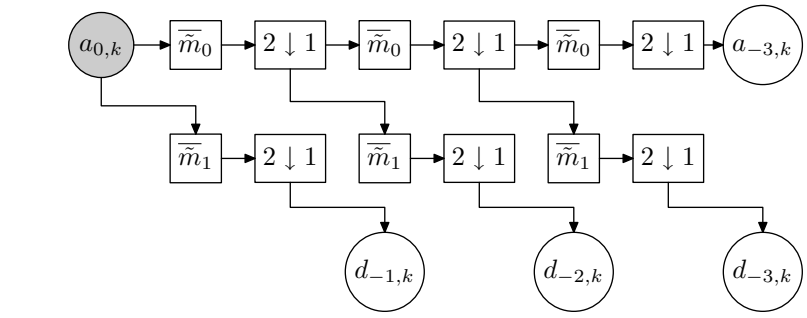
Des schémas de transformation directe et de transformation inverse sont représentés sur la figure 1.3 entre les échelles $j = 0$ et $j = -3$.

Pour un nombre fini N d'échantillons, une transformation en ondelettes (jusqu'à n'importe quelle profondeur autorisée par la taille de l'échantillon) prend moins de $A \times N$ opérations, où la constante A dépend naturellement de la taille des filtres. Ceci est en théorie meilleur qu'une transformée de Fourier rapide qui prend de l'ordre de $N \log N$ opérations.

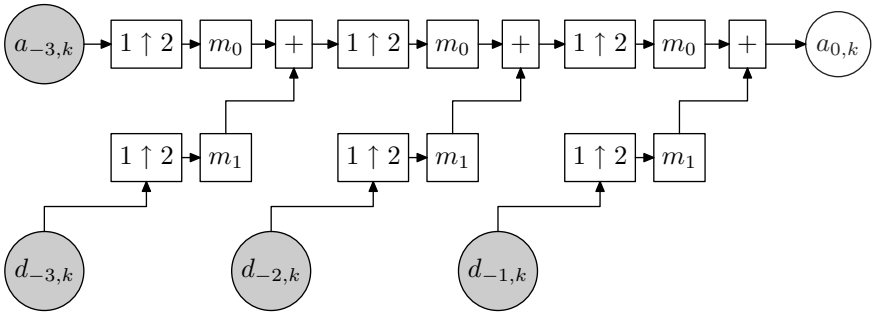
1.1.7. Les ondelettes orthogonales

Les ondelettes orthogonales sont des ondelettes ψ telles que la famille $\left(t \mapsto 2^{j/2}\psi(2^j t - k)\right)_{j,k\in\mathbf{Z}}$ soit une base orthogonale de $L_2(\mathbf{R})$. C'est le cas dès que $\phi = \tilde{\phi}$ et $\psi = \tilde{\psi}$, ce qui équivaut à écrire que la matrice de transfert et la matrice de transfert duale sont égales, soit encore que la matrice de transfert est unitaire pour tout ω. Ceci se traduit par la contrainte sur les filtres m_0 et m_1 par :

$$\begin{array}{rcrcll} |m_0(\omega)|^2 & + & |m_0(\omega+\pi)|^2 & = & 1 & \forall\omega \\ m_0(\omega)\overline{m_1(\omega)} & + & m_0(\omega+\pi)\overline{m_1(\omega+\pi)} & = & 0 & \forall\omega \\ |m_1(\omega)|^2 & + & |m_1(\omega+\pi)|^2 & = & 1 & \forall\omega \end{array}$$



(a) Transformée directe



(b) Transformée inverse

FIG. 1.3.: Transformées en ondelettes rapides. Les cercles grisés désignent les composantes d'entrée, tandis que les cercles blancs désignent les composantes de sortie.

Dans ce cas, les filtres m_0 et m_1 sont appelés filtres miroirs en quadrature, selon la terminologie d'Esteban et Galand reprise par Adelson et Simoncelli. De plus les sommes directes qui apparaissent dans les équations (1.24) sont toutes orthogonales. En pratique le choix des filtres se réduit au seul choix de m_0, car alors un choix pour m_1 s'impose :

$$m_1(\omega) = e^{i\omega}\overline{m_0(\omega+\pi)}$$

Historiquement, les premières ondelettes qui ont été mises au point sont les ondelettes orthogonales (Meyer, Mallat), si bien que des familles d'ondelettes non orthogonales ont reçu l'appellation d'ondelettes biorthogonales. Le préfixe « bi » est censé rappeler que deux bases d'ondelettes sont utilisées, une pour l'analyse (la base duale) et une pour la reconstruction. Une étude systématique des ondelettes biorthogonales a été menée par Cohen, Daubechies et al.

Il existe un certain nombre de familles d'ondelettes orthogonales couramment utilisées. Les plus connues sont sans doute les ondelettes de Daubechies. Ces ondelettes résultent de compromis optimaux entre deux critères contradictoires : le nombre de moments nuls des ondelettes et la taille de leur support (ces deux critères contradictoires rappellent, dans une certaine mesure, l'inégalité de Heisenberg). Il existe d'autres familles d'ondelettes orthogonales, comme les *coiflets*, du nom de Ronald Coifman, ou les *symmlets* qui sont des ondelettes presque symétriques.

Les bases d'ondelettes orthonormées ont un avantage théorique considérable dans les problèmes de compression ou de débruitage : la métrique d'erreur utilisée est en général la métrique L_2, et celle-ci s'exprime très simplement avec les coefficients d'une décomposition dans une base orthonormale. Dans le cas du débruitage, il se trouve également qu'un bruit blanc gaussien a une décomposition également très simple dans une base orthonormale : les coefficients sont alors également des variables gaussiennes indépendantes centrées et de même variance. En pratique cependant, les ondelettes orthogonales ne présentent pas la même flexibilité que les ondelettes biorthogonales.

1.1.8. Décroissance des coefficients et régularité

Il est désirable que les développements partiels d'une fonction convergent le plus vite possible vers la fonction originale. En d'autres termes, nous souhaitons que les coefficients du développement tendent rapidement vers 0 quand $j \to +\infty$. En fait, cette décroissance est liée au nombre de moments nuls de l'ondelette duale et à la régularité de la fonction.

Nous disons que $\tilde{\psi}$ possède p moments nuls si

$$\int_{\mathbf{R}} \tilde{\psi}(t)t^k dt = 0 \qquad\qquad \forall k \in \{0,\ldots,p-1\}.$$

Ceci est équivalent à supposer que la transformée de Fourier de $\tilde{\psi}$ possède un zéro d'ordre p en $\omega = 0$, ou encore à supposer que ψ est orthogonale à tout polynôme de degré inférieur à p.

Il est possible de montrer que si f est p fois différentiable avec une dérivée bornée d'ordre p sur un intervalle I, alors ses coefficients d'ondelettes décroissent en $2^{-j(p+1/2)}$

sur I, c'est-à-dire qu'il existe un réel M tel que :

$$|\langle \tilde{\psi}_{jk}, f\rangle| \leq M 2^{-j(p+1/2)} \qquad \text{si } \operatorname{supp} \tilde{\psi}_{jk} \subset I.$$

Il y a donc un lien étroit entre la régularité locale de la fonction et la décroissance des coefficients d'ondelettes. Une assertion quasi-inverse est aussi vraie : si la décroissance des coefficients est de la forme $|\langle \tilde{\psi}_{jk}, f\rangle| \leq M 2^{-j(p+1/2)}$ et si ψ est p-Lipschitzienne, alors f est r-Lipschitzienne pour tout $r < p$.

1.2. Le schéma de *lifting*

La présentation faite jusqu'à présent de la construction des ondelettes s'appuie essentiellement sur une approche fréquentielle utilisant la transformée de Fourier. Ces dernières années, Sweldens et Daubechies [49] ont proposé une nouvelle méthode de construction qui fait appel à des considérations purement spatiales : le schéma de *lifting*. Ce schéma permet d'obtenir simplement des ondelettes dites de "seconde génération", dans le sens qu'elles possèdent des propriétés que n'ont pas les ondelettes issues des dilatations et translations d'une ondelette-mère. Par exemple, il est possible de créer des ondelettes qui sont tout à fait adaptées à des grilles finies ou à des grilles irrégulières. D'autre part, le schéma de *lifting* permet de fabriquer des ondelettes non-linéaires.

1.2.1. Principe général

Soit un signal réel $\boldsymbol{x} = (x_k)_{k\in\mathbf{Z}}$. Nous séparons nos échantillons en échantillons pairs $(x_{2k})_{k\in\mathbf{Z}}$ et impairs $(x_{2k+1})_{k\in\mathbf{Z}}$: ils forment ainsi 2 nouveaux signaux, respectivement $\boldsymbol{x_e}$ et $\boldsymbol{x_o}$. Un étage de transformation en ondelettes consiste à passer d'un signal donné à deux signaux deux fois plus petits, l'un, $\boldsymbol{h}$, d'énergie beaucoup plus faible (détails), l'autre, $\boldsymbol{\ell}$ étant une représentation approximative du signal initial, en utilisant la corrélation contenue dans celui-ci. Ici, nous pouvons utiliser l'information contenue dans $\boldsymbol{x_e}$ pour prédire $\boldsymbol{x_o}$ de manière approximative, et donc remplacer $\boldsymbol{x_o}$ par la différence entre notre prédiction et le véritable signal. Soit P notre fonction de prédiction. Nous prenons alors :

$$\boldsymbol{h} = \boldsymbol{x_o} - P(\boldsymbol{x_e}) \tag{1.25a}$$

Il est clair qu'étant donné $\boldsymbol{x_e}$ et $\boldsymbol{h}$, il est possible d'obtenir exactement $\boldsymbol{x_o}$:

$$\boldsymbol{x_o} = \boldsymbol{h} + P(\boldsymbol{x_e}) \tag{1.25b}$$

Dans les endroits pour lesquels P aura bien prédit $x_{o,k}$, h_k sera faible. $\boldsymbol{h}$ a donc toutes les chances d'avoir une entropie nettement plus faible que $\boldsymbol{x_o}$ si P a été correctement choisi.

La transformation qui nous donne $(\boldsymbol{x_e}, \boldsymbol{h})$ à partir de $(\boldsymbol{x_e}, \boldsymbol{x_o})$ nous a permis de diminuer l'entropie. Néanmoins, si nous observons ce qui se passe dans le domaine fréquentiel, nous pouvons constater qu'il y a un fort risque d'aliasage, puisque nous n'avons fait que sous-échantillonner notre autre composante, sans lui appliquer de filtre passe-bas. En

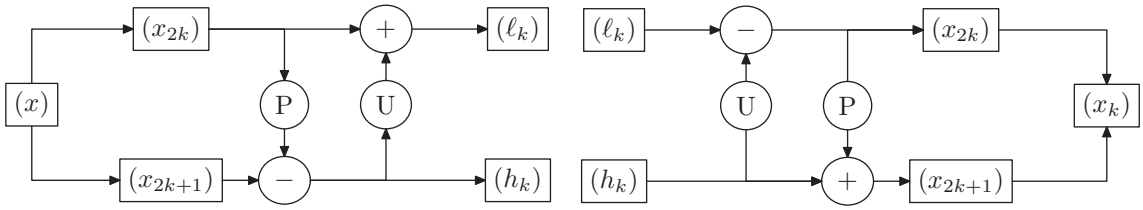
particulier, il serait souhaitable que notre signal approximé ait la même moyenne locale (à un facteur près) que le signal original. Pour ce faire, le signal $\boldsymbol{x_e}$ est transformé ("mis à jour") à l'aide de $\boldsymbol{h}$:

$$\boldsymbol{\ell} = \boldsymbol{x_e} + U(\boldsymbol{h}) \tag{1.26a}$$

Là encore, la transformation est inversible par :

$$\boldsymbol{x_e} = \boldsymbol{\ell} - U(\boldsymbol{h}) \tag{1.26b}$$

Nous avons donc défini une transformation complètement inversible possédant des caractéristiques similaires à une transformation en ondelettes. Nous pouvons représenter cette transformation à l'aide des blocs-diagrammes (1.4), qui montrent bien l'inversibilité triviale de la transformation.



(a) Transformation directe (b) Transformation inverse

FIG. 1.4.: Schéma de principe du *lifting*.

Daubechies et Sweldens [12] ont montré qu'en réalité, toute transformation en ondelettes linéaire à support compact peut être exprimée sous forme factorisée comme une alternance de prédictions, de mises à jour et de changements d'échelle. Leur démonstration part de l'équation (1.20), et montre que la matrice de transfert peut s'écrire sous la forme :

$$T = GL_1U_1 \cdots L_nU_n, \tag{1.27}$$

où les matrices $(L_i)_{1\leq i\leq n}$ (resp. $(U_i)_{1\leq i\leq n}$) sont des matrices triangulaires inférieures (resp. supérieures) de diagonale unitaire, et où G est une matrice diagonale. Or une matrice de transfert triangulaire de diagonale unitaire correspond exactement à ce que font les opérations de prédiction et de mise à jour : l'addition d'une fonction d'une des composantes (fonctions linéaires ici) à l'autre.

On peut donc espérer construire une vaste gamme de schémas de transformation avec des opérateurs de prédiction et de mise à jour non-linéaires et qui soient inversibles par construction.

1.2.2. Exemples linéaires

Donnons quelques exemples d'opérateurs de prédiction et de mise à jour simples, qui nous permettront d'obtenir des ondelettes connues.

1.2.2.1. Ondelettes de Haar

Une des prédictions les plus simples que l'on puisse effectuer - il est possible de ne rien prédire du tout, ce qui nous mène à la transformation polyphase, mais ceci n'a qu'un intérêt purement théorique - est de prendre comme prédiction de $x_{o,k}$ l'échantillon voisin $x_{e,k}$. On a alors :

$$h_k = x_{2k+1} - x_{2k} \tag{1.28a}$$

Pour la mise à jour, il est souhaitable, comme signalé précédemment, de maintenir la moyenne constante sur le signal à un facteur près. De plus, il est possible de conserver une certaine ressemblance entre l'étape de prédiction et celle de mise à jour, en ne faisant intervenir qu'un des voisins du point à mettre à jour.

Cela nous mène à :

$$\ell_k = x_{2k} + \frac{1}{2}h_k \tag{1.28b}$$

On vérifie aisément que

$$h_k = x_{2k+1} - x_{2k} \tag{1.29a}$$

$$\ell_k = \frac{1}{2}(x_{2k} + x_{2k+1}) \tag{1.29b}$$

Cela correspond, à la normalisation de $\boldsymbol{d}$ près, à la transformée par ondelettes de Haar.

1.2.2.2. Ondelettes biorthogonales 5/3

Il est possible de reprendre l'exemple précédent avec une interpolation à un ordre plus élevé. Avec une interpolation d'ordre 1 par exemple, on a

$$h_k = x_{2k+1} - \frac{1}{2}(x_{2k} + x_{2(k+1)}) \tag{1.30a}$$

Pour faire la mise à jour en se servant des deux voisins là aussi, cela impose de prendre

$$\ell_k = x_{2k} + \frac{1}{4}(h_k + h_{k+1}) \tag{1.30b}$$

Là encore, cela équivaut à faire

$$h_k = x_{2k+1} - \frac{1}{2}(x_{2k} + x_{2(k+1)}) \tag{1.31a}$$

$$\ell_k = \frac{3}{4}x_{2k} + \frac{1}{4}(x_{2k-1} + x_{2k+1}) - \frac{1}{8}(x_{2k-2} + x_{2k+2}) \tag{1.31b}$$

Ce qui correspond aux ondelettes bi-orthogonales 5/3.

1.2.2.3. Ondelettes bi-orthogonales 7/9

Il est aussi possible de recomposer les ondelettes bi–orthogonales 7/9, à l'aide des résultats de [12].

$$
\begin{aligned}
y_k &= x_{2k+1} - \alpha(x_{2k} + x_{2(k+1)}) && \alpha \simeq -1.58613 \\
z_k &= x_{2k} + \beta(x_{2(k-1)+1} + x_{2k+1}) && \beta \simeq -0.05298 \\
{h'}_k &= y_k - \gamma(z_k + z_{k+1}) && \gamma \simeq 0.88291 \\
{\ell'}_k &= z_k + \delta(y_{k-1} + y_k) && \delta \simeq 0.44351 \\
h_k &= \eta {h'}_k && \eta \simeq 1.14960 \\
\ell_k &= \frac{1}{\eta} {\ell'}_k
\end{aligned}
$$

Cependant, le résultat devient moins compréhensible, dans la mesure où il faut alterner 2 fois pas de prédiction et pas de mise à jour.

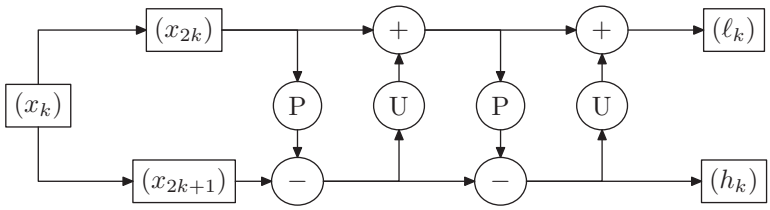


FIG. 1.5.: Schéma de principe du *lifting* correspondant aux ondelettes biorthogonales 7/9

L'exemple de ces ondelettes illustre les limites du procédé de factorisation et de la conception d'ondelettes linéaires à l'aide du schéma de *lifting*. En effet, si un schéma de lifting à deux étages reste compréhensible (annulation directe du signal de détail, puis réduction de l'aliasage), il le devient beaucoup moins dès que des étages supplémentaires lui sont adjoints. Les signaux intermédiaires $\boldsymbol{y}$ et $\boldsymbol{z}$ ne possèdent pas toujours de propriétés particulières facilement exprimables : chacun des groupes prédiction/mise–à–jour n'a plus de fonction précise pris individuellement, seule leur combinaison en a une.

1.2.3. Intérêts du schéma de *lifting*

Nous avons déjà cité quelques-uns des bénéfices à utiliser le schéma de *lifting*. Il est également possible d'en noter d'autres, et non des moindres.

1.2.3.1. Calcul plus rapide

Un des avantages au point de vue de l'implantation du schéma de *lifting* est la possibilité de réaliser tous les calculs *en place*, c'est-à-dire en utilisant seulement l'espace déjà utilisé pour stocker les données. En effet, les coefficients de détail sont stockés directement à la place des coefficients impairs, sans qu'il y ait un quelconque besoin d'allouer une mémoire

intermédiaire, dans la mesure où les calculs pour un coefficient de détail ne font intervenir aucun des autres coefficients impairs que celui dont il prend la place. Le raisonnement est le même lors d'une étape de mise à jour, pour les coefficients pairs.

D'autre part, Daubechies et Sweldens [12] ont montré que la complexité de l'algorithme du schéma de *lifting* était au pire égale à celle des algorithmes classiques de calcul de transformation par ondelettes. Si le temps est parfois comparable, on peut aussi espérer, dans le cas d'ondelettes un peu complexes, un gain de 50% en terme de nombre d'opérations.

Ces deux éléments contribuent à rendre le lifting intéressant comme schéma d'implantation pour son efficacité, et pas simplement pour la possibilité qu'il offre en terme de conception.

1.2.3.2. Visualisation de l'impact spatial

Avec le schéma de *lifting*, il est très aisé de visualiser l'impact spatial des opérations effectuées. Contrairement au design des ondelettes, il n'y a jamais besoin de passer par la transformée de Fourier pour déterminer les filtres duaux. Lors du design de filtres adaptatifs, le lifting permet de savoir comment l'adaptativité va modifier localement le signal, et, en fonction de ce que l'on aura fait à la première étape, d'adapter en conséquence localement la seconde étape, même si cela ne reste pas toujours simple.

1.2.4. Conception d'ondelettes non-linéaires

Une des principales motivations pour créer ces ondelettes de seconde génération est, comme nous l'avons signalé auparavant, soit de pouvoir nous adapter à des contraintes particulières, en terme d'espace de départ, ou d'espace d'arrivée - c'est le cas des ondelettes entières ou des ondelettes sur un intervalle - soit d'obtenir des ondelettes fondamentalement non-linéaires, dans la conception même des opérateurs de mise à jour et de prédiction - comme nous le verrons avec les exemples des ondelettes morphologiques, et des ondelettes *update-first* proposées par Claypoole et Baranuik.

1.2.4.1. Ondelettes entières

Une des difficultés des ondelettes est la difficulté d'en créer qui puissent conserver la nature entière des données, en particulier en vue de la compression sans perte. Comment nous assurer de l'aspect inversible de telles ondelettes ? Le *lifting* nous propose une solution toute simple, en utilisant la partie entière (cf. [7]). Définissons

$$h_k = x_{2k+1} - \lfloor \frac{1}{2}(x_{2k} + x_{2(k+1)}) \rfloor \tag{1.32a}$$

$$\ell_k = x_{2k} + \lfloor \frac{1}{4}(h_k + h_{k+1}) \rfloor \tag{1.32b}$$

Comme nous l'avons dèjà signalé, par la nature même du schéma de *lifting*, la tranformation proposée est inversible. Nous avons de plus la garantie que nous restons toujours sur des valeurs entières. Bien évidemment, plus le schéma aura d'étapes, plus le résultat

sera une approximation : si l'on utilise le schéma en 4 pas des ondelettes 9/7, l'opération de partie entière aura été utilisée 4 fois.

1.2.4.2. Ondelettes sur un intervalle

Le *lifting*, de par sa formulation spatiale, permet de gérer très facilement les problèmes de bords qui se posent dès que l'on travaille sur un intervalle fini. Par exemple, si x_{2k+1} est le dernier échantillon de l'intervalle, et que les échantillons sont habituellement prédits linéairement, on peut choisir

- soit de symétriser l'intervalle. Dans ce cas, $x_{2(k+2)} = x_{2k}$ et la prédiction de x_{2k+1} devient :
$$P(x_{2k+1}) = \frac{x_{2k} + x_{2k}}{2} = x_{2k},$$
- soit de continuer l'interpolation linéaire, toujours à partir des plus proches voisins dans l'ensemble des échantillons pairs, et alors de choisir comme prédiction de x_{2k+1} :
$$P(x_{2k+1}) = x_{2k} + \frac{x_{2k} - x_{2(k-1)}}{2}.$$

En fonction de la prédiction choisie, il est aussi plus simple de réaliser la construction d'étapes de mise à jour qui permettent au signal approximé de vérifier certaines propriétés (par exemple, maintien de la moyenne du signal, de son énergie, etc.)

En allant plus loin, il est aussi aisé de définir des ondelettes existant sur une grille quelconque (*"any-shape wavelets"*). À chaque échelle est associé un sous-ensemble de points de la grille G_i, tel que
$$i \leq j \Rightarrow G_i \subset G_j$$
Nous obtenons alors l'équivalent de nos points pairs (G_i) et impairs ($G_{i+1} \setminus G_i$). La disposition des points de la grille permet ensuite de définir des prédicteurs adéquats, puis les fonctions de mise à jour associées pour la réduction de l'aliasage.

1.2.4.3. Ondelettes morphologiques

Ondelettes 2D et grilles en quinconce Dans le cas linéaire, le passage des ondelettes 1D aux ondelettes 2D se fait à l'aide du produit tensoriel. On utilise la fonction d'échelle ϕ et l'ondelette-mère 1D ψ pour définir en 2D :

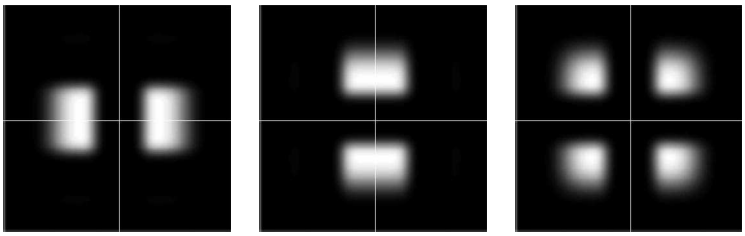
$$\phi_{\ell\ell} = \phi \otimes \phi \tag{1.33}$$
$$\psi_{\ell h} = \phi \otimes \psi \tag{1.34}$$
$$\psi_{h\ell} = \psi \otimes \phi \tag{1.35}$$
$$\psi_{hh} = \psi \otimes \psi. \tag{1.36}$$

$\phi_{\ell\ell}$ est notre nouvelle fonction d'échelle, et $\psi_{\ell h}$, $\psi_{h\ell}$ et ψ_{hh} forment un ensemble de 3 ondelettes-mères, capturant 3 types de détails différents, comme le montre le spectre de

chacune (figure 1.6). Ceci est possible dans la mesure où, puisque nous sommes dans le cas linéaire, si m_a et m_b sont deux filtres de convolutions alors :

$$(f * m_a) * m_b = (f * m_b) * m_a = f * (m_a * m_b).$$

Cette équation montre bien que l'ordre du filtrage, vertical puis horizontal ou bien horizontal puis vertical, n'a pas d'importance dans le cas des ondelettes linéaires.



(a) $\widehat{\psi h \ell}$ (b) $\widehat{\psi \ell h}$ (c) $\widehat{\psi h h}$

FIG. 1.6.: Spectre des ondelettes 2D classiques

Puisque P et U peuvent être de nature absolument quelconque, certains ont imaginé utiliser pour ceux-ci des opérateurs morphologiques. Nous sortons ici fortement du cadre linéaire. Dans le contexte de l'utilisation d'opérateurs non-linéaires, alterner filtrage vertical et filtrage horizontal n'est plus équivalent à faire d'abord le filtrage horizontal puis ensuite le filtrage vertical : les ondelettes que nous créons en appliquant le filtrage dans une direction puis une autre ne sont plus séparables. Si dans le cas des ondelettes entières, la différence reste faible[1], ce n'est pas nécessairement le cas lors de l'utilisation d'opérations morphologiques. D'autre part, la nature même des opérateurs morphologiques pousse à définir un cadre propre au travail à deux dimensions.

Les mathématiciens qui ont travaillé sur les multi-résolutions morphologiques utilisent donc des grilles en quinconce : plutôt que de séparer les points de l'image en deux ensembles, ceux d'abscisses paires et ceux d'abscisses impaires, puis d'effectuer un redécoupage entre ceux d'ordonnées paires et ceux d'ordonnées impaires, on sépare ici les points entre ceux dont la somme entre l'abscisse et l'ordonnée est paire, et ceux pour lesquels cette somme est impaire :

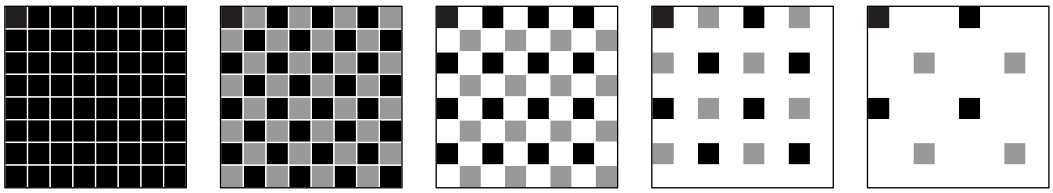
$$\boldsymbol{x_e} = (x_{i,j})_{i,j \in \mathbf{Z}, i+j \equiv 0 \mod 2} \tag{1.37}$$

$$\boldsymbol{x_o} = (x_{i,j})_{i,j \in \mathbf{Z}, i+j \equiv 1 \mod 2}, \tag{1.38}$$

comme le montre la figure 1.7. Les coefficients d'un même groupe sont alors placés en quinconce les uns par rapport aux autres. À l'échelle $j-2$, on revient à la grille habituelle (celle qui correspondrait à l'échelle $j-1$ dans le cas du produit tensoriel). Au lieu de réduire le nombre d'échantillons par 4, le procédé réduit entre chaque échelle le nombre d'échantillons par 2. La figure 1.8 montre comment représenter de façon compacte les images à ces nouvelles échelles intermédiaires.

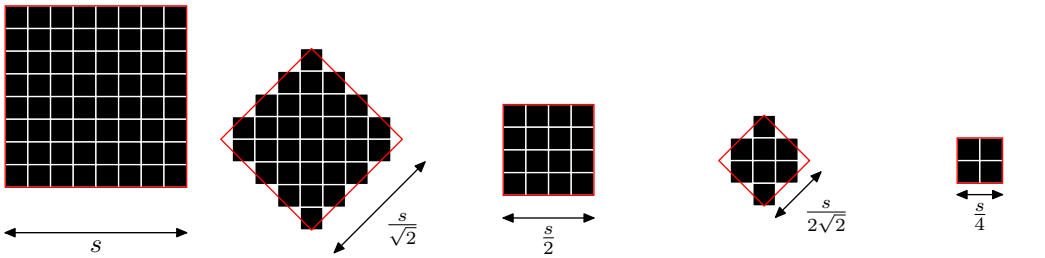
[1] il est possible de montrer que l'erreur commise en utilisant l'ondelette entière plutôt que l'ondelette linéaire à partir de laquelle elle a été construite est bornée.

Dès lors, chaque passage d'une échelle à une autre est complètement fait en 2 dimensions. Cela prend tout son sens dans le cadre de l'utilisation d'opérateurs morphologiques : on utilise alors les 4 points voisins ensemble, soit l'ensemble du voisinage utilisé habituellement, au lieu d'en utiliser 2 dans une direction, et 2 dans une autre.



(a) $j = 0$ (b) $j = -1$ (c) $j = -2$ (d) $j = -3$ (e) $j = -4$

FIG. 1.7.: Grilles en quinconce : échantillons de l'image à chaque échelle j. En noir les "pairs", en gris les "impairs".



(a) $j = 0$ (b) $j = -1$ (c) $j = -2$ (d) $j = -3$ (e) $j = -4$

FIG. 1.8.: Grilles en quinconce : représentation compacte de l'image à chaque échelle j. Chaque changement d'échelle divise par 2 le nombre de points de l'image.

Ondelettes médianes Il est par exemple imaginable de prendre comme opération de prédiction et de mise à jour la médiane sur un voisinage du point à considérer, plutôt qu'une moyenne, ou une interpolation polynômiale. On obtient alors, par exemple :

$$h_{i,j} = x_{i,j} - \operatorname{med}(x_{i-1,j}, x_{i+1,j}, x_{i,j-1}, x_{i,j+1}) \qquad \text{si } i+j \equiv 1 \mod 2 \qquad (1.39a)$$

$$\ell_{i,j} = x_{i,j} + \frac{1}{2}\operatorname{med}(h_{i-1,j}, h_{i+1,j}, h_{i,j-1}, h_{i,j+1}) \qquad \text{si } i+j \equiv 0 \mod 2 \qquad (1.39b)$$

Ondelettes max Dans leur article [20], Heijmans et Goutsias proposent aussi, comme opérateur à utiliser, le maximum. Cela permet de conserver le plus possible les maxima du signal.

$$h_{i,j} = x_{i,j} - \max(x_{i-1,j}, x_{i+1,j}, x_{i,j-1}, x_{i,j+1}) \qquad \text{si } i+j \equiv 1 \mod 2 \qquad (1.40a)$$

$$\ell_{i,j} = x_{i,j} + \max(h_{i-1,j}, h_{i+1,j}, h_{i,j-1}, h_{i,j+1}) \qquad \text{si } i+j \equiv 0 \mod 2 \qquad (1.40b)$$

Ondelettes cisl Un autre exemple d'opérateur pouvant être utilisé afin de définir des ondelettes morphologiques est l'infimum sur le semi-treillis-*inf* complet (*complete inf-semi-lattice*), comme le proposent Abhayaratne et Heijmans [1]. En définissant $\preceq$ par :

$$s \preceq t \iff 0 \leq s \leq t \text{ ou } t \leq s \leq 0 \qquad \forall s, t \in \mathbf{R}, \tag{1.41}$$

nous obtenons un tel semi-treillis. On montre entre autres, dans le cas des ensembles finis $\mathcal{K}$, que :

$$\curlywedge \mathcal{K} = \operatorname{med}(0, \min \mathcal{K}, \max \mathcal{K}).$$

L'ondelette est alors définie par :

$$h_{i,j} = x_{i,j} - \curlywedge\{x_{i-1,j}, x_{i+1,j}, x_{i,j-1}, x_{i,j+1}\} \qquad \text{si } i+j \equiv 1 \mod 2 \tag{1.42a}$$

$$\ell_{i,j} = x_{i,j} + \frac{1}{2} \curlywedge \{h_{i-1,j}, h_{i+1,j}, h_{i,j-1}, h_{i,j+1}\} \qquad \text{si } i+j \equiv 0 \mod 2 \tag{1.42b}$$

L'intérêt de cet opérateur réside dans son auto-dualité. En effet, les valeurs élevées sont traitées de la même facon que les valeurs faibles disposées symétriquement sur l'intervalle des valeurs possibles. En d'autres termes, cet opérateur ne favorise ni le clair ni le sombre, contrairement à l'érosion et à la dilatation classiques, qui favorisent le sombre et le clair respectivement.

1.2.4.4. Ondelettes adaptatives "update-first"

Un des inconvénients des ondelettes linéaires est leur incapacité à s'adapter à la régularité des données. En effet, dans les zones où le signal est très régulier, nous souhaiterions utiliser des ondelettes d'ordre élevé, tandis que dans les régions où il est irrégulier, nous préfèrerions utiliser des ondelettes d'ordre plus bas.

Cela permettrait que le support des ondelettes ne franchisse pas les sauts de la fonction, et donc que seul le véritable détail soit attrapé par les ondelettes. Dans le sens de la reconstruction du signal, cela diminuerait les effets de Gibbs dans le cas où l'on interpole sans signal de détail. Dans une telle perspective, Claypoole et Baraniuk [11] ont défini des ondelettes adaptatives, c'est-à-dire ici dont l'opération de prédiction s'ajuste à la régularité des données, ou, plus particulièrement, à la présence d'un bord.

Si l'on est en mesure de détecter la présence d'un bord dans les échantillons qui sont utilisés pour la prédiction, il ne faut pas se servir des échantillons qui sont disposés de l'autre côté du bord par rapport à l'échantillon que l'on veut prédire. En pratique, cela signifie que plus on se rapproche du bord, plus on diminue le nombre d'échantillons servant à prédire, et donc, l'ordre de la prédiction. L'idée initiale serait d'écrire :

$$\boldsymbol{h} = \boldsymbol{x_o} - P_{adap}(\boldsymbol{x_e}) \tag{1.43a}$$

$$\boldsymbol{\ell} = \boldsymbol{x_e} + U(\boldsymbol{h}) \tag{1.43b}$$

où $P_{adap}(\boldsymbol{x_e})(n) = \sum \boldsymbol{x_e}(k) F_n(n-k)$, l'ordre de F_n dépendant de la régularité de (x_e) en n.

Réaliser ceci dans le cadre classique pas de prédiction puis pas de mise à jour est faisable, mais cela présente au moins deux inconvénients :

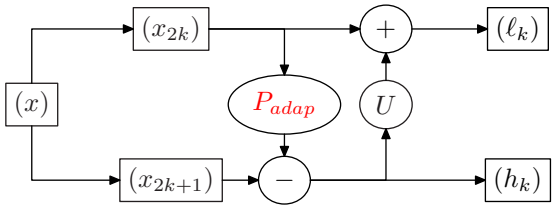


FIG. 1.9.: Schéma de principe du *lifting*, avec étape de prédiction non-linéaire.

- il est difficile de connaître les propriétés de l'étape de mise à jour qui suit la prédiction. En effet, la construction d'opérateurs de mise à jour respectant un jeu de contraintes donné dépend de la connaissance des caractéristiques de chacun des signaux $\boldsymbol{h}$ et $\boldsymbol{x}_e$ par rapport au signal d'origine. Dans le cas où il y a une incertitude sur la nature de l'opérateur de prédiction, cette connaissance est à peu près impossible et la construction d'opérateurs de mise à jour adéquats difficile.
- l'autre problème est lié à la stabilité de la reconstruction dans le cas d'une compression avec perte de données entre l'analyse et la synthèse.

  En supposant qu'une partie des données vient d'être sacrifiée sur l'autel de la réduction de leur entropie par le couperet de la quantification, l'étape de prédiction dans la reconstruction ne s'appuie pas sur les coefficients corrects, mais sur des coefficients modifiés par la quantification. Comme la décision du type de prédiction qui s'appuie sur ces données est discrète (un filtre ou l'autre) ou, pour reformuler, discontinue, cela signifie que, pour certains signaux, une quantification d'un pas arbitrairement fin (avec un signal quantifié $\tilde{\boldsymbol{x_e}}$ arbitrairement proche de l'original $\boldsymbol{x_e}$) engendrera une distorsion qui, elle, ne sera pas arbitrairement fine ; nous n'avons pas *a priori* :

  $$|P_{adap}(\tilde{\boldsymbol{x_e}}) - P_{adap}(\boldsymbol{x_e})| \nrightarrow 0.$$

  La solution généralement adoptée dans ce type de problème est de recourir à la synchronisation, c'est-à-dire d'utiliser les valeurs disponibles au décodeur pour effectuer notre décision. Mais ceci n'est pas applicable pour l'instant, puisque les valeurs dépendent de l'étape de mise à jour, et donc du choix même des filtres de prédiction.

La solution proposée par Claypoole et Baraniuk est d'effectuer d'abord l'étape de mise à jour, et seulement ensuite l'étape de prédiction (1.10).

Le premier problème est résolu d'office, si ce n'est qu'il demande une remise à l'échelle des coefficients, par exemple dans le cas du maintien de la moyenne. Pour le second, le déplacement de l'étape de prédiction juste avant l'étape de quantification permet de mettre en place le système de synchronisation, et donc de garantir une certaine stabilité des données par rapport à la dureté de la quantification.

Malheureusement, les schémas avec la mise à jour en premier présentent d'autres limitations. La non-linéarité introduite ne s'adapte correctement qu'aux schémas de *lifting* à deux échelles : si nous souhaitons introduire de l'adaptativité dans un schéma à plus de deux échelles, nous nous heurtons au problème de la compréhension du sens de celles-ci.

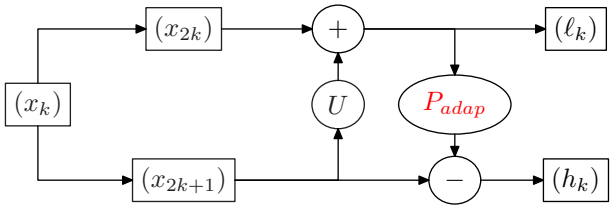


FIG. 1.10.: Schéma de principe du *lifting*, avec étape de mise à jour en premier, puis avec une étape de prédiction non-linéaire.

1.3. Conclusion

L'analyse par ondelettes linéaires et son extension via le schéma de *lifting* sont des outils puissants dans le cadre du traitement d'image. Le schéma de *lifting* est en particulier extrêmement prometteur de par sa complexité faible, sa capacité à générer des transformations adaptatives prenant en compte la structure du signal. Enfin, la garantie d'inversibilité de la transformation, indépendamment de la nature des opérateurs de prédiction et de mise à jour utilisés, explique, elle aussi, l'effervescence des recherches dans ce domaine. Il existe en particulier d'autres tentatives d'introduire des éléments d'adaptivité dans le schéma de lifting, utilisant des systèmes plus complexes, comme dans le cas des travaux de Piella et Heijmans [37], ou bien d'utiliser des données annexes du signal (par exemple le mouvement dans le cadre de la vidéo).

Chapitre 2.

L'estimation du mouvement

Lors de la prise de vue d'une scène, l'étude du déplacement des objets qui la composent est un atout pour nombre d'applications. En robotique, cela permet de déterminer et souvent donc de prévoir l'évolution de la position des objets avec lesquels l'interaction est possible. En compression vidéo, cela permet de comprendre le mieux possible où se trouve la redondance temporelle de la séquence et de savoir décrire une image à l'aide des images environnant celle qui nous intéresse. Enfin, les opérations de traitement d'image peuvent utiliser cette information à des fins diverses : segmentation, filtrage, etc.

Trois éléments sont nécessaires en pratique pour mesurer le mouvement. Tout d'abord, il faut un modèle de mouvement : ce modèle peut être déduit d'un modèle de mouvement des objets en 3 dimensions joint à un modèle de projection sur la caméra, ou bien déterminé plus arbitrairement en fonction des contraintes de représentation. Ensuite, il est nécessaire de choisir un critère à minimiser afin d'évaluer les qualités respectives des différentes valeurs possibles des paramètres du modèle. Enfin, le choix d'un algorithme de recherche est crucial dans l'obtention d'un équilibre entre la complexité du calcul et la qualité de la minimisation du critère.

Il existe donc plusieurs grandes familles de méthodes pour la mesure du mouvement, selon les différents choix effectués. Nous en présentons ici quelques-unes, après avoir formulé plus complètement le problème et donné l'équation du flot optique.

2.1. L'équation du flot optique

Nous considérons une séquence d'images comme une fonction réelle $I(t, \mathbf{x})$ du temps t et de l'espace à deux dimensions $\mathbf{x} = (x_1, x_2)$. La plupart des modèles développent leur théorie en supposant que les t et $\mathbf{x}$ sont continus, de même que I, puis discrétisent ensuite leurs équations. Chaque image est généralement le résultat de la projection d'objets réels sur le plan de l'image dont il est donc possible de suivre la trajectoire : si $\big(X_1(t), X_2(t), X_3(t)\big)$ est un point réel de la scène, il se projette en un point $(x_1(t), x_2(t))$.

Le flot optique à un point et un temps donnés est défini comme la vitesse de ce point :

$$\mathbf{v} = \frac{d\mathbf{x}(t)}{dt} \tag{2.1}$$

Nous pouvons suivre un point $\mathbf{x}(t)$ qui se déplace le long du champ de mouvement. L'intensité lumineuse de ce point de l'objet à chaque instant est $I(t, \mathbf{x}(t))$. La variation

de cette intensité au cours du temps peut s'exprimer de la manière suivante :

$$\frac{d}{dt}I(t,\mathbf{x}(t)) = \frac{\partial I}{\partial t} + \mathbf{v}\cdot\nabla I \tag{2.2}$$

La plupart des méthodes font l'hypothèse de l'illumination constante : la luminance d'un objet est constante au cours du temps, ce qui se traduit simplement par l'équation suivante :

$$\frac{d}{dt}I(t,\mathbf{x}(t)) = 0 \tag{2.3}$$

Les équations (2.2) et (2.3) nous mènent à l'équation dite "du flot optique" :

$$\frac{\partial I}{\partial t} + \mathbf{v}\cdot\nabla I = 0 \tag{2.4}$$

Cette équation nous fournit une contrainte à respecter pour le mouvement. Il faut souligner néanmoins tout de suite que cette équation ne nous fournit qu'une seule équation scalaire pour un point donné, alors qu'il nous faut déterminer deux inconnues, les deux composantes du vecteur de mouvement au point considéré. En pratique, si nous observons l'équation (2.4), nous pouvons tout de suite constater que les solutions ne sont contraintes que dans la direction du gradient de l'image, et qu'à l'inverse, il n'est pas possible de déterminer à l'aide de cette seule équation la composante du vecteur de mouvement orthogonale au gradient de l'image.

Ceci est dû au problème d'ouverture : si nous observons localement notre image sur une zone uniforme et donc invariante par translation, il ne sera pas possible d'en déduire le véritable mouvement. Plus généralement, si notre image est invariante par une translation donnée sur une petite fenêtre, le mouvement n'est mesurable qu'à cette translation près.

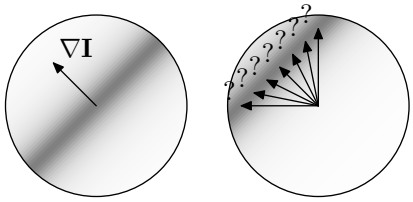


FIG. 2.1.: Illustration du problème d'ouverture. Une image régulière peut être considérée comme localement invariante par translation. Seule la composante parallèle au gradient ∇I du déplacement entre $t = 0$ et $t = 1$ peut alors être estimée. La composante orthogonale n'est pas mesurable.

D'autre part, lors de mouvements importants intervient le problème de l'aliasage : l'aspect local de l'équation (2.4) est mis à mal par l'échantillonnage temporel. Si l'objet étudié est sorti de la fenêtre de recherche donnée par l'équation, l'obtention d'un résultat valable est impossible.

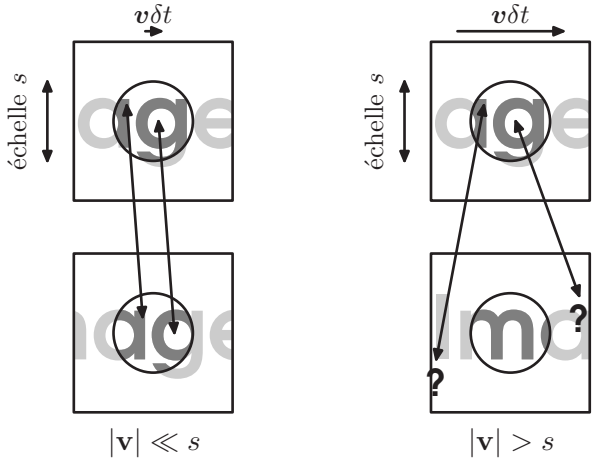


FIG. 2.2.: Ceci illustre le problème de l'aliasage temporel dans la mesure du flot optique. Si on considère des portions d'image de taille inférieure à s, on ne peut pas estimer de déplacement $\mathbf{v}$ de longueur $|\mathbf{v}|$ supérieure à s car alors les deux portions d'images considérées n'ont rien en commun (à droite).

En pratique, toutes les méthodes s'appuyant sur le flot optique effectuent une ou plusieurs hypothèses supplémentaires sur la nature du champ de mouvement, permettant d'obtenir des contraintes additionnelles. Il est possible aussi d'utiliser l'information de couleur dans le cas où l'on en dispose, mais les équations supplémentaires que l'on obtient sont généralement étroitement corrélées et n'apportent donc pas une échappatoire au problème d'ouverture. Elles améliorent cependant parfois les résultats, malheureusement souvent au prix d'un temps de calcul plus important.

2.2. L'estimation par appariement

2.2.1. Différence de régions déplacées

Après discrétisation de l'équation (2.3), un des critères qu'il est possible de minimiser est la différence entre la région courante et une région correspondant au déplacement à tester dans l'image de référence.

$$\varepsilon_{\mathbf{x_0},t} = I(t, \mathbf{x_0} + \mathbf{v}(\mathbf{x_0})) - I(t+1, \mathbf{x_0}) \tag{2.5}$$

Selon les différentes méthodes, le critère exact obtenu à partir de ces différences peut être la somme de leurs carrés, la somme de leurs valeurs absolues, ou encore divers estimateurs robustes sur le jeu de données auquel une région correspond.

2.2.2. Le block-matching simple

La différence de régions déplacées est généralement utilisée avec un modèle très simple de champ de mouvement, dans lequel le champ est constant par morceaux. Ce modèle a un nombre de paramètres faible et ceux-ci sont, de plus, indépendants. Ceci permet d'effectuer une recherche par appariement où différentes valeurs du paramètre sont essayées tour à tour, et l'optimum choisi parmi les différentes valeurs expérimentées. L'amélioration du matériel aidant, il est possible d'effectuer une recherche exhaustive, selon la précision choisie, sur une grille $\mathcal{G}$. Si le critère est la somme des valeurs absolues des erreurs, comme c'est le cas dans de nombreuses implantations de codeurs vidéo actuels, alors le champ de mouvement sur une région W_0 est :

$$\mathbf{v}_0 = \underset{\mathbf{v}\in\mathcal{G}}{\operatorname{argmin}} \iint_{W_0} |I(t, \mathbf{x}+\mathbf{v}) - I(t+1, \mathbf{x})| d^2\mathbf{x} \tag{2.6}$$

On peut noter que la fonctionnelle à minimiser n'est pas convexe (par rapport aux vecteurs de mouvement), ce qui justifie la recherche exhaustive par appariement pour ce critère.

2.2.3. Amélioration du block-matching

Divers raffinements ont été proposés à partir de cette base simple. Le premier a été de remplacer la trame de référence I par une trame interpolée $\tilde{I}$. Le champ de mouvement devient :

$$\mathbf{v}_0 = \underset{\mathbf{v}\in\mathcal{G}}{\operatorname{argmin}} \iint_{W_0} |\tilde{I}(t, \mathbf{x}+\mathbf{v}) - I(t+1, \mathbf{x})| d^2\mathbf{x} \tag{2.7}$$

Les positions interpolées sont autant de positions supplémentaires possibles pour les vecteurs de mouvement. Généralement, cette précision est fractionnaire, puisqu'il s'agit de fournir un raffinement de la grille naturelle de l'image. Dans la mesure où le block-matching n'utilise pas le gradient de l'image, il est possible d'utiliser un interpolant qui soit continu, alors que dans le cas contraire, un interpolant au moins continûment dérivable est requis.

Une autre modification du block-matching (par exemple [51]) est d'utiliser, non pas une pure partition du domaine de l'image, mais un recouvrement par des blocs qui se chevauchent légèrement. Cela revient à découpler la région W_1 sur laquelle le critère est calculé de la région W_0 du modèle de représentation du mouvement. L'obtention du paramètre du champ de mouvement se fait donc grâce à l'équation :

$$\mathbf{v}_0 = \underset{\mathbf{v}\in\mathcal{G}}{\operatorname{argmin}} \iint_{W_1} |\tilde{I}(t, \mathbf{x}+\mathbf{v}) - I(t+1, \mathbf{x})| d^2\mathbf{x} \tag{2.8}$$

où $W_0 \subset W_1$. Il existe alors deux types de points :

- ceux qui n'appartiennent qu'à une seule zone de recherche. La valeur du champ de mouvement en ces points est celle associée à la zone de recherche en question.

– ceux qui appartiennent à plusieurs zones de recherche. La valeur du champ de mouvement en ces points est interpolée à l'aide des valeurs du champ de mouvement associées aux zones de recherche de sorte que la variation du champ soit continue.

L'objectif est de permettre une transition plus douce entre les différentes zones du champ de mouvement, et donc d'atténuer les discontinuités que pourrait provoquer la déformation de l'image de référence à partir d'un champ discontinu. Il est de plus aussi possible d'attribuer des poids à chaque point de la fenêtre, pour accentuer l'effet de lissage.

D'autre part, des méthodes d'accélération du calcul de l'optimum ont été proposées, qui visent à obtenir plus rapidement un résultat, au risque de ne pas trouver le véritable optimum, au sens de l'équation (2.6). Il est possible par exemple d'effectuer une recherche hiérarchique, en testant par exemple les vecteurs de précision 2^k, puis une fois le minimum sur cette grille obtenu, tester les vecteurs de précision 2^{k-1} autour de ce minimum afin de trouver un nouveau minimum, et ce, jusqu'à arriver à la précision désirée. Cette méthode est utilisée, par exemple, dans les codeurs de test pour le standard H.264, où toutes les positions à précision entière sont testées, puis seulement les positions demi-pixelliques autour du minimum obtenu à l'étape précédente, puis encore raffinées au quart de pixel. Ceci permet un bon compromis entre l'apport de la précision sous-pixellique et le temps nécessaire pour effectuer la recherche : dans la plupart des situations, une valeur de champ proche de l'optimum est obtenue, pour un temps de calcul 16 fois plus faible. Le fait que la valeur obtenue soit proche de l'optimum vient du fait que les images que nous traitons sont à bande limitée : l'échantillonnage de l'image a déjà tronqué les plus hautes fréquences de la véritable scène. Dans ces conditions, les valeurs que l'on obtiendra grâce à l'interpolation de la fonctionnelle à minimiser seront généralement proches des valeurs des points voisins sur la grille.

2.3. Horn et Schunk

Horn et Schunk [21] ont proposé en 1980 une hypothèse de régularité du champ de mouvement. Ils notent tout d'abord que la résolution de (2.4) est équivalente à la minimisation de la fonctionnelle suivante :

$$M(\mathbf{v}) = \iint \left(\frac{\partial I}{\partial t} + \mathbf{v} \cdot \nabla I\right)^2 d^2\mathbf{x}. \tag{2.9}$$

La contrainte de régularité qu'ils imposent s'exprime sous la forme d'une fonctionnelle additionnelle de la forme :

$$R(\mathbf{v}) = \iint \|\mathbf{v}\|_{\mathcal{H}}^2 d^2\mathbf{x} \tag{2.10}$$

où $\|\cdot\|_{\mathcal{H}}$ est la norme du laplacien du champ dans le cas de la méthode de Horn et Schunk, mais pour laquelle n'importe quelle norme de Hilbert régularisante est possible (comme une norme de Sobolev). Cette contrainte correspond à l'observation empirique que le mouvement est globalement régulier.

Le champ de vecteur $\mathbf{v}$ est alors :

$$\mathbf{v} = \operatorname{argmin}\left(M(\mathbf{v}) + \lambda R(\mathbf{v})\right). \tag{2.11}$$

Rajouter la contrainte de régularité permet de transformer le problème mal posé de l'équation (2.4) en un problème bien posé, dans le sens où il existe a priori à notre équation (2.11) une solution unique qui dépend de façon continue des données. La résolution de ce problème est faite par une minimisation globale, en l'occurrence par l'inversion d'un système symétrique où chaque point de l'image fournit deux inconnues. Le paramètre λ permet de donner plus ou moins de poids à la contrainte de régularité par rapport à la contrainte d'illumination constante et permet d'équilibrer, entre autres, la sensibilité au bruit par rapport à l'adéquation aux données.

2.4. Techniques fréquentielles

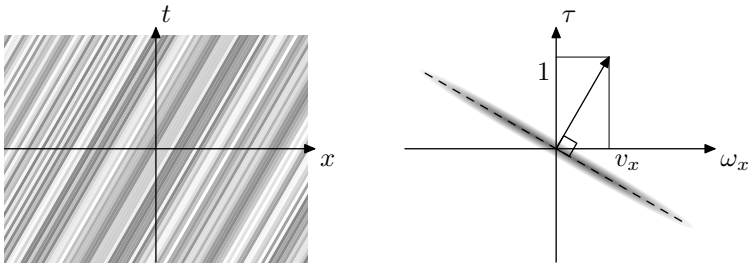
Une autre catégorie de techniques utilise des transformations, telles que la transformée de Fourier $\mathcal{F}$, afin de réaliser une analyse spectrale locale sur une petite fenêtre espace-temps. Si nous supposons que le mouvement est uniforme sur cette petite fenêtre, nous avons alors :

$$I(t, \mathbf{x}) = I(0, \mathbf{x}) * \delta(\mathbf{x} - \mathbf{v}t). \tag{2.12}$$

Si nous prenons la transformée de Fourier de cette équation, nous obtenons simplement :

$$\hat{I}(\omega, \xi) \propto \hat{I}(0, \xi)\delta({}^t\mathbf{v}\xi - \omega). \tag{2.13}$$

Cela signifie que l'essentiel de l'énergie de la transformée de Fourier de la fenêtre sera concentré sur un plan orthogonal au mouvement. Il ne reste donc plus qu'à déterminer ce plan pour obtenir localement la valeur du champ de mouvement.



Séquence en (x, t) Spectre de la séquence en (ω_x, τ)

FIG. 2.3.: Illustration du principe de base de la mesure du flot optique par les techniques fréquentielles. À gauche une séquence d'images en translation uniforme représentée en fonction des variables d'espace x et de temps t. À droite, la transformée de Fourier spatio-temporelle de cette séquence est localisée sur un hyperplan dont l'inclinaison indique la vitesse de déplacement. Plus précisément, la normale de ce plan est $(v_x, 1)$. Ainsi, en identifiant l'inclinaison du plan, on peut retrouver le flot optique v_x.

Il est à noter que ce genre de technique permet de pouvoir éventuellement déterminer des mouvements multiples, y compris leur simple existence. En effet, en présence de

plusieurs mouvements, il existera plusieurs plans différents dans le domaine spectral, chacun correspondant à un mouvement différent.

2.5. Lucas et Kanade

Lucas et Kanade [27] ont proposé une technique plus locale pour déterminer la valeur du champ de mouvement, en minimisant la fonctionnelle suivante :

$$M(\mathbf{v}) = \sum_{\mathbf{x}\in W(\mathbf{x_0})} \rho(\mathbf{x})^2 \big(\nabla I \cdot \mathbf{v} + \frac{\partial I}{\partial t}\big)^2 \tag{2.14}$$

où $W(\mathbf{x}_0)$ est une fenêtre autour du point pour lequel on souhaite déterminer le champ de mouvement, et ρ une fonction de poids sur cette fenêtre, donnant plus d'importance au centre qu'à la bordure (selon le modèle de mouvement choisi). La résolution est alors effectuée par l'équation suivante :

$$^tAR^2A\mathbf{v} = {}^tAR^2\mathbf{b} \tag{2.15a}$$

où, pour les N points de $W(\mathbf{x}_0)$ à l'instant t,

$$A = {}^t\big[\nabla I(\mathbf{x}_1), \ldots, \nabla I(\mathbf{x}_N)\big] \tag{2.15b}$$

$$R = \mathrm{diag}(\rho(\mathbf{x}_1), \ldots, \rho(\mathbf{x}_N)) \tag{2.15c}$$

$$\mathbf{b} = -{}^t(\frac{\partial I}{\partial t}(\mathbf{x}_1), \ldots, \frac{\partial I}{\partial t}(\mathbf{x}_N)) \tag{2.15d}$$

Cette résolution est d'autant plus simple que $^tAR^2A$ est une matrice 2×2, facilement inversible quand elle n'est pas singulière. La confiance que l'on peut donner à la mesure dépend de la norme des valeurs singulières de $^tAR^2A$ [45].

2.6. Approches multi-échelles

Un certain nombre d'auteurs ont proposé des méthodes d'estimation multi-échelles. Ces méthodes présentent, du point de vue de la résolution, de multiples avantages. D'une part, en agissant d'abord sur des versions filtrées et sous-échantillonnées des images de la séquence, ce type d'approche permet d'éviter un certain nombre de minima locaux qui se trouvent proches du champ nul.

D'autre part, connaître une valeur approximative du champ de mouvement permet de choisir une discrétisation adaptée. Si une telle précaution n'est pas prise, les mouvements importants ne peuvent pas être correctement calculés du fait d'approximations discrètes, certes valables pour un champ nul, mais plus pour des déplacements importants. Cela correspond aussi au problème de l'aliasage temporel : dans la mesure où l'objet finit par être déconnecté sur l'axe temporel, les dérivations effectuées dans la direction de cet axe mèlent alors des éléments complètement décorrélés, et sont donc inutilisables. Ce problème d'aliasage peut se voir aussi dans le cas des méthodes fréquentielles.

Les méthodes multi-échelles estiment le mouvement de façon incrémentale. La première estimation v^j est effectuée sur un domaine B_r de rayon r centré en 0 pour chacune des composantes, éventuellement restreint à une grille de la forme $2^j\mathbf{Z}^2$, puis elle est ensuite raffinée en lui ajoutant un incrément v^{j-1}, dont la valeur est, elle aussi, dans un domaine $B_{\frac{r}{2}}$ centré autour de 0, mais de rayon $\frac{r}{2}$. L'estimation résultante :

$$v^{\{j,j-1\}} = v^j + v^{j-1} \tag{2.16}$$

aura une valeur appartenant donc à $\{v^1\} + B_{\frac{r}{2}}$, toujours éventuellement restreint à une grille de la forme $2^{j-1}\mathbf{Z}^2$. Le procédé est itéré jusqu'à la précision souhaitée.

Il permet ainsi d'estimer des déplacements bien plus importants que ne le permettent les mêmes méthodes dénuées de l'approche multi-échelles, en diminuant les difficultés dues à l'aliasage temporel.

2.7. Flot optique sur base d'ondelettes

Dans sa thèse, Bernard [5] décrit une méthode d'obtention du champ de mouvement par projection de l'équation du flot optique sur une base d'ondelettes. Il part d'une famille d'ondelettes-mères ψ^n. Nous avons alors comme base la famille des $\psi^n_{j,\mathbf{k}}$ telle que

$$\psi^n_{j,\mathbf{k}}(x) = 2^{j/2}\psi^n(2^j\mathbf{x} - \mathbf{k}) \qquad ,j \in \mathbf{Z},\, k \in \mathbf{Z}^2. \tag{2.17}$$

L'équation du flot optique est projetée sur l'ondelette à l'échelle j et au point $\mathbf{k}$:

$$\iint \Big(\nabla I \cdot \mathbf{v} + \frac{\partial I}{\partial t}\Big)\psi^n_{j,\mathbf{k}}(\mathbf{x})d^2\mathbf{x} = 0 \qquad \forall n = 1 \ldots N \tag{2.18}$$

en reprenant la notation classique

$$\langle f, g\rangle = \iint f(\mathbf{x})\overline{g(\mathbf{x})}d^2\mathbf{x}$$

(2.18) se réécrit comme :

$$\Big\langle \frac{\partial I}{\partial x_1}v_1, \psi^n_{j,\mathbf{k}}\Big\rangle + \Big\langle \frac{\partial I}{\partial x_2}v_2, \psi^n_{j,\mathbf{k}}\Big\rangle + \Big\langle \frac{\partial I}{\partial t}, \psi^n_{j,\mathbf{k}}\Big\rangle = 0 \qquad \forall n = 1 \ldots N \tag{2.19}$$

À chaque échelle de la base, le champ de mouvement est considéré comme étant constant sur le support de l'ondelette. Cela permet de sortir la vitesse du produit scalaire et, après une intégration par partie permettant de déplacer la dérivation spatiale sur les ondelettes, l'équation (2.19) devient alors :

$$\Big\langle I, \frac{\partial \psi^n_{j,\mathbf{k}}}{\partial x_1}\Big\rangle v_1(\mathbf{k}) + \Big\langle I, \frac{\partial \psi^n_{j,\mathbf{k}}}{\partial x_2}\Big\rangle v_2(\mathbf{k}) = \frac{\partial}{\partial t}\Big\langle I, \psi^n_{j,\mathbf{k}}\Big\rangle \qquad \forall n = 1 \ldots N \tag{2.20}$$

Nous obtenons donc un système de plusieurs équations de la forme, au lieu de la simple équation du flot optique, au prix de l'approximation que le flot est localement constant.

Cette approche présente certains avantages : contrairement à la méthode de Horn et Schunk et aux méthodes de résolution globale en général, cette méthode a l'avantage de ne résoudre que des systèmes locaux très simples avec peu d'inconnues : elle doit minimiser des systèmes de la forme $|Av - b|$, où les dimensions de A sont faibles. D'autre part, à cause de la nature même des ondelettes, l'adaptation de la résolution des systèmes à une approche multi-échelles est immédiate. Dans le cadre posé par Bernard, l'approche multi-échelles va permettre de mesurer les grands déplacements à l'échelle la plus grossière (ce qui correspond aux ondelettes de plus large support), puis de raffiner autour des mesures obtenues aux échelles précédentes. La connaissance d'une valeur approximative du mouvement à une échelle donnée permet de choisir l'approximation la plus correcte possible de la dérivée temporelle de l'image afin de pallier le problème soulevé en 2.6. Enfin, afin d'éviter les problèmes de stabilité posés par l'annulation des ondelettes réelles, la base d'ondelettes sur laquelle le mouvement est projeté est une base d'ondelettes analytiques.

2.8. Conclusion

Comme nous avons pu le voir, il existe une grande variété d'approches, aussi bien dans le choix du modèle du champ de mouvement que dans la manière d'obtenir les paramètres de celui-ci. Notre présentation n'est pas exhaustive. On pourra se référer à l'article de Stiller et Konrad [46] qui proposent une typologie, là encore loin d'être complète, des méthodes d'estimation de mouvement. Pour une comparaison des méthodes, une étude a été faite par Barron, Fleet et Beauchemin [4] : différentes techniques sont évaluées sur des séquences synthétiques de référence à l'aide d'une même mesure de qualité. Celle-ci est l'écart angulaire par rapport à la valeur correcte du champ de mouvement. Si cette mesure permet de comparer entre elles les méthodes d'estimation de mouvement, elle n'est correcte que pour la recherche du véritable champ de mouvement de la scène. En revanche, pour d'autres applications, d'autres métriques peuvent être pertinentes et sont utilisées de façon interne par les algorithmes dans leur processus d'estimation, favorisant ainsi la diversité du domaine.

Deuxième partie .

Estimation

Chapitre 3.

Compression vidéo avec mesure du mouvement en ondelettes

Le travail décrit dans (2.7) se plaçait dans le cadre de l'estimation pure du mouvement. Nous avons souhaité voir dans quelle mesure ce mode d'estimation est utilisable dans le cadre de la compensation de mouvement utilisée en compression vidéo. Quelles sont les adaptations permettant d'améliorer la qualité du champ pour l'utilisation que nous nous proposons de lui donner ?

3.1. Schéma de codage de référence

Le schéma de compression de la plupart des algorithmes de compression vidéo s'appuie sur quelques éléments simples :

- une méthode d'estimation de mouvement. Cette méthode permet de décrire une approximation de la trame à encoder à l'aide de la trame précédente.
- une transformée. Une fois obtenue l'approximation de la trame grâce au mouvement, les codeurs essaient d'exprimer la différence de la meilleure façon possible. Une transformation spatiale est utilisée pour tenter d'éliminer la corrélation spatiale, et regrouper l'énergie du signal sur quelques coefficients.
- un schéma de codage des coefficients. L'objectif est de représenter le plus efficacement les coefficients à transmettre : ceux de l'image de différence transformée ainsi que ceux représentant les vecteurs de mouvement, ainsi que les différents éléments syntaxiques, à l'aide de modèles sur leur distribution.

3.1.1. Quantification

Lorsqu'on travaille à des débits faibles, il n'est pas possible de représenter exactement les coefficients issus d'une transformée. La compression devient alors une compression avec pertes : ce sont des valeurs approximatives des coefficients qui sont transmises, et le signal reconstitué par le décodeur n'est pas identique au signal initial. Il y a donc apparition d'une distorsion.

Supposons que nous disposons d'un jeu de N coefficients, $(F[k])_{0 \leq k < N}$. Ces coefficients appartiennent tous à un intervalle $\mathcal{J} \in \mathbf{R}$, et peuvent y prendre une valeur quelconque. Afin de transformer ces coefficients en des nombres appartenant à un ensemble fini de

valeurs, $\bar{\mathcal{J}}$, on utilise la quantification. Celle-ci, dans le cas scalaire, consiste à partitionner la droite des réels en boîtes, dites boîtes de quantification, de la forme $[y_n, y_{n+1}[$. À chacune de ces boîtes est attribué un nombre entier, ici n. Chaque coefficient est alors représenté par le nombre associé à la boîte dans laquelle il se trouve. Le décodeur va utiliser ce nombre pour déterminer à quel intervalle $[y_n, y_{n+1}[$ le coefficient appartenait, et pour prendre comme valeur approchée $\bar{F}[k]$ de $F[k]$ un élément $x_n \in [y_n, y_{n+1}[$. Nous avons donc :

$$F[k] \in [y_n, y_{n+1}[\Rightarrow \bar{F}[k] = x_n.$$

Nous verrons dans la section 5.2 qu'on utilise en général des quantificateurs quasi-uniformes. Pour ces quantificateurs, toutes les boîtes ont la même taille Δ, sauf la boîte centrée en zéro, de taille T, le rapport optimal $\theta = T/\Delta$ dépendant de la distribution des coefficients. Ici, nous prendrons $\theta = 2$.

Le nombre de valeurs possibles pour n étant nettement plus faible que celui pour les $F[k]$[1], un nombre plus faible de bits sera nécessaire pour représenter le coefficient. En compensation, une distorsion apparaîtra dans l'image décodée, d'autant plus forte que l'approximation de $F[k]$ par $\bar{F}[k]$ sera fausse. Afin de pouvoir évaluer le compromis effectué, il faut pouvoir mesurer la distorsion de l'image d'une part, et d'autre part le coût de codage des coefficients quantifiés.

3.1.2. Rapport signal sur bruit

Dans le traitement du signal, la mesure la plus couramment utilisée pour mesurer la distorsion introduite par un traitement est le rapport signal sur bruit, noté SNR (*Sound-to-Noise Ratio*), mesuré en décibels. Si nous notons $I(x, y)$ les valeurs des pixels de l'image originale, $\bar{I}(x, y)$ ceux de l'image transmise et $\mathcal{I}$ l'ensemble des points de l'image, le SNR se définit par :

$$\mathrm{SNR}(I, \bar{I}) = 10 \log_{10} \frac{\sum_{(x,y)\in\mathcal{I}} I(x,y)^2}{\sum_{(x,y)\in\mathcal{I}} (\bar{I}(x,y) - I(x,y))^2}.$$

En pratique, dans le cadre du traitement d'images, l'intervalle des valeurs que peut prendre le signal est fini et fixe. D'autre part, une image noire (dont les pixels valent 0) a un sens identique à une image blanche (dont les pixels valent 255). Un bruit donné pertubera autant chacune de ces deux images. Le rapport signal sur bruit ne convient donc pas, puisqu'il rapporte le bruit à l'amplitude du signal.

En traitement d'image, c'est donc le rapport signal maximal sur bruit qui est utilisé, noté PSNR (*Peak Signal-to-Noise Ratio*). Si l'on note V_{max} la valeur maximale que peut prendre le signal, le PSNR se définit comme :

$$\mathrm{PSNR}(I, \bar{I}) = 10 \log_{10} \frac{V_{max}^2 \operatorname{card} \mathcal{I}}{\sum_{(x,y)\in\mathcal{I}} (\bar{I}(x,y) - I(x,y))^2}.$$

[1] Ce nombre est fini puisqu'il est limité par la précision de la représentation informatique des réels.

3.1.3. Entropie

Intéressons-nous maintenant à la mesure du coût de codage des coefficients. Toutes les valeurs de ceux-ci ne seront pas nécessairement équiprobables. En particulier, après une transformation spatiale, l'énergie des coefficients sera la plupart du temps concentrée sur quelques-uns d'entre eux, les autres étant proches de 0. Cette valeur a donc plus de chances d'apparaître que les autres. Cette différence de probabilité d'apparition est recherchée dans la mesure où elle permet d'utiliser des codes plus courts pour représenter les valeurs des coefficients les plus fréquentes et au contraire des codes plus longs pour les valeurs les moins fréquentes.

Les coefficients sont modélisés comme la réalisation d'une variable aléatoire X, de distribution p pouvant prendre ses valeurs dans un alphabet $\mathcal{X}$. L'entropie d'une telle variable aléatoire, notée $\mathcal{H}(X)$ est définie par la formule :

$$\mathcal{H}(X) = -\sum_{x \in \mathcal{X}} p(x) log_2 p(x)$$

L'entropie sert à mesurer l'incertitude d'une variable aléatoire. Plus la valeur d'une réalisation de celle-ci est prévisible, plus l'entropie sera faible. Il est possible de montrer que l'entropie est la limite inférieure de la moyenne du nombre de bits nécessaires pour coder une réalisation de la variable, si nous connaissons la distribution p de celle-ci. De plus, cette limite peut être atteinte asymptotiquement à l'aide du codage arithmétique.

L'entropie est donc une bonne mesure pour évaluer le coût de codage des coefficients. Dans le cas particulier de la compression vidéo, il s'agit simplement d'une estimation pour au moins deux raisons. La première est qu'il est possible de tirer parti des corrélations spatiales des coefficients (y compris après transformée). La seconde est que la distribution des coefficients dépend du contenu de l'image. La distribution des coefficients est alors modélisée, et la différence entre la distribution modélisée et la distribution réelle introduit un surcoût au codage.

3.1.4. Modèle simplifié de codage

À des fins d'expérimentation, nous nous placerons dans un modèle simplifié :

- le mouvement est estimé par la méthode que nous allons tester entre la trame courante et la trame précédente.
- le champ de mouvement subit une transformation en ondelettes puis est quantifié. Nous estimons son entropie.
- le champ de mouvement est ensuite déquantifié, puis subit la transformée en ondelettes inverse.
- nous effectuons la compensation de mouvement à l'aide de ce nouveau champ.
- l'image d'erreur subit une transformation en ondelettes, puis est quantifiée. Nous estimons son entropie.
- l'image d'erreur est déquantifiée, puis subit la transformée en ondelettes inverse.
- l'image finale est déterminée, et nous estimons le rapport signal-bruit.

Notre objectif est d'obtenir une entropie totale minimale pour un rapport signal-bruit donné. Il nous faudra obtenir la meilleure répartition possible entre le débit utilisé par le champ de mouvement, et celui consacré au codage des coefficients.

3.2. Intérêt a priori de la régularisation

Un coup d'oeil sur le champ de mouvement issu de la méthode décrite en (2.7) nous donne un champ de mouvement qui est très irrégulier. Cette irrégularité est due à une forte sensibilité de cette méthode au bruit. Le champ de mouvement possède donc une forte entropie, ce qui le rend cher à coder, et donc *a priori* inefficace pour une utilisation en compression. Cela a-t-il un sens de coder cette irrégularité, qui est pour la partie résultant de la sensibilité au bruit, du bruit elle-même ? En réalité, il n'est utile de coder avec précision le mouvement que si cela correspond au mouvement d'une *texture* substantielle. Nous pouvons essayer d'atteindre cet objectif en introduisant un procédé de régularisation au sein de la méthode d'estimation : ceci limitera l'influence du bruit et des composantes non-essentielles du champ de mouvement.

3.2.1. Sensibilité au bruit

Dans la méthode décrite dans la section (2.7), la solution est obtenue par une méthode des moindres carrés. Malheureusement, il est possible que certains systèmes soient mal conditionnés, ce qui éloigne la solution obtenue de la solution réelle dès qu'il y a un soupçon de bruit. Nous pouvons définir une zone d'incertitude à ε, $\mathcal{E}_\varepsilon$, telle que :

$$\mathcal{E}_\varepsilon = \{\mathbf{v} \text{ tel que } |A\mathbf{x} - \mathbf{b}| < \varepsilon\} \tag{3.1}$$

Lorsque les contraintes ne sont pas fortes dans une direction, la zone d'incertitude va s'étendre dans cette direction. Cela correspond au fait que, pour la mise en correspondance de deux objets de la scène, il n'est pas très faux de prendre un autre vecteur dans la zone d'incertitude plutôt que la solution donnée par la résolution aux moindres carrés. C'est le cas lorsque nous nous trouvons sur une frontière entre deux régions plates : la composante du mouvement qui est perpendiculaire au mouvement sera bien définie, tandis que celle qui lui est tangentielle ne le sera pas. Nous retrouvons ici complètement le problème de l'ouverture, qui se pose dès que nous essayons d'estimer le mouvement d'une zone assez petite dont nous ne connaissons pas suffisamment les frontières, ou des points d'accroche (voir figure 3.1).

En l'absence de telles caractéristiques, l'incertitude sur les directions concernées est importante. La méthode décrite en (2.7) ne règle pas ce problème et choisit à chaque fois la solution qui minimise purement l'équation. Introduire un procédé de régularisation permettra de diminuer la sensibilité au bruit en lissant l'influence de celui-ci.

Cela correspond aussi à l'idée que le modèle de mouvement sur chacun des objets de la scène doit nous donner des zones de mouvement régulières. D'autre part, le choix de la solution possède aussi une grande importance dans le cas d'une approche multi-échelles. En effet, si la solution choisie à une échelle donnée sert à initialiser le système à l'échelle

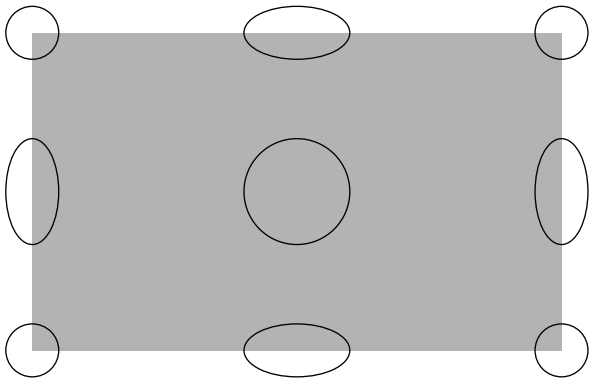


FIG. 3.1.: Incertitudes selon la texture. L'incertitude vraisemblable sur la mesure du mouvement est indiquée par une ellipse. Plus l'objet est texturé, plus l'incertitude sur son mouvement est faible. Au centre du rectangle, la texture est absente, l'incertitude sera forte. Sur les bords, l'incertitude diminuera dans la direction perpendiculaire à ce bord. Enfin, dans les coins, l'incertitude sera assez faible.

suivante - comme c'est le cas ici, un choix non pertinent à une échelle fournira un système qui sera défectueux (la compensation de l'aliasing temporelle étant incorrecte), et ceci se propagera donc au travers des échelles consécutives, sans qu'on puisse nécessairement espérer que le défaut se corrige de lui-même. De plus, le nombre de systèmes croissant avec l'échelle, un système incorrect à une échelle conduit à 4 systèmes incorrects à l'échelle suivante, et cela continue selon une progression géométrique.

Si donc la régularisation nous permet de trouver une solution plus sensée à une échelle, en diminuant notamment l'influence du bruit, il y a toutes les chances pour que la précision soit améliorée doublement à chaque étape : non seulement par le fait qu'une solution *a priori* plus correcte est adoptée à chaque échelle mais aussi du fait que les systèmes résultants aux échelles plus fines seront plus pertinents.

3.2.2. Réduction de l'entropie du champ de mouvement

La méthode d'estimation considérée ne cherche pas à obtenir un champ dont l'expression globale est la plus simple possible, ce qui signifie, en d'autres termes, qu'elle contient probablement de l'information quasiment inutile, dans la mesure où elle n'utilise pas les degrés de liberté qui lui sont offerts. A qualité d'estimation équivalente, il est préférable, dans le cadre d'une application à la compression vidéo, de choisir le champ représentable le plus simplement (voir figure 3.2). Régulariser la résolution du mouvement nous donnera des vecteurs de mouvement spatialement plus corrélés. En effet, dans les endroits où il n'y a pas de texture, choisir un vecteur ou un autre ne présente pas un grand enjeu pour le codage de la texture. En revanche, choisir un vecteur de mouvement similaire à ses voisins sera intéressant. Cette corrélation sera exploitée par la transformée en ondelettes pour fournir au codeur de coefficient le signal le plus creux possible. Ainsi, imposer une certaine régularité au champ de mouvement permettra de diminuer son coût de codage

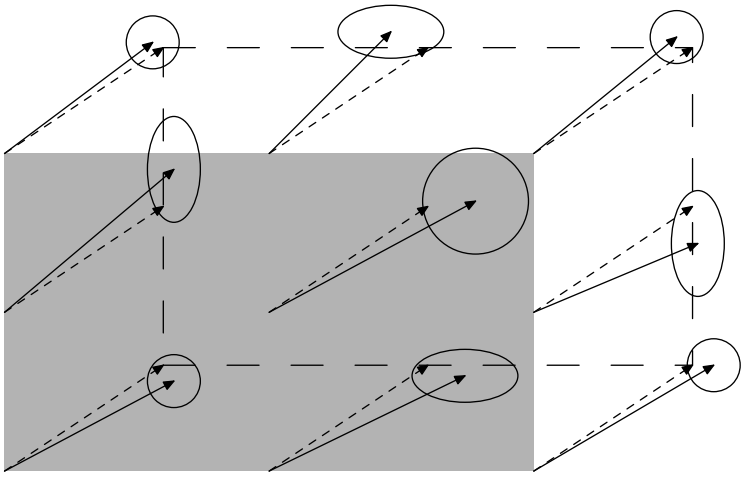


FIG. 3.2.: Recherche de solutions plus cohérentes à l'aide des ellipses d'incertitudes. Une fois déterminées les ellipses d'incertitudes, nous pouvons rechercher s'il n'existe pas un jeu de vecteurs plus cohérents au sein des ellipses. Sur notre illustration, les vecteurs en pointillé sont des alternatives possibles aux vecteurs initialement déterminés dans notre environnement bruité. Si certains ne prédisent pas "mieux" que les vecteurs initiaux, ils sont néanmoins plus proches des vecteurs environnants, ce qui diminuera l'entropie du champ de mouvement.

(en terme d'entropie), en dégradant faiblement ses propriétés prédictives : nous espérons ainsi obtenir un équilibre plus satisfaisant entre la distorsion d'une part, et le coût total de codage d'autre part, c'est-à-dire l'entropie du champ de mouvement ajoutée à celle de la trame d'erreur.

3.2.3. Précautions à prendre

La régularisation de la méthode d'estimation de mouvement permet de réduire l'entropie du champ de mouvement, tout en correspondant plus vraisemblablement au déplacement réel des objets.. En contrepartie, elle risque d'induire plusieurs difficultés. Une attention particulière doit être apportée aux endroits sur lesquels se situe une irrégularité du champ. À ces endroits, comme le résultat est nécessairement imparfait, imposer une sorte de régularité se justifie comme un choix qui ne sera pas nécessairement plus pénalisant qu'une autre solution.

D'autre part, si la contrainte du champ régulier se justifie aux échelles fines, elle devient de plus en plus fausse aux échelles grossières. Il faut donc régulariser avec de plus en plus de vigueur au fur et à mesure que l'on parcourt les échelles de la plus grossière à la plus fine.

3.3. Méthode de régularisation

3.3.1. Reformulation du problème de facon générale

Afin de régulariser le champ, nous allons utiliser la méthode suivante décrite dans toute sa généralité.

3.3.1.1. Méthode sans régularisation

Considérons un espace euclidien E à N dimensions, muni d'une base orthonormale $(e_i)_{1\leq i\leq N}$. Le domaine de l'image est de la forme

$$\mathcal{I} = \mathcal{I}_1 \times \mathcal{I}_2 \times \cdots \times \mathcal{I}_N,$$

où chaque $\mathcal{I}_i$ est un segment de $\mathbf{R}$. Nous considérons maintenant une grille multi-résolutions $(G^r)_{r_{min}\leq r\leq r_{max}}$ s'appuyant sur cette base. Plus précisément, chacune de ces grilles s'écrit sous la forme

$$G^r = 2^r\mathbf{Z} \cap \mathcal{I}.$$

À chaque point $\mathbf{p}$ de la grille G^r sont associés une matrice $A^r_\mathbf{p}$ de taille $m \times n$, un vecteur $\mathbf{b}^r_\mathbf{p}$ de taille n et une inconnue vectorielle $\mathbf{v}^r_\mathbf{p}$ de taille m. Désignons par A^r (resp. $\mathbf{b}^r$, $\mathbf{v}^r$) le champ de matrices (resp. vecteurs) sur G^r. Nous définissons les opérations classiques entre deux champs en appliquant l'opération sous-jacente en chacun des points du champ : par exemple,

$$(A^r\mathbf{v}^r)_\mathbf{p} = A^r_\mathbf{p}\mathbf{v}^r_\mathbf{p}.$$

Définissons aussi la somme d'un champ de scalaires a^r par

$$\sum a^r = \sum_{\mathbf{p}\in G^r} a^r_\mathbf{p}.$$

Ceci définit en chaque point un système d'équations sur-déterminé de la forme

$$A^r_\mathbf{p}\mathbf{v}^r_\mathbf{p} = \mathbf{b}^r_\mathbf{p}.$$

De plus, ces éléments sont reliés entre eux par

$$A^r = \mathcal{F}(\mathbf{v}^{r+1}, \mathcal{D})$$
$$\mathbf{b}^r = \mathcal{G}(\mathbf{v}^{r+1}, \mathcal{D})$$

où $\mathcal{D}$ est un ensemble de données (comme les images I_t et I_{t+1} dans le cas de l'estimation du mouvement entre ces deux images) et où nous supposons que

$$\mathbf{v}^{r_{max}+1} = 0.$$

Nous faisons enfin l'hypothèse que la signification des différents systèmes est cohérente sur chacune des grilles (par exemple, que chacun des systèmes a été obtenu de manière identique).

L'objectif est d'obtenir la valeur de $\mathbf{v}^{r_{min}}$. La résolution simple consiste à obtenir successivement chacun des $\mathbf{v}^r$, de r_{max} à r_{min}, via une résolution au moindre carré à chacune des résolutions. Nous cherchons donc pour chaque point $\mathbf{p}$ à chaque échelle r, $\text{argmin}_{v^r_\mathbf{p}} \left(|A^r_\mathbf{p} \mathbf{v}^r_\mathbf{p} - \mathbf{b}^r_\mathbf{p}|^2 \right)$. Une des solutions est obtenue en résolvant

$$ {}^tA^r{}_\mathbf{p} A^r_\mathbf{p} \mathbf{v}^r_\mathbf{p} - {}^tA^r{}_\mathbf{p} \mathbf{b}^r_\mathbf{p} = 0, \tag{3.2} $$

ce qui est possible grâce à la forme de ${}^tA^r{}_\mathbf{p} A^r_\mathbf{p}$ qui est symétrique positive.

Nous pouvons voir la résolution à une échelle donnée comme la minimisation globale d'une fonction $\mathcal{A}^r$, définie par

$$ \mathcal{A}^r(\mathbf{v}^r) = \sum |A^r \mathbf{v}^r - \mathbf{b}^r|^2. $$

Cette somme est constituée de termes complètement indépendants, c'est la raison pour laquelle il est possible d'implémenter une minimisation locale de la fonctionnelle.

3.3.1.2. Aspects théoriques de la régularisation introduite

Nous souhaitons maintenant introduire la régularisation dans ce procédé, et ainsi que nous l'avons dit, nous désirerions le faire à chacune des échelles. Nous pouvons considérer qu'il s'agit maintenant de modifier la fonction à minimiser pour amener un peu de régularité, en essayant de trouver :

$$ \underset{\mathbf{v}^r}{\text{argmin}} \left(\mathcal{A}^r(\mathbf{v}^r) + \lambda \mathcal{B}^r(\mathbf{v}^r) \right) \tag{3.3} $$

où $\mathcal{B}^r$ est une fonctionnelle de régularisation, qui va lier les valeurs de chacune des composantes d'un vecteur de mouvement aux les valeurs de la même composante des vecteurs environnants. Elle donnera ainsi une certaine structure spatiale au champ, qui n'était qu'assez faible jusqu'à présent. Dans cet objectif, nous choisissons comme fonctionnelle :

$$ \mathcal{B}^r(\mathbf{v}^r) = \sum \sum_{1 \leq i \leq m} |\nabla \mathbf{v}^r_i|^2 \qquad \text{(ici, } (\mathbf{v}^r_i)_\mathbf{p} = \mathbf{v}^r_{\mathbf{p},i}) $$

où le gradient est approché par des différences finies sur la grille.

Il nous faudra, bien évidemment, choisir λ et déterminer comment effectuer cette minimisation. En effet, les termes sont désormais interdépendants : $\mathcal{A}$ impose une contrainte sur les composantes d'un même vecteur, tandis que $\mathcal{B}$ crée un lien entre différents vecteurs indépendamment sur chaque composante de ceux-ci. La résolution ne peut donc plus être locale.

Définissons B^r, M^r, H^r et $\mathbf{c}^r$ par

$$ \begin{aligned} (B^r \mathbf{v}^r)_{\mathbf{p},i,j} &= \frac{\partial \mathbf{v}^r_{\mathbf{q},j}}{\partial \mathbf{q}_i}(\mathbf{p}) && (B^r \text{ correspond à } \nabla) \\ M^r &= {}^tA^r A^r \\ H^r &= {}^tB^r B^r && (H^r \text{ correspond à la dérivée seconde}) \\ \mathbf{c}^r &= {}^tA^r \mathbf{b}^r \end{aligned} $$

A^r, B^r, M^r et H^r sont finalement des tenseurs, avec des propriétés particulières :

$$\begin{aligned}
(A^r)^{i'_1\dots i'_N,j'}_{i_1\dots i_N,j} &= 0 && \text{si } \exists\, k \text{ tel que } i_k \neq i'_k \\
(B^r)^{i'_1\dots i'_N,j'}_{i_1\dots i_N,j,l} &= 0 && \text{si } j \neq j' \\
(M^r)^{i'_1\dots i'_N,j'}_{i_1\dots i_N,j} &= 0 && \text{si } \exists\, k \text{ tel que } i_k \neq i'_k \\
(H^r)^{i'_1\dots i'_N,j'}_{i_1\dots i_N,j} &= 0 && \text{si } j \neq j'
\end{aligned}$$

L'équation 3.3 devient alors :

$$\operatorname*{argmin}_{\mathbf{v}^r} \left(\sum |A^r\mathbf{v}^r - \mathbf{b}^r|^2 + \lambda \sum {}^t\mathbf{v}^r H^r \mathbf{v}^r \right),$$

soit encore :

$$\operatorname*{argmin}_{\mathbf{v}^r} \left(\sum \left({}^t\mathbf{v}^r \left((M^r + \lambda H^r)\mathbf{v}^r - 2\mathbf{c}^r \right) + {}^t\mathbf{b}^r\mathbf{b}^r \right) \right).$$

Il nous faut choisir une valeur pour le paramètre λ. Un choix d'une valeur *a priori* paraît malaisé au vu de la forme de la fonctionnelle. λ est en fait un paramètre permettant d'équilibrer le poids respectif des données et de la contrainte de régularisation. Nous pouvons donc nous appuyer sur la valeur λ_0 définie comme :

$$\lambda_0 = \frac{\operatorname{tr} M^r}{\operatorname{tr} H^r}. \tag{3.6}$$

λ_0 est en fait le rapport des traces de chacun des tenseurs associés aux fonctions à minimiser. En supposant que l'approximation du gradient est indépendante de la position, de la direction de dérivation et de la composante vectorielle considérée, nous obtenons alors :

$$(H^r)^{i'_1\dots i'_N,j}_{i_1\dots i_N,j} = (H^r)^{(i'_1-i_1)\dots(i'_N-i_N),1}_{0\dots0,1}, \tag{3.7}$$

et (3.6) se simplifie en :

$$\lambda_0 = \frac{\operatorname{tr} M^r}{\sum_{i_1,\dots i_N,j} (H^r)^{i_1\dots i_N,j}_{i_1\dots i_N,j}} \tag{3.8}$$

$$= \frac{\operatorname{tr} M^r}{mN \operatorname{card} G_r (H^r)^{0\dots0,1}_{0\dots0,1}} \tag{3.9}$$

Le calcul de λ_0 est donc très simple.

Pour choisir la force de la régularisation, nous pourrons nous appuyer sur ce nombre, en fixant le rapport λ/λ_0. En particulier, nous pourrons faire varier ce rapport le long des échelles, afin d'avoir une force de régularisation plus faible aux échelles grossières, puis qui augmentera au fur et à mesure que nous allons vers les échelles fines..

3.3.1.3. Schéma de la régularisation introduite

Le gradient de l'expression que nous cherchons à minimiser vaut :

$$\nabla(\mathcal{A} + \lambda B) = 2[(M^r + \lambda H^r)\mathbf{v}^r - \mathbf{c}^r]$$

Nous allons effectuer une descente de gradient classique afin de trouver une solution convenable[2]. La descente sera initialisée à partir de la solution $(\mathbf{v}_\mathbf{p}^{r,0})$ obtenue par la résolution aux moindres carrés de la minimisation de $\mathcal{A}$ (équation (3.2)). Ceci nous permet de modifier l'algorithme existant en y ajoutant simplement le module de régularisation. Nous suivrons ensuite le schéma suivant, pour un nombre n_{reg} d'itérations :

$$\mathbf{v}^{r,i+1} = (1 - \epsilon(M^r + \lambda H^r))\mathbf{v}^{r,i} + \epsilon\mathbf{c}^r \tag{3.10}$$

ϵ doit être évalué de façon à ce que le schéma reste convergent, ce qui équivaut à obtenir une fonction contractante. Une condition suffisante est :

$$0 < \epsilon < \frac{2}{\max \mathrm{vp}(M^r + \lambda H^r)}. \tag{3.11}$$

Si nous faisons la supposition que le schéma d'approximation de la dérivée partielle est indépendant de la direction de la dérivation, de la composante et du point considérés, les valeurs propres de H^r sont toutes bornées par $\kappa = 4N$. L'équation (3.11) devient alors :

$$0 < \epsilon < \frac{2}{\max \mathrm{vp}(M^r) + \lambda\kappa} \tag{3.12}$$

Par mesure de sécurité, on pourra prendre ϵ égal au quart de sa borne supérieure.

Il faut noter que, pour faire varier la régularité introduite le long des échelles, il paraît plus raisonnable de modifier λ/λ_0 plutôt que n_{reg}, car il est difficile d'estimer la vitesse de convergence du schéma.

3.3.2. Application au cas du champ de vecteurs

Notre objectif est bien évidemment d'appliquer le formalisme ainsi défini à la méthode d'estimation du mouvement et au champ de vecteurs résultant. Dans un tel cadre, tout se passe bien. L'obtention des valeurs propres des $M_\mathbf{p}^r$ est faite en résolvant des polynômes caractéristiques du second degré. Nous pouvons approximer les dérivées partielles par :

$$\frac{\partial \mathbf{v}_{l,k}}{\partial x} = \mathbf{v}_{l+1,k} - \mathbf{v}_{l,k}.$$

Le calcul des dérivées secondes par H^r devient donc de la forme :

$$\frac{\partial^2 \mathbf{v}_{l,k}}{\partial x^2} = \mathbf{v}_{l+1,k} - 2\mathbf{v}_{l,k} + \mathbf{v}_{l-1,k}$$

[2]Notre objectif est d'améliorer les vecteurs, et non pas d'obtenir le minimum exact. Nous pouvons donc nous arrêter largement avant d'avoir atteint la convergence [38]. C'est pourquoi nous n'avons pas utilisé d'algorithme de minimisation plus sophistiqué, contrairement au cas de l'algorithme du chapitre 4.

De plus, dans ce cas, nous obtenons comme particularisation de (3.9) et de (3.12) :

$$\lambda_0 = \frac{\sum_{l,k}(M^r_{l,k,xx} + M^r_{l,k,yy})}{8N^r_x N^r_y} \tag{3.13a}$$

$$\epsilon < \frac{2}{\max \text{vp}\, M^r + 8\lambda} \tag{3.13b}$$

où N_x et N_y sont les dimensions de la grille à l'échelle considérée. Enfin, le schéma (3.10) devient :

$$\begin{aligned} \mathbf{v}^{r,i+1}_{l,k,x} = \mathbf{v}^{r,i}_{l,k,x} &- \epsilon(M^r_{l,k,xx}\mathbf{v}^{r,i}_{l,k,x} + M^r_{l,k,xy}\mathbf{v}^{r,i}_{l,k,y}) \\ &- \epsilon\lambda(\mathbf{v}^{r,i}_{l-1,k,x} - 2\mathbf{v}^{r,i}_{l,k,x} + \mathbf{v}^{r,i}_{l+1,k,x}) \\ &+ \epsilon c^r_{l,k,x} \end{aligned} \tag{3.14}$$

$$\begin{aligned} \mathbf{v}^{r,i+1}_{l,k,y} = \mathbf{v}^{r,i}_{l,k,y} &- \epsilon(M^r_{l,k,yx}\mathbf{v}^{r,i}_{l,k,x} + M^r_{l,k,yy}\mathbf{v}^{r,i}_{l,k,y}) \\ &- \epsilon\lambda(\mathbf{v}^{r,i}_{l,k-1,y} - 2\mathbf{v}^{r,i}_{l,k,y} + \mathbf{v}^{r,i}_{l,k+1,y}) \\ &+ \epsilon \mathbf{c}^r_{l,k,y} \end{aligned} \tag{3.15}$$

Les échelles (qui sont à la fois celles de la bases d'ondelette et celles de la densité du champ) sont parcourues de la plus grossière à la plus fine jusqu'à avoir un vecteur pour chaque carré de 2 pixels sur 2.

3.3.3. Détail du protocole de tests et résultats

Afin de tester l'efficacité de la régularisation du champ de vecteur, une expérience simplifiée de compression a été effectuée. L'objectif est de mesurer l'entropie $\mathcal{H}_v$ de la transformée en ondelettes quantifiée du champ, ce qui nous donnera une idée du coût de son encodage, l'entropie $\mathcal{H}_{err}$ de la transformée en ondelettes de l'image d'erreur obtenue par compensation grâce au champ après déquantification et transformée inverse, ce qui nous donnera une première idée de la précision des vecteurs ainsi obtenus et enfin le rapport S pic du signal-bruit (PSNR, Peak Signal to Noise Ratio), ce qui nous donnera une idée finale de la qualité des vecteurs. À partir de $\mathcal{H}_v$ et de $\mathcal{H}_{err}$, il est possible de calculer $\mathcal{H}_{total}$ qui représente le coût moyen du codage d'un pixel (mouvement et erreur) par

$$\mathcal{H}_{total} = 0.25\mathcal{H}_v + \mathcal{H}_{err}.$$

L'expérience est donc la suivante (voir figure 3.3) :

- choix de λ/λ_0, de n_{reg}
- estimation du champ de mouvement v entre I_t et I_{t-1}
- transformation par ondelettes de v (ici, les ondelettes de Daubechies 7/9) : $\tilde{v}$
- quantification de pas Δ_v du champ $\tilde{v}$: $Q_1(\tilde{v})$
- calcul de l'entropie $\mathcal{H}_v$
- déquantification de $Q_v(\tilde{v})$: $\bar{\tilde{v}}$

- transformation inverse de $\bar{\bar{v}} : \bar{v}$
- calcul de l'image prédite I_t^{pred} à partir de I_{t-1} et de $\bar{v}$
- calcul de l'image d'erreur $\mathcal{E}_t$ entre I_t^{pred} et I_t
- transformation par ondelettes de $\mathcal{E}_t$ (ici, les ondelettes de Daubechies 7/9) : $\tilde{\mathcal{E}}_t$
- quantification de pas Δ_{err} du champ $\tilde{\mathcal{E}}_t$: $Q_{err}(\tilde{\mathcal{E}}_t)$
- calcul de l'entropie $\mathcal{H}_{err}$
- déquantification de $Q_{err}(\tilde{\mathcal{E}}_t)$: $\bar{\bar{\mathcal{E}}}_t$
- transformation inverse de $\bar{\bar{\mathcal{E}}}_t$: $\bar{\mathcal{E}}_t$
- détermination de l'image "décodée" $\bar{I}_t$ à l'aide de I_t^{pred} et de $\bar{\mathcal{E}}_t$
- mesure du PSNR entre $\bar{I}_t$ et I_t

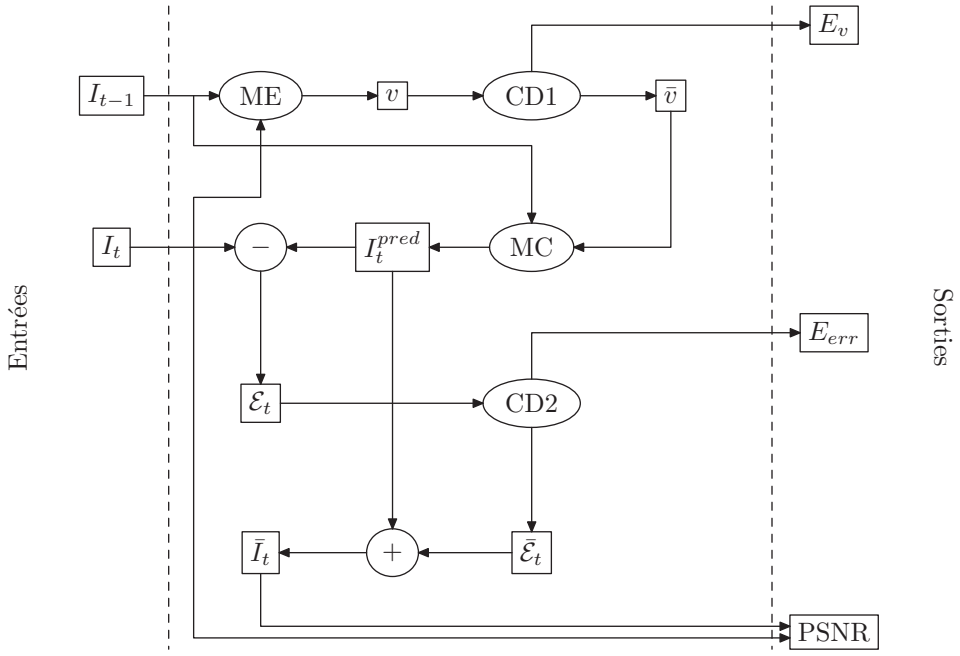


FIG. 3.3.: Schéma de principe de la simulation de codage. ME : méthode d'estimation de mouvement. MC : compensation du mouvement. CDi : séquence transformation en ondelettes, quantification Q_i avec un paramètre Δ_i, évaluation de l'entropie, déquantification, transformée en ondelettes inverse.

La transformée en ondelettes des images s'est faite sur 4 échelles pour les images CIF (352×288 pixels), et 3 échelles pour les images QCIF (176×144 pixels). La transformée en ondelettes du champ s'est faite sur une échelle de moins, c'est-à-dire respectivement 3 et 2. Nous avons pris comme valeur pour ϵ le quart de la borne supérieure (voir équation (3.13b)).

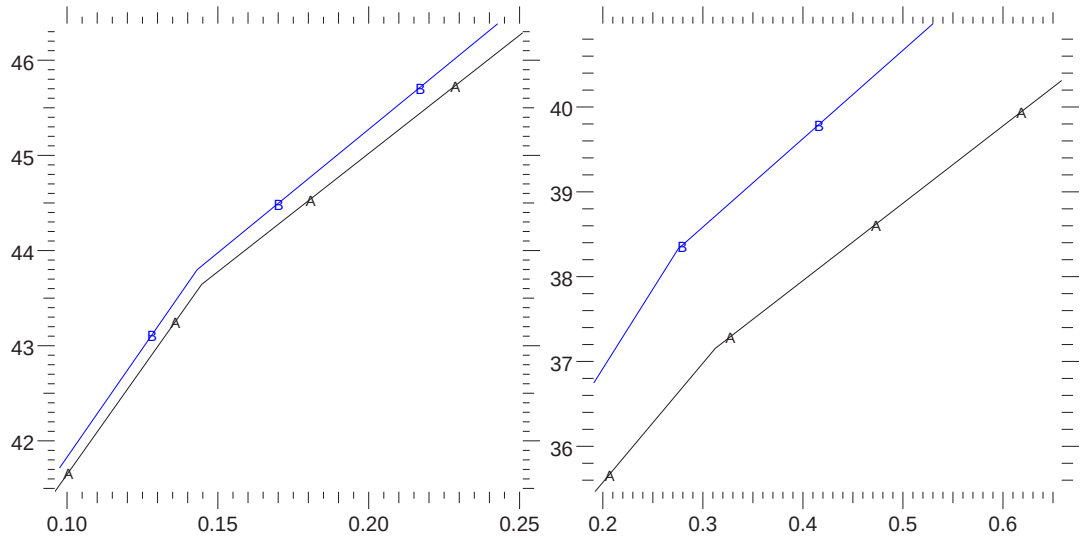
Les séquences de test sont présentés dans la table 3.1.

Nom	Nombre de trames codées	Format
News	300	QCIF
Container	300	QCIF
Foreman	300	QCIF
Silent	300	QCIF
Mobile	151	CIF
Paris	201	CIF
Tempete	260	CIF
Bus	144	CIF

TAB. 3.1.: Liste des séquences de test

3.3.3.1. Comparaison du schéma d'interpolation

Nous avons expérimenté les splines d'ordre 1 (équivalentes à l'interpolation bilinéaire) et d'ordre 3. Les splines d'ordre 3 ($n_{spl} = 3$) donnent le meilleur résultat en terme de qualité de prédiction, pour un temps de calcul quasiment équivalent aux splines d'ordre 1 ($n_{spl} = 1$). Dans le cas de la séquence *Foreman*, le gain est significatif (supérieur à 1 dB). Il est moins important dans les autres séquences, mais reste toujours positif.

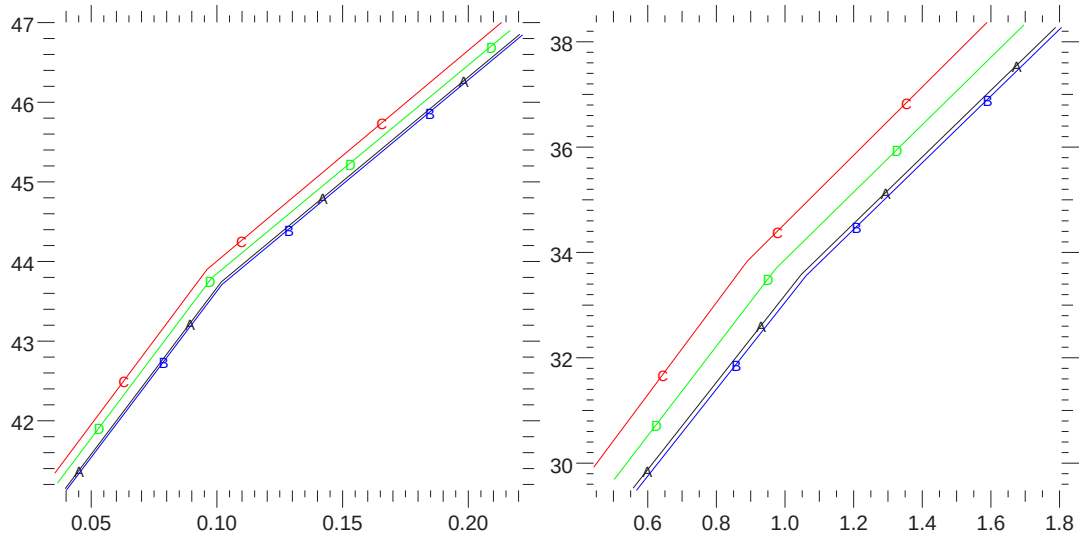


(a) *News* (b) *Foreman*

FIG. 3.4.: PSNR $= f(\mathcal{H}_{total})$. Pour cette expérience, $\lambda/\lambda_0 = 0.25$, $\Delta_v = 1$, $n_{reg} = 1000$. Δ_{err} prend les valeurs 32, 16, 8. Courbe A/noire : $n_{spl} = 1$; courbe B/bleue : $n_{spl} = 3$

3.3.3.2. Choix du paramètre λ/λ_0

Nous avons testé 2 valeurs de λ/λ_0 : 0.25, 0.125. Si la différence entre les deux valeurs ne se fait pas sentir pour un nombre d'itérations faible, en revanche, un rapport λ/λ_0 de 0.25 donne de meilleurs résultats lorsque nous augmentons le nombre d'itérations (voir figure 3.5). Ceci montre que donner du poids à la fonctionnelle de régularisation nous permet d'obtenir plus rapidement un champ de meilleure qualité du point de vue de la compression.



(a) *News* (b) *Bus*

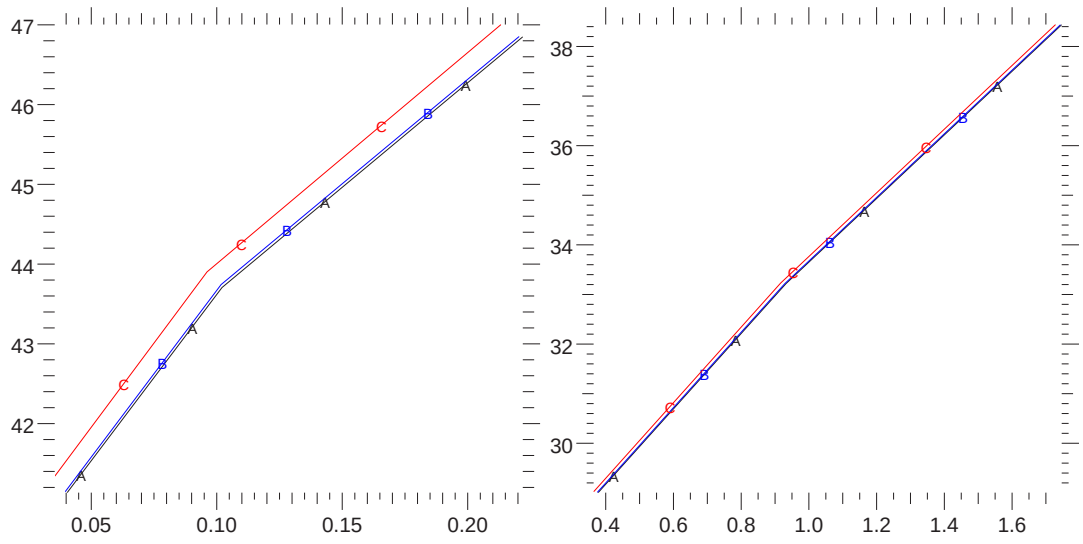
FIG. 3.5.: PSNR $= f(\mathcal{H}_{total})$. Pour cette expérience, $\Delta_v = 4$, $n_{spl} = 1$,. Δ_{err} prend les valeurs 32, 16, 8. Courbe A/noire : $n_{reg} = 100$ et $\lambda/\lambda_0 = 0.25$; courbe B/bleue : $n_{reg} = 100$ et $\lambda/\lambda_0 = 0.125$; courbe C/rouge : $n_{reg} = 1000$ et $\lambda/\lambda_0 = 0.25$; courbe D/verte : $n_{reg} = 1000$ et $\lambda/\lambda_0 = 0.125$.

3.3.3.3. Nombre d'itérations

100 itérations donnent un début de résultat satisfaisant, pour un surplus de temps de calcul très léger : 20% de temps en plus par rapport à l'algorithme dépourvu du module de régularisation. Mis à part pour les séquences *Mobile* et *Paris*, passer à 1000 itérations améliore la qualité du champ, au prix d'un surcoût notable en temps de calcul, puisque celui-ci triple. Ceci est valable aussi bien lorsque $\lambda/\lambda_0 = 0.25$ que quand $\lambda/\lambda_0 = 0.125$ (voir figures 3.6 et 3.7).

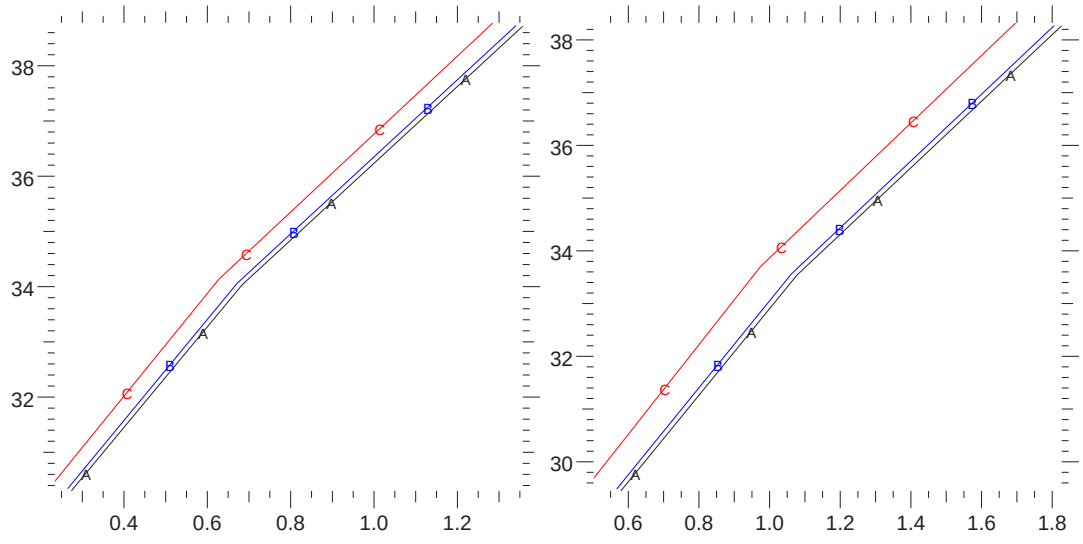
3.3.3.4. Quantification du champ de mouvement

Intéressons-nous à l'influence de la quantification du champ de mouvement. Fort logiquement, plus nous quantifions durement le champ de mouvement, plus celui-ci est



(a) *News* (b) *Mobile*

FIG. 3.6.: PSNR $= f(\mathcal{H}_{total})$. Pour cette expérience, $\lambda/\lambda_0 = 0.25$, $n_{spl} = 1$, $\Delta_v = 4$. Δ_{err} prend les valeurs 32, 16, 8. Courbe A/noire : $n_{reg} = 0$; courbe B/bleue : $n_{reg} = 100$; courbe C/rouge : $n_{reg} = 1000$.

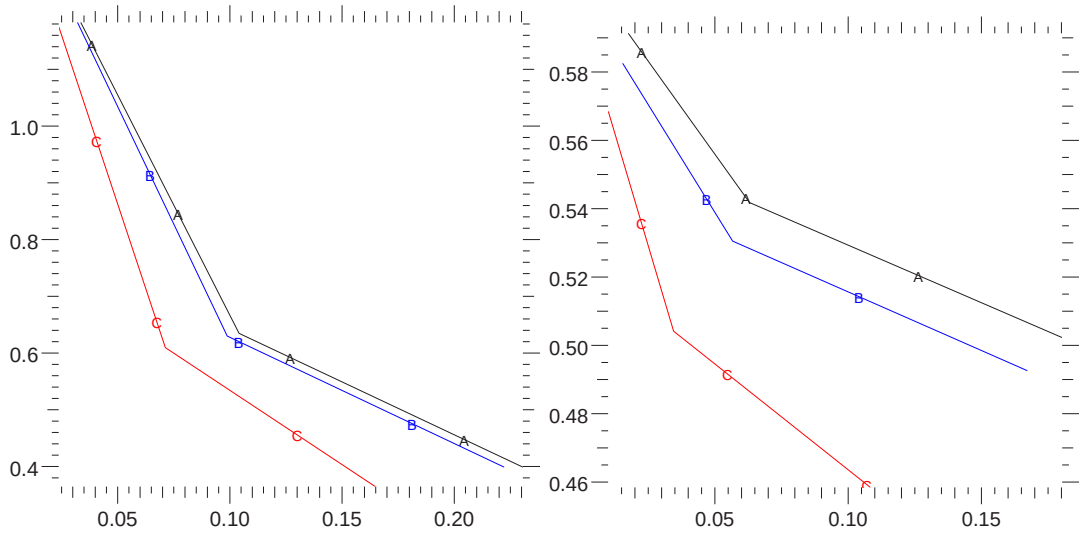


(a) *Tempete* (b) *Bus*

FIG. 3.7.: PSNR $= f(\mathcal{H}_{total})$. Pour cette expérience, $\lambda/\lambda_0 = 0.125$, $n_{spl} = 1$, $\Delta_v = 4$. Δ_{err} prend les valeurs 32, 16, 8. Courbe A/noire : $n_{reg} = 0$; courbe B/bleue : $n_{reg} = 100$; courbe C/rouge : $n_{reg} = 1000$.

inexact, et donc plus la qualité de l'image d'erreur se dégrade. Dans le même temps, l'entropie du champ diminue. Nous avons donc une décroissance de $\mathcal{H}_{err}$ quand $\mathcal{H}_v$ augmente (figure 3.8). Cette décroissance fait que le coût de codage total par pixel, $\mathcal{H}_{total}$, n'est pas nécessairement une fonction monotone. Si, pour la séquence *Foreman*, nous observons que nous pourrions encore augmenter le poids du champ dans le coût de codage pour diminuer le coût total de codage, en revanche, la valeur $\Delta_v = 2$ est celle qui minimise le coût de codage (parmi les valeurs essayées) pour la séquence *Paris* (figure 3.9).

Si nous étudions les courbes donnant le PSNR en fonction du coût total de codage, nous constatons que le choix du pas optimal de quantification dépend du contenu de la séquence. Dans la séquence *News*, la structure du champ permet de quantifier celui-ci durement, tandis que dans le cas de la séquence *Foreman*, nous avons intérêt à donner de la place au champ dans la façon de coder un pixel (figure 3.10).

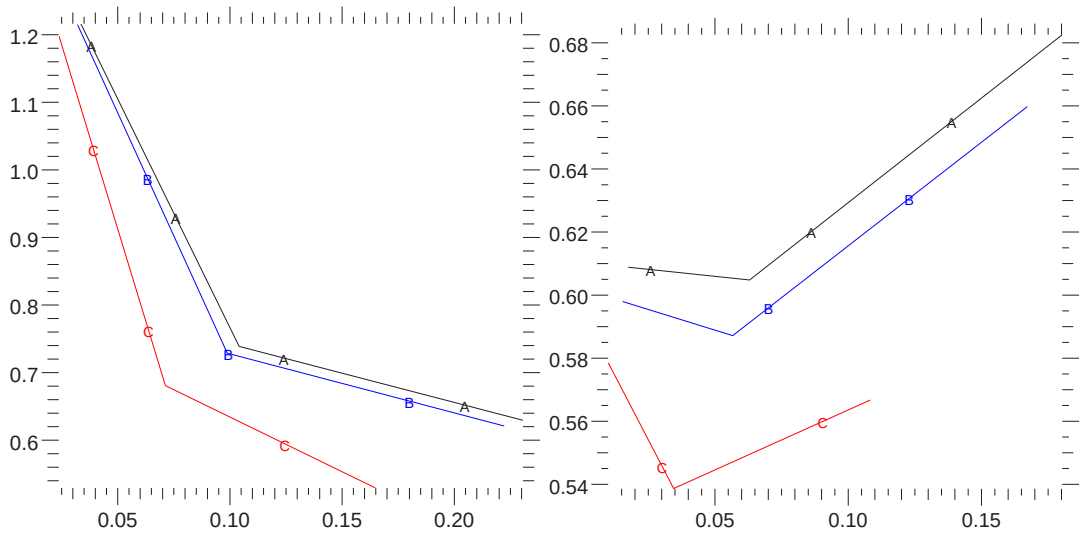


(a) *Foreman* (b) *Paris*

FIG. 3.8.: $\mathcal{H}_{err} = f(\mathcal{H}_v)$. Pour cette expérience, $\lambda/\lambda_0 = 0.25$, $n_{spl} = 3$, $\Delta_{err} = 8$. Δ_v prend les valeurs 4, 2, 1. Courbe A/noire : $n_{reg} = 0$; courbe B/bleue : $n_{reg} = 100$; courbe C/rouge : $n_{reg} = 1000$.

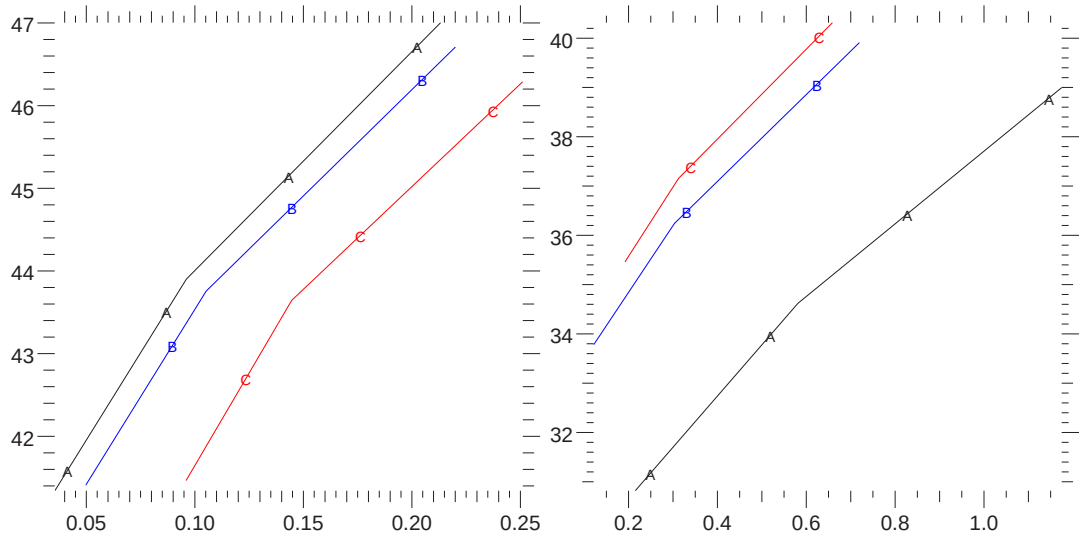
3.3.3.5. Bilan de l'étude des paramètres

Lors des expériences, nous avons constaté que nous pouvions donner pour certains paramètres, dont les paramètres de la méthode d'estimation (n_{reg}, λ/λ_0), des valeurs correctes indépendamment des séquences. En revanche, les paramètres de quantification, qui vont régler le poids de l'information que nous transmettrons sur le champ de mouvement par rapport à celui des coefficients, sont plus difficiles à établir, dans la mesure où ils dépendent du contenu de la séquence. Nous avons besoin d'un mécanisme d'allocation de débit entre codage du champ et codage des coefficients d'erreur qui trouve la précision



(a) *Foreman* (b) *Paris*

FIG. 3.9.: $\mathcal{H}_{total} = f(\mathcal{H}_v)$. Pour cette expérience, $\lambda/\lambda_0 = 0.25$, $n_{spl} = 3$, $\Delta_{err} = 8$. Δ_v prend les valeurs 4, 2, 1. Courbe A/noire : $n_{reg} = 0$; courbe B/bleue : $n_{reg} = 100$; courbe C/rouge : $n_{reg} = 1000$.



(a) *News* (b) *Foreman*

FIG. 3.10.: PSNR $= f(\mathcal{H}_{total})$. Pour cette expérience, $\lambda/\lambda_0 = 0.25$, $n_{reg} = 1000$, $n_{spl} = 1$. Δ_{err} prend les valeurs 32,16,8. Courbe A/noire : $\Delta_v = 4$; courbe B/bleue : $\Delta_v = 2$; courbe C/rouge : $\Delta_v = 1$

Nom	Format	PSNR Block-matching	PSNR Bernard régularisée
News	QCIF	41.71	40.39
Container	QCIF	44.26	44.35
Foreman	QCIF	37.44	37.10
Silent	QCIF	46.14	45.12
Mobile	CIF	29.37	29.23
Paris	CIF	35.90	34.18
Tempete	CIF	31.42	28.56
Bus	CIF	29.17	23.52

TAB. 3.2.: PSNR : block-matching contre méthode de Bernard régularisée

optimale pour chacun des deux éléments. Actuellement, de tels mécanismes de changement automatique de la précision de codage du champ de mouvement sont encore peu répandus dans les codeurs de test des standards, même si des champs de mouvements progressifs font leur apparition.

3.3.3.6. Comparaison avec le block-matching

Comme notre but était d'étudier la viabilité d'une dérivation de la méthode de Bernard pour la compression vidéo, nous avons effectué un test simple de comparaison avec le block-matching. Nous avons utilisé un algorithme de block-matching simple, avec une zone de recherche de 16 pixels, une précision au quart de pixel, et une taille de bloc de 8 pixels sur 8. Nous avons effectué la comparaison avec notre méthode, en utilisant 1000 itérations, et un rapport $\lambda/\lambda_0 = 0.125$. Dans les deux cas, l'interpolation est effectuée à l'aide de splines d'ordre 3. Nous mesurons le PSNR de la trame compensée par rapport à la trame courante, moyennée sur les 20 premières trames de la séquence. Les résultats sont donnés dans la table 3.2.

Nous voyons donc que le block-matching fait généralement mieux que notre méthode, les écarts variant de -0.09 dB à 5.65 dB. Ceci est d'autant plus problématique que nous générons 16 fois plus de vecteurs que celui-ci, comme nous obtenons un vecteur par bloc de 2 pixels sur 2, au lieu d'un seul par bloc de 8 pixels sur 8 pour le block-matching. La méthode expérimentée ne sera donc pas compétitive pour la compression vidéo, dans la mesure où il faut transmettre non seulement la trame d'erreur mais aussi les vecteurs de mouvement. Bien que ceux que nous générons bénéficient d'une certaine régularité, celle-ci ne peut pas compenser leur grand nombre.

3.4. Conclusion

L'approche initiale de ce travail était d'améliorer autant que possible les résultats de la méthode de Bernard en terme de PSNR. Dans ce cadre, l'ajout d'un module de régularisation apporte une amélioration claire, en fournissant des résultats plus probants, débarrassés du bruit qui rend un champ dense inefficace pour la compression. Néanmoins,

la méthode sur laquelle nous nous appuyons essaye de minimiser une fonctionnelle d'erreur correspondant à l'équation du flot optique projeté. Si le champ véritable que nous cherchons à estimer correspond effectivement au modèle sur lequel nous nous appuyons (champ localement constant), le choix de la fonctionnelle aurait été neutre. En réalité, notre modèle n'est pas capable d'exprimer le champ réel. Dans ces conditions, toutes les fonctionnelles ne sont pas équivalentes. Le block-matching, qui minimise explicitement la mesure d'erreur utilisée pour la compression, possède donc un avantage sur notre méthode qui s'intéresse à la validité de l'équation du flot optique. Il n'est pas étonnant que celle-ci fournisse des PSNR inférieurs à ceux obtenus avec le block-matching. Dans le cadre d'une estimation d'un champ de mouvement (ou plus généralement d'une donnée quelconque), il faut clairement avoir en tête l'application qui est visée, et définir une mesure de qualité qui soit adéquate par rapport à celle-ci et que nous minimiserons explicitement.

Chapitre 4.

Estimation du mouvement par détermination de nœuds d'une grille

Les résultats décevants de la méthode exposée dans le chapitre 3 nous poussent à rechercher une autre manière d'estimer le mouvement. Auparavant, nous avons essayé de minimiser une fonctionnelle qui n'était que formellement équivalente à l'objectif réel de minimisation, et à la mesure que nous utilisions. Si nous souhaitons éliminer ce défaut, il nous faut donc essayer de minimiser explicitement le PSNR. C'est finalement la technique qui est utilisée dans les codeurs de tests, par exemple celui de H.264 qui tente explicitement de minimiser la somme du coût de l'erreur (après transformation) et du codage des vecteurs de mouvement. Nous nous contenterons ici de la minimisation du PSNR, la minimisation du coût de codage du champ de mouvement étant dépendante de la méthode de codage et du modèle de répartition de mouvement que l'on se donne.

4.1. Représentation du mouvement par déformation d'une grille

4.1.1. Formulation du problème

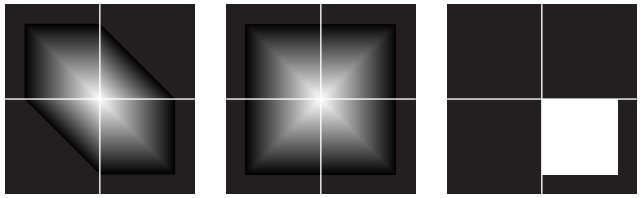
Parmi les nombreuses façons de représenter le champ de mouvement, une des formulations assez générales est de le décrire en le décomposant sur une base de fonctions. Celle-ci est souvent une base d'interpolation, auquel cas on peut associer chaque fonction ϕ_{ij} de la base au point $\mathbf{x}_{ij}$ de l'image. Les points $\mathbf{x}_{ij}$ sont généralement les "nœuds" d'une grille répartissant ceux-ci uniformément sur l'image, bien que, dans un modèle général, ce ne soit pas obligatoire. Nous avons alors :

$$\mathbf{v}(\mathbf{x}) = \sum_{i,j} \begin{bmatrix} v_{ij1} \\ v_{ij2} \end{bmatrix} \phi_{ij}(\mathbf{x}) \tag{4.1}$$

Parmi les modèles de champs de mouvement qu'il est possible d'incorporer dans cette formulation, citons, de façon non-exhaustive :

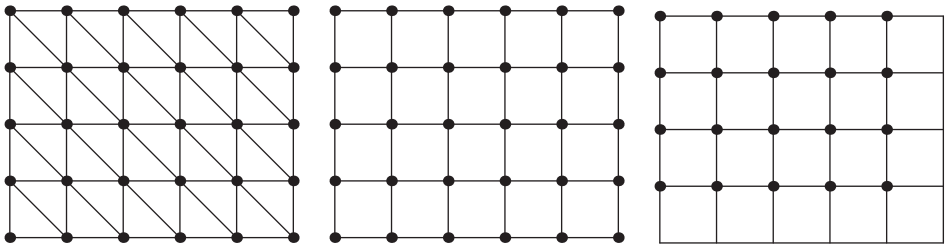
- les modèles de champs constants par bloc (block-matching, BM), où les fonctions ϕ_{ij} sont constantes sur un carré et nulles ailleurs
- les modèles du type Control Grid Interpolation (CGI), bilinéaires par morceau [47, 35]

- les modèles du type Triangle Motion Compensation (TMC), où le mouvement à l'intérieur d'un triangle est calculé comme le barycentre du mouvement aux sommets du triangle [6, 33, 34]
- les modèles affines de compensation globale du mouvement



(a) TMC (b) CGI (c) BM

FIG. 4.1.: Différentes fonctions ϕ_{00} possibles.



(a) TMC (b) CGI (c) BM

FIG. 4.2.: Grilles de points $\mathbf{x_{ij}}$ associées aux fonctions ϕ_{00}

Notre objectif est d'estimer les paramètres de notre champ de mouvement. Comme il a été souligné précédemment, il s'agit d'un problème sous-contraint. L'application à laquelle la mesure est destinée permet de rajouter les contraintes adéquates. Dans le cadre de la compression video, la minimisation de la somme des erreurs entre l'image prédite à l'aide du champ de mouvement et l'image de référence selon une norme donnée est un bon critère d'optimalité.

$$[v]_{opt} = \underset{[v]}{\operatorname{argmin}} \sum_{\mathbf{x}} \| \big(I_{t+1}(\mathbf{x} + \mathbf{v}(\mathbf{x})) - I_t(\mathbf{x}) \big) \|_A \tag{4.2}$$

$\|\cdot\|_A$ est généralement $\|\cdot\|_1$ ou $\|\cdot\|_2$, cas dans lequel on se placera pour la suite des calculs, puisqu'il correspond au PSNR que nous souhaitons explicitement minimiser.

L'association des deux équations (4.1) et (4.2) nous donne :

$$[v]_{opt} = \underset{[v]}{\operatorname{argmin}} \sum_{\mathbf{x}} \left(I_{t+1}\Big(\mathbf{x} + \sum_{i,j} \phi_{ij}(\mathbf{x}) \begin{bmatrix} v_{ij1} \\ v_{ij2} \end{bmatrix} \Big) - I_t(\mathbf{x}) \right)^2 \tag{4.3}$$

4.1.2. Méthode de résolution

Éliminons tout d'abord un cas particulier de notre analyse : le cas où le vecteur $\mathbf{v}$ de chaque point $\mathbf{x}$ donné ne dépend que d'un seul des paramètres. Dans ces conditions, chaque paramètre peut être optimisé indépendamment des autres, ce qui favorise une approche exhaustive pour la résolution du problème. Une illustration typique de cette situation est le block-matching, où le champ est constant par bloc. Le mouvement de chaque bloc est donc défini par un unique paramètre et la recherche semi-exhaustive reste la plus utilisée dans la mesure où elle rentre dans les temps de calcul des machines.

Retournons au cas général. Nous cherchons donc ici à effectuer une minimisation aux moindres carrés d'un modèle $\chi^2([v_{ijl}])$ *a priori* non-linéaire par rapport aux paramètres, où :

$$\chi^2\Big(\,[v]\,\Big) = \sum_{\mathbf{x}} \left(I_{t+1}\Big(\mathbf{x} + \sum_{ij} \phi_{ij}(\mathbf{x}) \begin{bmatrix} v_{ij1} \\ v_{ij2} \end{bmatrix}\Big) - I_t(\mathbf{x}) \right)^2 . \tag{4.4}$$

La méthode couramment employée est celle de Levenberg-Marquardt. L'objectif de cet algorithme (Algorithme 1) est d'opérer un glissement entre la méthode des plus grandes pentes, où les paramètres sont déplacés dans la direction du gradient, et les méthodes quasi-newtoniennes, qui nécessitent l'inversion du hessien, plus fines lorsqu'on s'approche d'une approximation correcte de l'optimum.

Si nous disposons d'une solution proche de la solution recherchée, $\chi^2([v_{ijl}])$ sera normalement bien approchée par une fonctionnelle quadratique :

$$\chi^2([v]) = \gamma - \nabla\chi^2 \cdot [v] + \frac{1}{2}\,[v] \cdot \mathrm{H}\chi^2 \cdot [v] \tag{4.5}$$

Si effectivement l'approximation est correcte, alors les méthodes quasi-newtonniennes nous suggèrent de faire évoluer notre solution temporaire $[v]_{\text{courant}}$ par :

$$[v]_{\text{suivant}} = [v]_{\text{courant}} + D^{-1}(-\nabla\chi^2([v]_{\text{courant}})). \tag{4.6}$$

En revanche, si notre approximation est mauvaise, nous avons plutôt intérêt à nous déplacer dans la direction du gradient, c'est-à-dire à effectuer :

$$[v]_{\text{suivant}} = [v]_{\text{courant}} - \text{constante} \cdot \nabla\chi^2([v]_{\text{courant}}), \tag{4.7}$$

en nous arrangeant pour que la constante ne nous fasse pas dépasser la zone de descente du gradient. Nous pouvons essayer d'utiliser chacun de ces deux types d'étapes afin de minimiser notre fonctionnelle.

Notons

$$\mathbf{x}' = \mathbf{x} + \mathbf{v}(\mathbf{x}). \tag{4.8}$$

$\mathbf{x}'$ contient donc l'influence de notre champ de mouvement, si nous réécrivons notre fonctionnelle comme :

$$\chi^2(v) = \sum_{\mathbf{x}} \big(I_{t+1}(\mathbf{x}') - I_t(\mathbf{x})\big)\,.$$

La dérivée de notre critère par rapport à chacun des v_{ijl} s'écrit :

$$\frac{\partial \chi^2}{\partial v_{ijl}} = 2 \sum_{\mathbf{x}} \phi_{ij}(\mathbf{x}) \frac{\partial I_{t+1}}{\partial x_l}(\mathbf{x}') \Big(I_{t+1}(\mathbf{x}') - I_t(\mathbf{x}) \Big) \tag{4.9}$$

En dérivant une seconde fois, on obtient :

$$\frac{\partial^2 \chi^2}{\partial v_{ijl} \partial v_{i'j'l'}} = 2 \sum_{\mathbf{x}} (\phi_{ij}\phi_{i'j'})(\mathbf{x}) \left[\frac{\partial^2 I_{t+1}}{\partial x_l \partial x_{l'}}(\mathbf{x}') \Big(I_{t+1}(\mathbf{x}') - I_t(\mathbf{x}) \Big) + \Big(\frac{\partial I_{t+1}}{\partial x_l} \frac{\partial I_{t+1}}{\partial x_{l'}} \Big)(\mathbf{x}') \right] \tag{4.10}$$

En pratique, on considère que le premier terme est négligeable, ce qui nous permet d'écrire finalement :

$$\frac{\partial^2 \chi^2}{\partial v_{ijl} \partial v_{i'j'l'}} = 2 \sum_{\mathbf{x}} (\phi_{ij}\phi_{i'j'})(\mathbf{x}) \Big(\frac{\partial I_{t+1}}{\partial x_l} \frac{\partial I_{t+1}}{\partial x_{l'}} \Big)(\mathbf{x}') \tag{4.11}$$

Afin d'écrire de façon plus compacte, notons :

$$\beta_k = \frac{\partial \chi^2}{\partial v_{ijl}}$$
$$\alpha_{kk'} = \frac{1}{2} \frac{\partial^2 \chi^2}{\partial v_{ijl} \partial v_{i'j'l'}},$$

où k et k' sont des indices agrégés correspondant respectivement à ijl et $i'j'l'$.

L'équation (4.6) se réécrit alors :

$$\sum \alpha_{kk'} \delta v_{k'} = \beta_k, \tag{4.12}$$

tandis que l'équation (4.7) devient :

$$\delta v_k = \text{constant} \times \beta_k \tag{4.13}$$

L'algorithme de Levenberg-Marquardt s'appuie sur deux idées complémentaires :

- Afin de trouver une valeur correcte de la constante dans l'équation (4.7), il est possible de se servir du hessien. En effet, $1/\alpha_{kk}$ est de la dimension de β_k. Nous pouvons introduire une constante sans dimension, λ, identique pour chacun des indices k, et remplacer l'équation (4.13) par :

$$\delta v_k = \frac{1}{\lambda \alpha_{kk}} \beta_k \qquad\qquad \lambda \alpha_{kk} \delta v_k = \beta_k. \tag{4.14}$$

  Ceci est possible dans la mesure où la définition de α_{kk} nous garantit que ce nombre est positif. Si α_{kk} est nul, alors le gradient est nul lui aussi, et nous aurons un pas nul.

– Il est possible de combiner les équations (4.12) et (4.14) en définissant une nouvelle matrice $[\alpha']$ à partir de $[\alpha]$:

$$\begin{aligned} \alpha'_{kk} &= (1+\lambda)\alpha_{kk} \\ \alpha'_{kk'} &= \alpha_{kk'} \qquad\qquad \text{si } k \neq k'. \end{aligned}$$

Nous avons alors :

$$\sum \alpha'_{kk'} \delta v_{k'} = \beta_k \tag{4.15}$$

Selon la valeur de λ, la résolution de (4.15) sera plus proche de l'approche quasi-newtonienne ou bien de la descente de gradient. Si λ est important, la matrice $[\alpha']$ sera à diagonale dominante, et nous aurons un système proche de (4.14). Si au contraire λ est faible, les matrices $[\alpha]$ et $[\alpha']$ ne seront que peu différentes, et nous aurons un système proche de (4.12).

Dans ces conditions, nous obtenons l'algorithme 1.

Algorithme 1 Obtention du champ de mouvement via l'algorithme de Levenberg-Marquardt

initialisation de $[v]$
initialisation de λ
répéter
  $\delta[v] \leftarrow (\mathrm{H}\chi^2([v]) + \lambda Id)^{-1} \nabla\chi^2([v])$
  si $\chi^2([v] + \delta[v]) < \chi^2([v])$ **alors**
    $\lambda \leftarrow \lambda/10$
    $[v] \leftarrow [v] + \delta[v]$
  sinon
    $\lambda \leftarrow 10\lambda$
  fin si
jusqu'à $\chi^2 < \varepsilon_1$ ou $\delta\chi^2 < \varepsilon_2$

4.1.3. Interpolation de l'image et de sa dérivée

Nous avons supposé jusqu'à présent que l'image au temps $t+1$ était continue et donc que nous connaissions ses valeurs pour des valeurs quelconques du champ de mouvement. Il nous faut revenir aux données dont nous disposons, c'est–à–dire une image échantillonnée. Le choix d'une méthode d'interpolation nous permettra de passer de nos données discrètes à une image continue. Celle-ci devra être régulière dans la mesure où nous devons calculer le gradient de l'image en des points quelconques. L'interpolation par B-splines bicubiques présente de cette qualité.

On calcule d'abord les coefficients de développement B-spline m_{ij}, par un algorithme récursif.

On a ensuite par définition de ces coefficients, pour un point $\mathbf{q}$ quelconque :

$$I_{t+1}(\mathbf{q}) = \sum_{i,j} m_{ij}\psi_i(q_x)\psi_j(q_y) = \sum_{i,j} m_{ij}\Psi_{i,j}(\mathbf{q}) \tag{4.16}$$

Et donc :

$$\nabla I_{t+1}(\mathbf{q}) = \sum_{i,j} m_{ij}\nabla\Psi_{i,j}(\mathbf{q}) \tag{4.17}$$

Les calculs de décomposition sont effectués une seule fois, pour toute l'image, et permettent ensuite de calculer simplement les valeurs de l'image et de sa dérivée en tout point.

4.1.4. Résolution multi-échelles

4.1.4.1. Principe

La plupart des méthodes de minimisation sont sensibles à l'existence de minima locaux. La procédure de minimisation peut s'arrêter dans un de ceux-ci qui valide la condition d'arrêt sur l'annulation du gradient et la positivité du hessien dans ses environs. Dans le cadre de l'estimation de mouvement, ces minima locaux sont nombreux, en particulier à cause de la texture des objets de la scène. Il nous faut donc trouver une parade.

Une des solutions classiques est d'utiliser des versions lissées des images. Le lissage va faire disparaître les petits minima locaux et faciliter l'approche de la solution optimale. Une fois obtenue une estimation de l'optimum sur les images lissées, la méthode repasse à des images plus nettes.

Plus précisément, nous allons travailler avec une multi-résolution de l'image et nous munir d'un jeu complet de versions simplifiées de nos deux images. Nous obtiendrons une première idée du mouvement avec la version la plus floue, puis nous nous servirons de la valeur obtenue pour initialiser l'algorithme 1 quand nous travaillerons sur la version un peu moins floue, et ainsi de suite, jusqu'à travailler sur l'image originale.

4.1.4.2. Sous-échantillonnage et sur-échantillonnage par rapport à l'échantillonnage critique

Il est désirable de rester modéré sur le temps de calcul afin que celui-ci "n'explose" pas. Cela veut dire que nous ne pouvons pas travailler sur une série de versions de l'image de taille identique à celle-ci. Pour éviter cette augmentation trop importante, il nous faut donc sous-échantillonner afin de travailler sur des versions filtrées, mais aussi réduites, de l'image.

Il faut cependant bien prendre en compte le problème de l'aliasage dû au sous-échantillonnage lorsque l'on effectue de l'estimation de mouvement associée à celui-ci.

Supposons que nous disposons d'un signal aléatoire $\boldsymbol{s}$, de moyenne nulle. Nous avons

$$E(\boldsymbol{s}) = 0.$$

Nous définissons le spectre en amplitude (désigné dans ce qui suit comme le spectre) de $\boldsymbol{s}$, $\mathrm{Sp}\,\boldsymbol{s}$, par :

$$\mathrm{Sp}\,\boldsymbol{s} = \mathrm{E}(|\widehat{\boldsymbol{s}}|).$$

Il s'agit donc de l'amplitude - au carré pour des raisons de simplicité - des coefficients de la décomposition de Fourier de notre signal.

Il est aussi possible de voir le spectre comme la transformée de Fourier de l'auto-corrélation du signal. Celle-ci se définit comme :

$$\mathrm{AutoCorr}(\boldsymbol{s})(x) = \mathrm{E}(\int s(t)\overline{s(t+x)}dt)$$

Grâce à la conservation du produit scalaire lors de la transformée de Fourier, nous avons aussi :

$$\begin{aligned}\mathrm{AutoCorr}(\boldsymbol{s})(x) &= \mathrm{E}(\int \widehat{s}(\omega)e^{ix\omega}\overline{s(\omega)}d\omega) \\ &= \mathcal{F}^{-1}\left(E(\omega \mapsto |\widehat{\omega}|^2)\right)(x)\end{aligned}$$

Nous avons donc bien :

$$\mathrm{Sp}\,\boldsymbol{s} = \widehat{\mathrm{AutoCorr}}\,\boldsymbol{s}$$

Que se passe-t-il si nous convoluons notre signal avec un filtre f ? Nous avons :

$$\begin{aligned}\mathrm{Sp}\,\boldsymbol{s} \star f &= \mathrm{E}(|\widehat{s}\widehat{f}|^2) \\ &= |\widehat{f}|^2 E(|\widehat{s}|^2) \\ &= \mathrm{Sp}\,f\,\mathrm{Sp}\,\boldsymbol{s}\end{aligned}$$

dans la mesure où f ne dépend pas de la réalisation de $\boldsymbol{s}$. La convolution par un filtre f donne donc une multiplication des spectres.

L'échantillonnage, lui, ne correspond pas à une convolution, mais à une multiplication par le peigne de Dirac adéquat. Le peigne de Dirac de période T se définit par :

$$\delta_T(x) = \sum_{k\in\mathbf{Z}} \delta(x - kT).$$

La transformée de Fourier du peigne de Dirac est elle-même un peigne de Dirac :

$$\mathcal{F}\delta_T = \frac{1}{T}\delta_{\frac{2\pi}{T}}.$$

Que se passe-t-il maintenant si nous échantillonnons ou sous-échantillonnons notre signal, c'est-à-dire si nous le multiplions par le peigne de Dirac de période T ?

Nous avons :

$$\begin{aligned}\mathrm{Sp}\,\boldsymbol{s}\delta_T &= \mathrm{E}(|\widehat{s} \star \frac{1}{T}\delta_{\frac{2\pi}{T}}|^2) \\ &= \mathrm{E}(\omega \mapsto |\frac{1}{T}\sum_{k\in\mathbf{Z}} \widehat{s}(\omega + \frac{2k\pi}{T})|) \\ &\leq \sum_{k\in\mathbf{Z}} \mathrm{E}(\omega \mapsto |\frac{1}{T}\widehat{s}(\omega + \frac{2k\pi}{T})|)\end{aligned}$$

Si notre signal a déjà un spectre inclus dans $[-\frac{\pi}{T}, \frac{\pi}{T}]$, l'échantillonnage effectue simplement une périodisation du spectre. En revanche, si le spectre n'est pas inclus dans cette gamme de fréquence ou bien n'est pas déjà périodique de période $\frac{2\pi}{T}$, les différents éléments de la somme vont interagir entre eux. Il y aura alors repliement spectral.

Si nous avons un signal échantillonné à T et que nous souhaitons le sous-échantillonner à $2T$, nous aurons un repliement spectral d'une moitié de l'intervalle de fréquence sur l'autre. Dans le cas d'un signal *a priori* quelconque, dont on suppose juste la décroissance du spectre sur $\mathbf{R}_+$, cela peut être dévastateur.

Comment obtenir une solution satisfaisante dans ces conditions ? Supposons que nous disposons d'un filtre passe-bas non-idéal h_T, qui atténue le spectre essentiellement hors de $[-\frac{\pi}{T}, \frac{\pi}{T}]$. Le spectre du signal s après un premier filtrage par convolution avec h est :

$$\mathrm{Sp}(\boldsymbol{s} \star h_T) = \mathrm{Sp}\, \boldsymbol{s}\, \mathrm{Sp}\, h_T.$$

Le filtre h_{2T} peut être vu comme le filtre h_T, dans lequel nous avons inséré des zéros entre chaque élément. Dans ces conditions, nous avons :

$$\widehat{h}_{2T}(\omega) = \widehat{h}_T(2\omega)$$

Si nous filtrons ensuite avec h_{2T}, nous obtenons comme spectre :

$$\mathrm{Sp}(\boldsymbol{s} \star h_T \star h_{2T}) = \mathrm{Sp}\, f\, \mathrm{Sp}\, h_T\, \mathrm{Sp}\, h_{2T}.$$

Posons

$$\begin{aligned}\widehat{k}_i(\omega) &= \prod_{j=0}^{i} \widehat{h}_T(2^j\omega) \\ &= \prod_{j=0}^{i} \widehat{h}_{2^jT}(\omega).\end{aligned}$$

Si l'on trouve i tel que :

$$|\widehat{k}_i(\omega)|^2 \ll 1 \qquad \text{si } \omega \in [\pi/2, \pi], \tag{4.18}$$

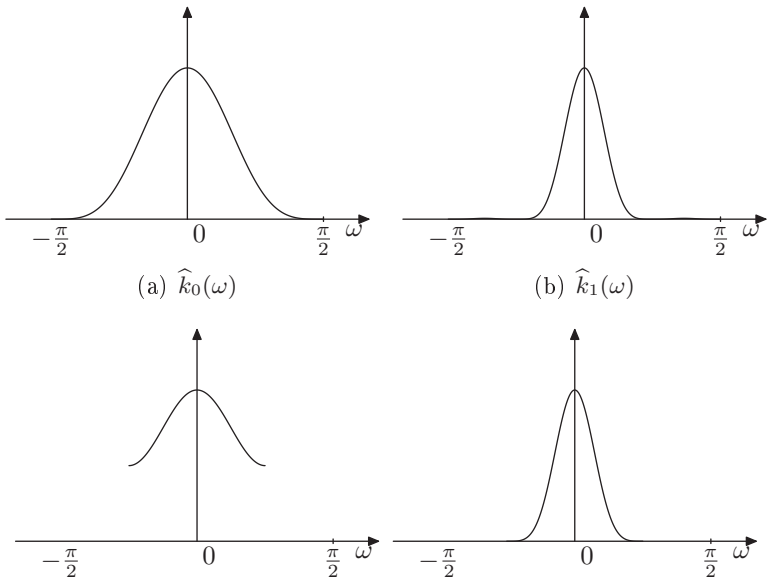
alors il sera possible de commencer à sous-échantillonner à partir du i-ème filtrage. En effet, l'équation (4.18) signifie que la partie du spectre qui se replie à cause de l'échantillonnage est a priori négligeable par rapport au spectre normal, si les deux parties étaient initialement de même ordre, et donc que le repliement ne causera pas de souci particulier.

En pratique, si nous utilisons un filtre de noyau $(1/4, 1/2, 1/4)$, que se passe-t-il ? Le spectre de ce filtre est :

$$\mathrm{Sp}(h)(\omega) = \cos^4 \frac{\omega}{2}.$$

Le repliement spectral risque encore de poser problème si le spectre du signal d'origine ne décroit pas suffisamment vite comme cela apparaît sur les figures 4.3(a) et 4.3(c). En

revanche, deux étapes de filtrage multiplient le spectre du signal par $\cos^4 \frac{\omega}{2} \cos^4 \omega$. Sur l'intervalle $[\pi/4, \pi/2]$, le maximum de cette fonction est inférieur à 0.01, à comparer au maximum absolu de 1. Le repliement spectral est donc peu important si l'on maintient un sur-échantillonnage de 2 par rapport à l'échantillonnage critique lors de la séquence de filtrage/sous-échantillonnage. Ceci correspond aux figures 4.3(b) et 4.3(d).



(a) $\widehat{k}_0(\omega)$ (b) $\widehat{k}_1(\omega)$

(c) $\widehat{k}_0(\omega)$ après repliement (d) $\widehat{k}_1(\omega)$ après repliement

FIG. 4.3.: Spectres de filtrage.

4.1.4.3. Algorithme multi-échelles

En pratique, nous utilisons un noyau $G = (1/4, 1/2, 1/4)$ que nous convoluerons à l'image pour la filtrer. L'opérateur d sous-échantillonne l'image d'un facteur 2 dans chaque direction. Enfin, l'opérateur u sur-échantillonne en insérant des zéros dans les trous.

Selon les conclusions de la section 4.1.4.2, nous pouvons obtenir, sans trop craindre l'aliasage, à partir de notre image initiale I_t, une image filtrée :

$$J_t^0 = G \star I_t$$

de même taille, puis une suite de K images $(J_t^i)_{1 \leq i \leq K}$ de plus en plus petites, et telles que :

$$J_t^{i+1} = d(G \star J_t^i).$$

A l'inverse, il nous faut aussi opérer une transformation pour trouver, à partir des paramètres du champ obtenus à l'échelle i, les paramètres du champ à l'échelle $i-1$. Dans l'idéal, les nouveaux paramètres doivent donner un champ le plus proche possible

du champ fourni par les anciens paramètres. Dans notre implémentation, nous avons choisi comme fonction pour notre base nous permettant d'exprimer le champ :

$$\phi(\mathbf{x}) = \mathbf{1}_{[-1,1]}(|1 - x_1|)\mathbf{1}_{[-1,1]}(|1 - x_2|),$$

dilatée, et translatée de manière à ce que le support de ϕ_{ij} recouvre exactement les blocs voisins du nœud associé à cette fonction. Si ceux-ci sont de taille $n \times n$, nos fonctions de base sont alors :

$$\phi^n_{ij}(\mathbf{x}) = \phi(i - \frac{x_1}{n}, j - \frac{x_2}{n})$$

Ces fonctions correspondent à une interpolation bilinéaire à partir des nœuds de la grille. En prenant comme valeurs pour nos nouveaux paramètres les valeurs interpolées aux nouveaux points associés, nous retrouvons exactement le même champ. Les nouvelles valeurs se situant à mi-chemin des anciennes, il est possible d'obtenir la nouvelle grille de paramètres à partir de l'ancienne, en lui appliquant l'opérateur u, puis en convoluant avec ϕ^2_{00}.

Notre résolution multi-échelles se traduit par l'algorithme 2.

Algorithme 2 Obtention du champ de mouvement par résolution multi-échelles

$J^0_t \leftarrow G \star I_t)$
pour i de 1 à K **faire**
  $J^i_t \leftarrow \mathrm{d}(\mathrm{u}(G) \star J^{i-1}_t)$
fin pour
$v^{K+1} \leftarrow \mathbf{0}$
pour i de K à 0 **faire**
  $w^i \leftarrow \phi^2_{00} \star \mathrm{u}(v^{i+1})$
  $v^i \leftarrow$ résultat de l'algorithme (1) pour l'image J^i_t et J^i_{t+1}, initialisé avec w^i
fin pour
$w \leftarrow v^0$
$v_opt \leftarrow$ résultat de l'algorithme (1) pour l'image I_t et I_{t+1}, initialisé avec w

4.2. Résultats

Les tests ont été de deux espèces : nous avons tout d'abord comparé notre nouvelle méthode au block-matching qui constitue notre référence. Nous avons fait ensuite des tests en situation réelle, en intégrant notre méthode d'estimation au sein du codec MC-EZBC [10].

4.2.1. Comparaison avec le block-matching

4.2.1.1. Présentation des tests

Nous avons comparé notre méthode à un block-matching sans contrainte, c'est-à-dire pour lequel la minimisation n'implique aucune contrainte sur la forme des vecteurs pro-

duits autre que la pure minimisation de l'erreur de prédiction. Pour chacune des méthodes, nous avons regardé le PSNR entre l'image originale, et l'image prédite à partir du champ de mouvement estimé et de l'image précédente dans la séquence, dans son état original.

Notre méthode calcule des paramètres avec une précision quelconque. Cette caractéristique, bien qu'intéressante a priori, ne doit pas nous faire oublier que, dans une situation pratique, les paramètres de mouvement sont représentés par des entiers. Ils seront donc quantifiés. En revanche, le modèle interpole toujours les vecteurs de mouvement à partir des paramètres. Les vecteurs de mouvement utilisés auront donc une résolution différente des paramètres de mouvement. Le block-matching, quant à lui, fournit directement des paramètres de mouvement représentés par des entiers. Nous avons voulu comparer, à précision des paramètres égale, le PSNR au cours d'une séquence entre les deux méthodes. Le calcul a été effectué pour une précision des paramètres entière (1/1), demi-entière (1/2), au quart de pixel (1/4) et au huitième de pixel (1/8).

D'autre part, il est possible de choisir différentes tailles de blocs. Nous avons utilisé des blocs de 4 pixels par 4 (4×4), et des blocs de 8 pixels par 8 (8×8). Enfin, le nombre d'échelles sur lesquelles le calcul a été effectuée était de 4 ou de 6.

La légende pour chacune des figures suivantes est donnée par la table :

	Modèle bilinéaire par morceau	Block-Matching
4×4	A/rouge	C/vert
8×8	B/bleu	D/noir

4.2.1.2. Analyse

Notons tout d'abord que la comparaison présente en réalité un léger biais en faveur de notre nouvelle méthode, puisqu'à taille d'image équivalente, celle-ci doit fournir quelques paramètres supplémentaires pour représenter le champ.

Néanmoins, cela n'empêche pas le block-matching de donner dans quasiment toutes les séquences des résultats supérieurs à notre méthode lorsque la plus petite taille de blocs, 4×4 est utilisée. Pour cette petite taille de blocs, la modélisation du champ sur chaque bloc grâce à l'interpolation bilinéaire ne se distingue pas suffisamment du cas du champ constant par morceaux pour compenser ses deux désavantages : d'une part, la nécessaire continuité du champ de mouvement, contrairement au block-matching, d'autre part, le fait que le block-matching effectue une recherche exhaustive de la solution optimale, tandis que nous effectuons une descente de gradient qui, bien que sophistiquée, pourra s'arrêter dans des mimimas locaux.

En revanche, dans le cas des blocs 8×8, notre modèle de champ bénéficie véritablement de sa différence, même s'il reste pénalisé par son mode de résolution. Le modèle bilinéaire apporte une véritable amélioration pour la séquence mobile qui contient, parmi d'autres mouvements simples, des mouvements de rotations. Cependant, à cause de l'imposition de la continuité, il n'est pas compétitif par rapport au block-matching pour la séquence Football, dont le mouvement est rapide. Autre avantage de l'interpolation bilinéaire, notre

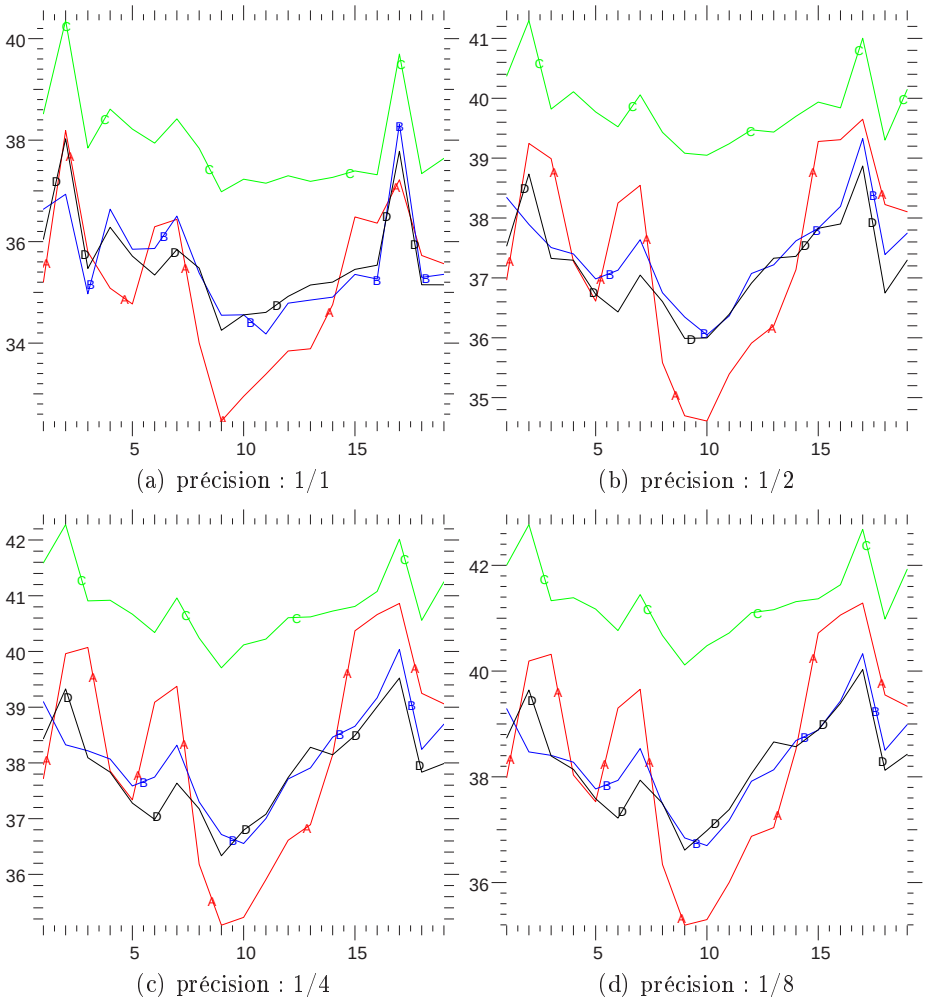
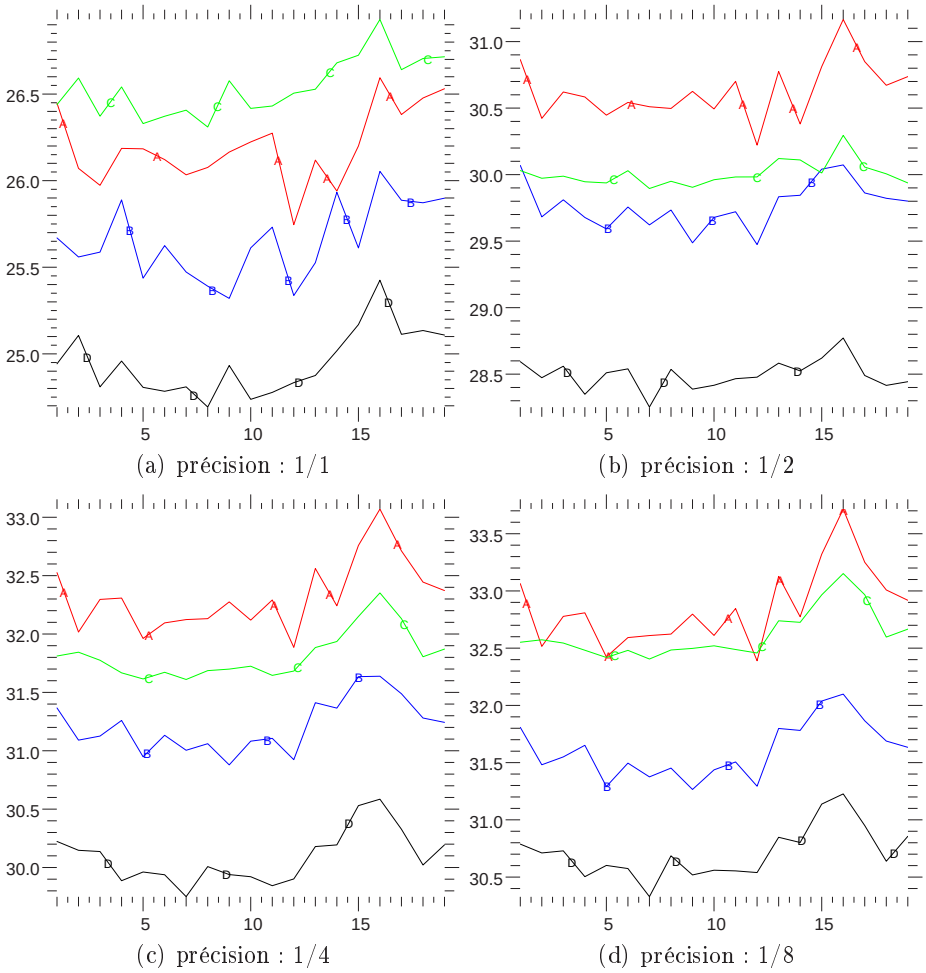


FIG. 4.4.: PSNR au cours du temps. *Foreman* CIF@30Hz 6 échelles. Courbe A/rouge : bilinéaire 4×4 ; courbe B/bleu : bilinéaire 8×8 ; courbe C/vert : BM 4×4 ; courbe D/noir : BM 8×8.



(a) précision : 1/1 (b) précision : 1/2

(c) précision : 1/4 (d) précision : 1/8

FIG. 4.5.: PSNR au cours du temps. *Mobile* CIF@30Hz 6 échelles. Courbe A/rouge : bilinéaire 4×4 ; courbe B/bleu : bilinéaire 8×8 ; courbe C/vert : BM 4×4 ; courbe D/noir : BM 8×8.

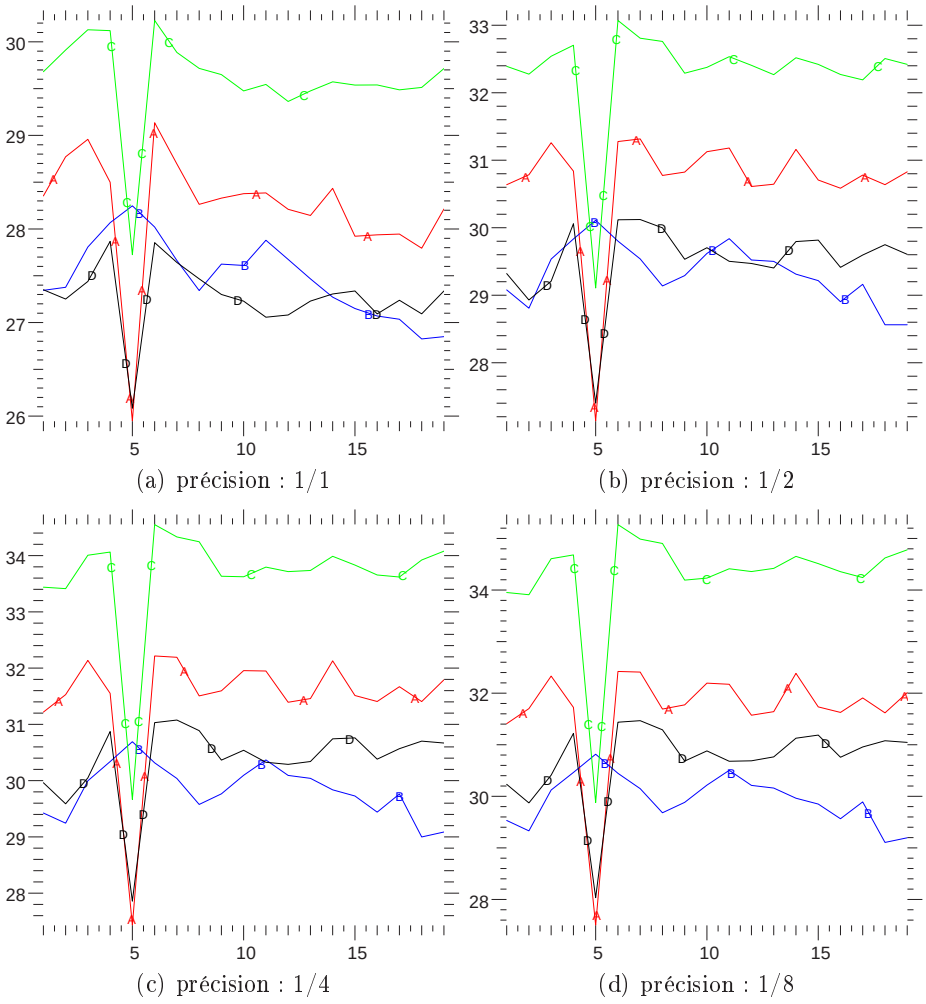
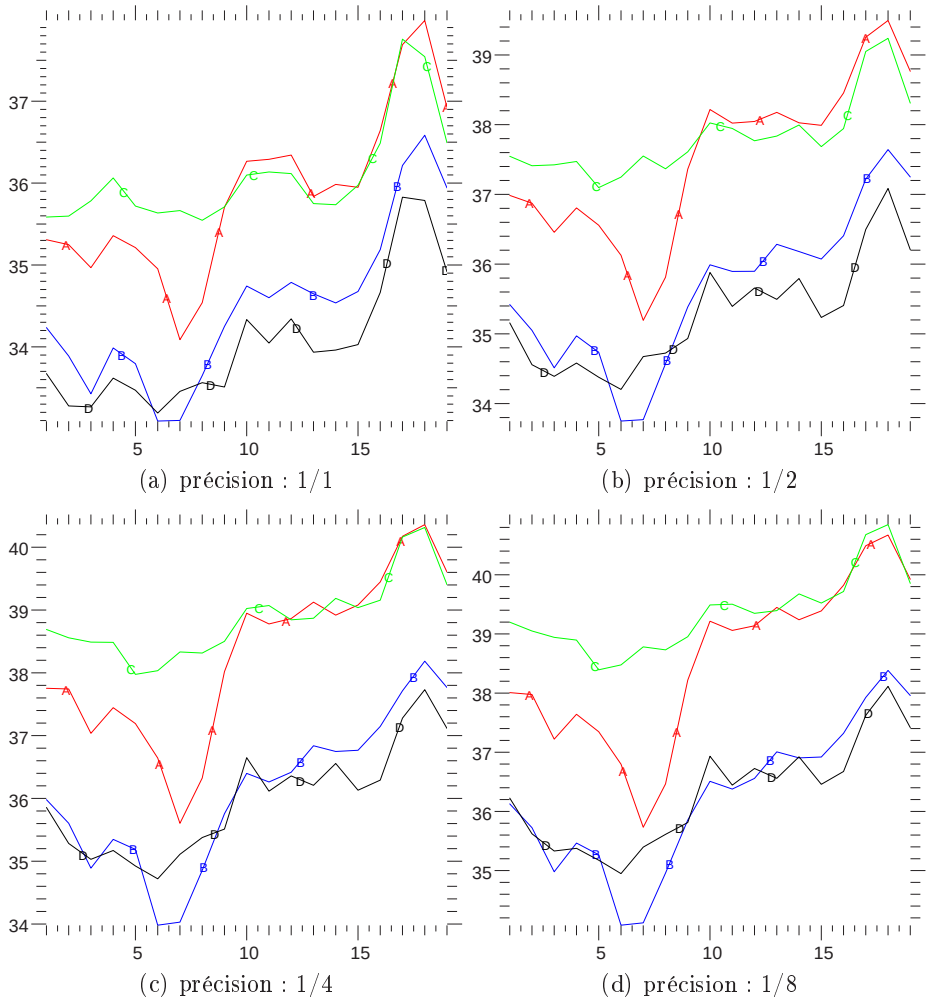


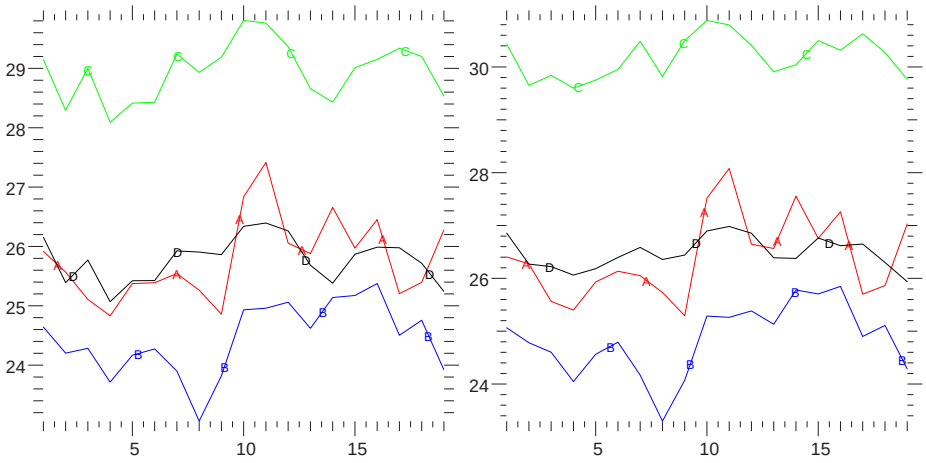
FIG. 4.6.: PSNR au cours du temps. *Bus* CIF@30Hz 6 échelles. Courbe A/rouge : bilinéaire 4×4 ; courbe B/bleu : bilinéaire 8×8 ; courbe C/vert : BM 4×4 ; courbe D/noir : BM 8×8.



(a) précision : 1/1 (b) précision : 1/2

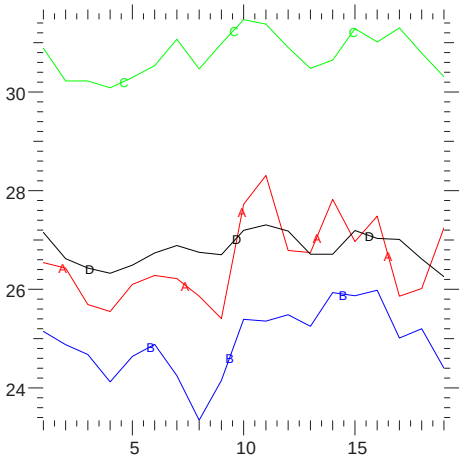
(c) précision : 1/4 (d) précision : 1/8

FIG. 4.7.: PSNR au cours du temps. *Paris* CIF@30hz 6 échelles (trame 201-220). Courbe A/rouge : bilinéaire 4×4 ; courbe B/bleu : bilinéaire 8×8 ; courbe C/vert : BM 4×4 ; courbe D/noir : BM 8×8.

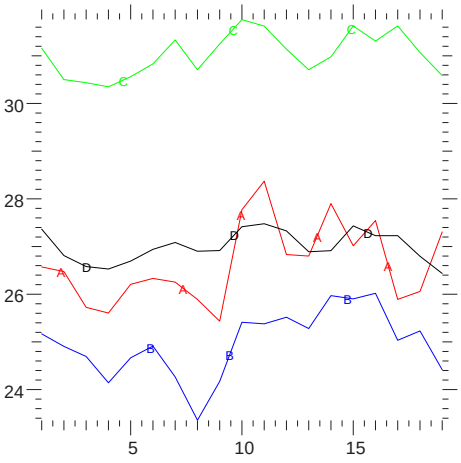


(a) précision : 1/1

(b) précision : 1/2

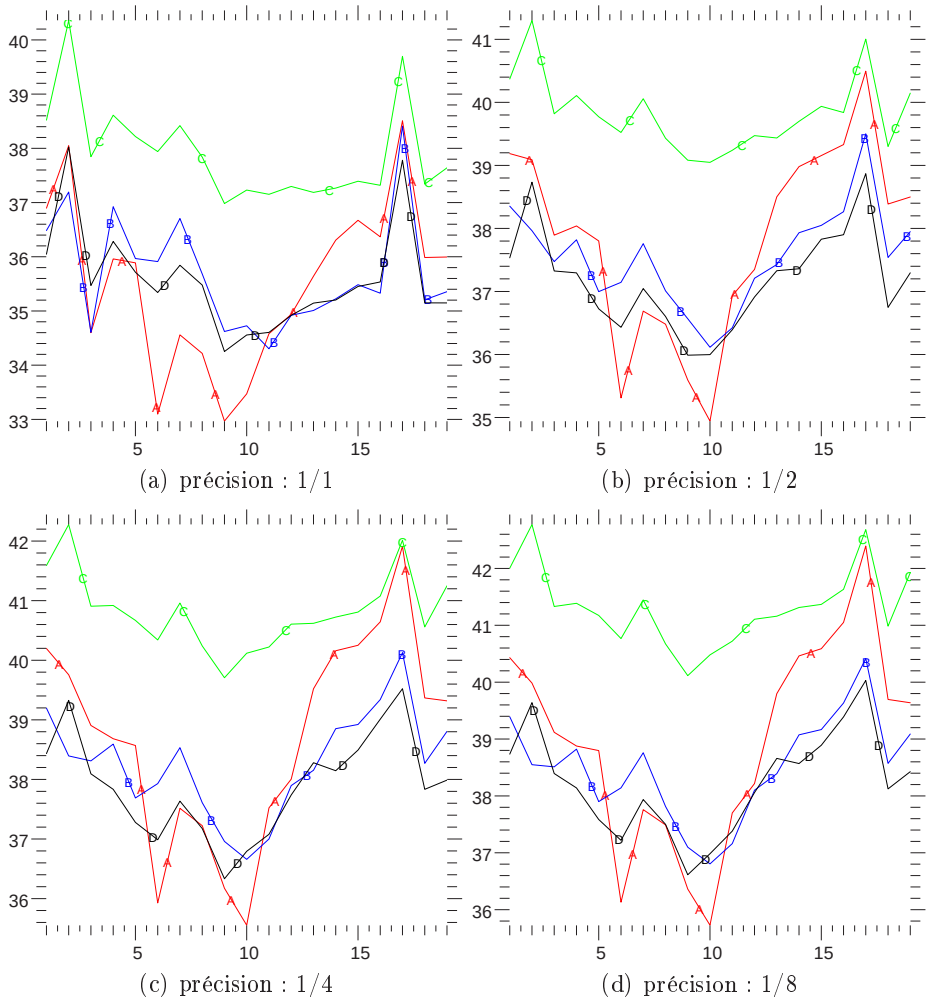


(c) précision : 1/4



(d) précision : 1/8

FIG. 4.8.: PSNR au cours du temps. *Football* SIF@30Hz 6 échelles. Courbe A/rouge : bilinéaire 4×4 ; courbe B/bleu : bilinéaire 8×8 ; courbe C/vert : BM 4×4 ; courbe D/noir : BM 8×8.



(a) précision : 1/1 (b) précision : 1/2

(c) précision : 1/4 (d) précision : 1/8

FIG. 4.9.: PSNR au cours du temps. *Foreman* CIF@30Hz, 4 échelles. Courbe A/rouge : bilinéaire 4×4 ; courbe B/bleu : bilinéaire 8×8 ; courbe C/vert : BM 4×4 ; courbe D/noir : BM 8×8.

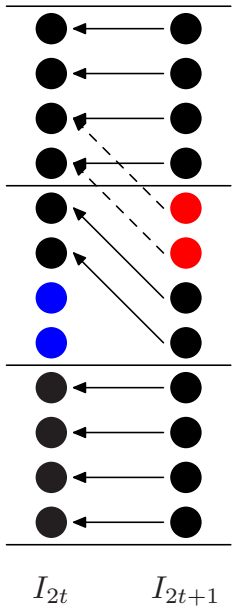


FIG. 4.10.: Repérage des différents types de pixels pour le filtrage temporel compensé en mouvement. En noir : pixel connecté. En bleu, pixel "non-référencé". En rouge, pixel non-connecté.

modèle a tendance à légèrement mieux supporter la quantification (voir par exemple la figure 4.6).

4.2.2. Résultats au sein du codeur MC-EZBC

La méthode d'estimation de mouvement a été intégrée au sein du codeur MC-EZBC de Woods et al. [10, 22, 9]. Avant d'analyser les résultats obtenus, donnons une rapide présentation de ce codeur.

4.2.2.1. Le codeur MC-EZBC

Filtrage temporel compensé en mouvement Le codeur MC-EZBC, libre d'accès à des fins expérimentales, fait partie de la nouvelle génération de codeurs qui s'appuient sur un filtrage temporel compensé en mouvement (« MCTF » pour Motion-Compensated Time Filtering). Ce type de filtrage est rendu possible grâce à l'utilisation du *lifting*.

Avant d'effectuer le filtrage, il nous faut classifier les pixels en plusieurs catégories (voir figure 4.10) :

– L'ensemble $\mathcal{D}_{nr}$ des pixels non-référencés est l'ensemble des pixels de l'image I_{2t} vers lesquels aucun vecteur de mouvement arrondi($\mathbf{v}$) ne pointe.

$$\mathbf{x} \in \mathcal{D}_{nr} \iff \forall \mathbf{y} \in I_{2t+1}, \mathbf{y} - \text{arrondi}(\mathbf{v}(\mathbf{y})) \neq \mathbf{x}$$

- Un point multi-connecté est un point de I_{2t} tel qu'il existe plusieurs vecteurs arrondi($\mathbf{v}$) pointant vers lui. On appelle antécédents les points associés.
- L'ensemble $\mathcal{D}_{nc}$ des pixels non-connectés est l'ensemble des pixels de l'image I_{2t+1} pour lesquels le vecteur de mouvement arrondi(v) pointe vers un point multi-connecté dont un des autres antécédents donne une différence de prédiction plus faible.

$$\mathbf{x} \in \mathcal{D}_{nc} \iff \exists \mathbf{y} \in I_{2t}, \exists \mathbf{x}' \in I_{2t+1}, \mathbf{y} = \mathbf{x} - \text{arrondi}(\mathbf{v}(\mathbf{x})) = \mathbf{x}' - \text{arrondi}(\mathbf{v}(\mathbf{x}')), \\ |I_{2t}(\mathbf{y}) - I_{2t+1}(\mathbf{x}')| < |I_{2t}(\mathbf{y}) - I_{2t+1}(\mathbf{x}')|. \quad (4.19)$$

- Les points qui ne sont ni dans $\mathcal{D}_{nr}$ ni dans $\mathcal{D}_{nc}$ sont dits connectés. Quand on a retiré les anciennes connections des points non-connectés, tous les points connectés deviennent simplement connectés.

Nous notons $\tilde{J}$ l'image interpolée de J à la position fournie. La première étape du *lifting* est la suivante :

$$H(\mathbf{x}) = \frac{1}{\sqrt{2}} \left(I_{2t+1}(\mathbf{x}) - \tilde{I}_{2t}(\mathbf{x} - \mathbf{v}(\mathbf{x})) \right)$$

Pour la seconde étape du *lifting*, séparons le cas des pixels connectés de celui des pixels non-référencés. Ces derniers sont traités de la façon suivante :

$$L(\mathbf{y}) = \sqrt{2} I_{2t}(\mathbf{y}).$$

Si un point $\mathbf{y}$ de I_{2t} est connecté, alors nous avons un point $\mathbf{x}$ de I_{2t+1} tel que $\mathbf{y} = \mathbf{x} - \mathbf{v}(\mathbf{x})$. Le point $\mathbf{y}$ est alors traité de la façon suivante :

$$L(\mathbf{y}) = \sqrt{2} I_{2t}(\mathbf{y}) + \tilde{H}(\mathbf{y} + \mathbf{v}(\mathbf{x}))$$

Ces deux étapes définissent ainsi une opération inversible qui effectue un filtrage de Haar dans la direction temporelle. À partir de deux images I_{2t} et I_{2t+1}, nous avons obtenu une image passe-bas L et une image passe-haut H. Il est possible de répéter le filtrage sur les images passe-bas successives afin d'obtenir plusieurs échelles de décomposition.

Il est possible de coder certains pixels d'une trame en codage intra, plutôt que d'utiliser une prédiction temporelle. Cette décision est faite par le codeur selon l'erreur de prédiction effectuée.

Décomposition spatiale et codage Une fois que le filtrage temporel est effectué, le codeur applique une décomposition temporelle en ondelettes 7/9.

Les coefficients sont ensuite encodés par le codeur EZBC qui est un codeur par plan de bits. L'information est placée dans un *quad-tree* où la valuation de chaque noeud est égal à l'amplitude maximale des coefficients dans la région associée au nœud, sur toutes les sous-bandes. La valuation de chaque nœud du *quad-tree* est encodée par un codeur arithmétique à contexte. Enfin, les vecteurs de mouvement, qui sont obtenus par un block-matching hiérarchique à taille de bloc variable (HVSBM), sont codés par une prédiction hiérarchique et un codage arithmétique.

4.2.2.2. Implémentation de l'estimation de champ bilinéaire par morceau au sein du codeur MC-EZBC

La principale difficulté d'implémentation provient du fait que le nombre de paramètres de mouvement à fournir n'est pas le même que pour les techniques de block-matching. Il y a en effet un nœud de plus dans chaque dimension qu'il n'y a de blocs. D'autre part, nous ne tirons pas pleinement parti de la taille variable qui peut être utilisée avec le block-matching hiérarchique, puisque nous devons fournir le champ de mouvement à tous les nœuds de la grille que nous nous sommes fixés. En revanche, nous bénéficions de la possibilité de coder une zone en intra (c'est-à-dire sans prédiction temporelle) au cas où l'estimation n'est pas bonne. En effet, notre méthode peut fournir des vecteurs de mouvement qui sortent de la trame de référence. Ici, nous pouvons donc, plutôt que tronquer le vecteur pour s'assurer qu'il prendra bien un point de la trame, demander simplement le codage intra.

4.2.2.3. Expériences

Nous avons encodé plusieurs séquences dans des blocs de 16 images avec le codeur MC-EZBC original, utilisant l'algorithme HVSBM, et le codeur MC-EZBC modifié, utilisant notre méthode d'estimation de mouvement (BBL). La précision des paramètres du champ de mouvement est fixée à $1/8^{\text{ème}}$ de pixel. Cependant, nous interpolons ces valeurs bilinéairement, et nous obtenons des fractions de dénominateurs supérieurs. Nous utilisons une taille de blocs de 8 pixels sur 8 pour la méthode BBL. Nous avons juste expérimenté la progressivité en qualité, sans utiliser la progressiviité spatiale et temporelle.

Les résultats obtenus mettent en valeur une perte de plusieurs décibels du fait de l'utilisation de l'algorithme BBL. Cette perte peut s'expliquer entre autres par l'adaptation du système de codage des vecteurs à l'algorithme hiérarchique, aspect dont nous ne pouvons pas profiter. D'autre part, il est à craindre que la possibilité de coder en intra ne soit pas suffisante pour réaliser un gain car ces pixels possèdent souvent une forte énergie.

4.3. Conclusion

Nous avons, dans ce chapitre, développé une méthode d'estimation de mouvement qui tente, contrairement à celle exposée au chapitre précédent, de minimiser explicitement le PSNR. La plupart des éléments de cette méthode sont des briques connues. Nous avons essayé d'apporter une attention particulière à leur combinaison, et en particulier, aux problèmes d'aliasage qui risquent d'apparaître dans les étapes de filtrage/sous-échantillonnage de la résolution multi-échelles. Néanmoins, faute de pouvoir effectuer une recherche exhaustive, notre méthode ne possède pas d'avantage évident en l'état sur le block-matching.

Plusieurs pistes existent afin d'améliorer nos résultats. Tout d'abord, il est possible qu'utiliser un découpage en triangles de l'image plutôt qu'en carrés se révèle plus efficace. Dans la mesure où la grande partie de ce que nous avons décrit est indépendant de la base utilisée, cette adaptation est aisée.

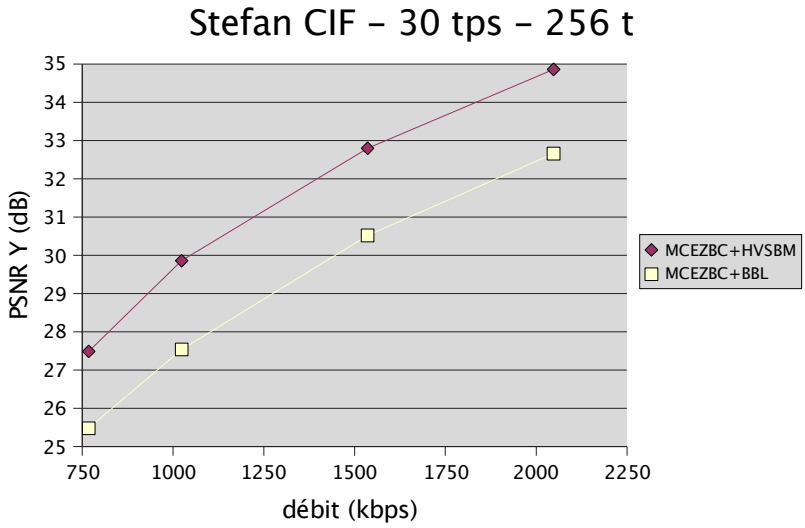


FIG. 4.11.: *Stefan*@30Hz. Comparaison entre les algorithmes BBL et HVSBM au sein du codeur de Woods.

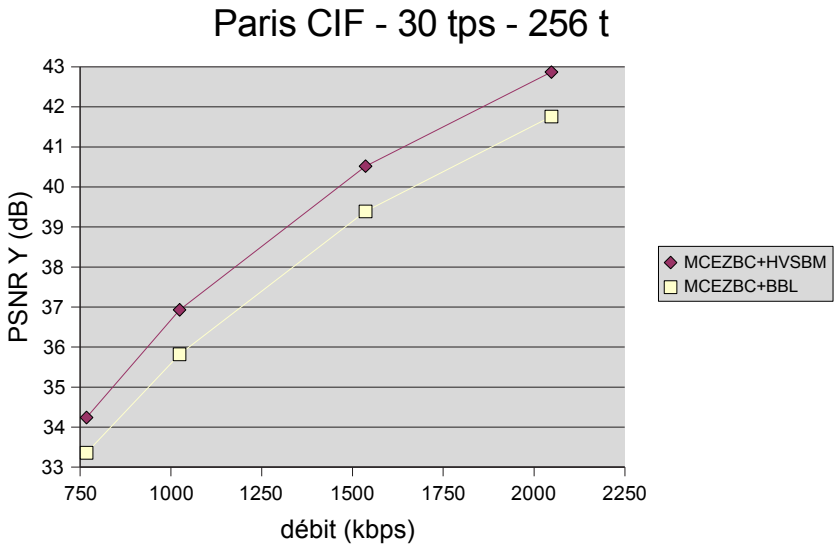


FIG. 4.12.: *Paris*@30Hz. Comparaison entre les algorithmes BBL et HVSBM au sein du codeur de Woods.

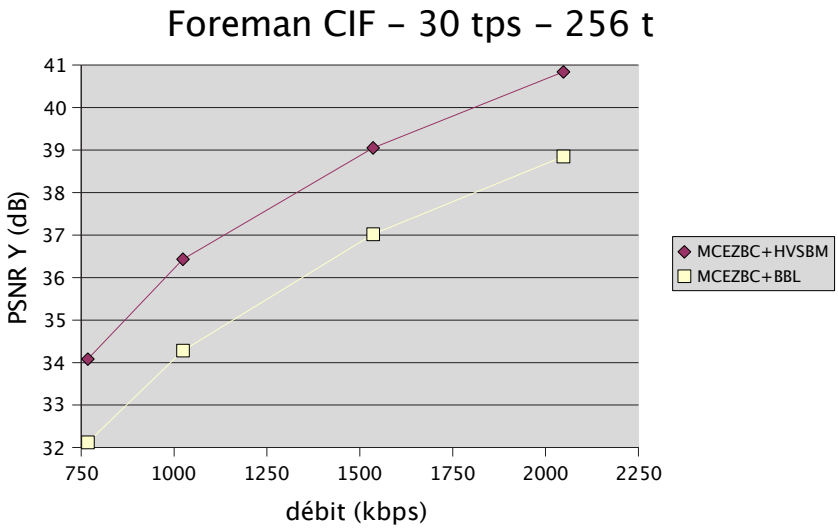


FIG. 4.13.: *Foreman*@30Hz. Comparaison entre les algorithmes BBL et HVSBM au sein du codeur de Woods.

D'autre part, un des défauts actuels de la méthode est que le modèle de mouvement sous-jacent est complètement continu, ce que nous savons faux en réalité. Il est possible d'envisager d'introduire des discontinuités dans notre modèle, comme le fait Ohm [35] avec un modèle similaire : les zones sur lesquelles nous effectuons l'interpolation peuvent être coupées en deux, et nous donnons alors un modèle distinct pour chaque morceau. La méthode de résolution est alors plus complexe, puisqu'il faut déterminer, outre les paramètres de mouvement sur les deux morceaux, la position de la cassure.

Troisième partie .

Compression

Chapitre 5.

Contrôle de débit et compression vidéo hybride

5.1. Le rôle du contrôle de débit

Dès que l'on dispose de données à transmettre ou à stocker, il est naturel de s'interroger sur la concision de la représentation de ces données. En effet, les supports de stockage (disque dur, DVD, etc.) sont de taille limitée, et il en est de même pour les canaux de transmission (fibre optique, signal hertizien, etc.). Il faut donc s'assurer que la représentation des données qu'on utilise respecte la contrainte matérielle posée. La contrainte peut s'exprimer selon les cas de façon globale ou locale. Dans le cas d'une séquence vidéo stockée sur un disque dur, le paramètre principal est la taille totale de la séquence sur le disque. En revanche, lorsqu'on effectue de la diffusion vidéo sur internet, la contrainte est donnée par le débit maximal du canal de transmission. Il y a un temps de latence initial acceptable entre l'émission et la réception des données, mais ensuite, l'utilisateur ne doit pas avoir à attendre l'arrivée des données pour suivre la vidéo : les données doivent donc pouvoir arriver au moins aussi vite que la vidéo se déroule. Nous avons donc une contrainte locale sur la taille des données à chaque unité de temps. Dans le cas du DVD, la contrainte est à la fois globale (la vidéo doit pouvoir tenir sur le support) et locale (le flux de données doit pouvoir être lu assez vite par le lecteur, et traité par le processeur). La transmission de données satellitaires fournit un autre exemple de contrainte locale. Un satellite d'observation acquiert constamment des données, qui doivent être transmises sur Terre. La capacité de mémoire du satellite étant limitée, il faut être capable de transmettre les données au fur et à mesure de l'arrivée de celles-ci – avec un léger temps de latence – grâce au signal hertzien.

Il existe deux grandes méthodes pour assurer le respect de la contrainte globale : la première est d'imposer avec une certaine souplesse une contrainte locale qui correspond en terme de débit à la contrainte globale. L'autre est de faire un premier passage sur l'ensemble des données afin d'étudier leur structure, puis d'effectuer la compression en utilisant la connaissance ainsi acquise. Cette deuxième méthode n'est pas toujours à disposition, par exemple dans le cas où il est nécessaire d'effectuer une compression à la volée des données, au fur et à mesure de leur obtention.

Dans le cas de la vidéo, la taille des données est généralement trop importante pour que l'on puisse espérer transmettre ou stocker celles-ci sans avoir à les dégrader. Nous avons donc à résoudre deux problèmes joints : faire respecter à la représentation de nos

données la contrainte globale ou locale qui nous est imposée, d'une part, et d'autre part, limiter au mieux la distorsion qu'introduit notre nouveau mode de représentation.

Les formats de stockage progressifs sont vraisemblablement une des réponses les plus souples à ce problème, puisque chaque bit du fichier résultant est censé être optimal en terme de diminution marginale de la distorsion. Cependant, la nature même de ces schémas peut déboucher sur une légère inefficacité, et c'est pour cette raison que les algorithmes qui visent à créer une version ciblant des valeurs précises du critère sont encore intéressants.

En pratique, les questions auxquelles se doit de répondre un algorithme d'allocation de débit en compression vidéo sont les suivantes :

- comment répartir le débit entre chacune des images ou comment assurer le respect de la contrainte de débit au long des images ?
- comment répartir le débit au sein d'une image ou comment assurer le respect de la contrainte de débit au sein de l'image ?

Un algorithme de débit qui fait référence a été proposé pour le standard H263, et intégré au codeur/décodeur TMN.8 de ce standard. Plus récemment, He et al. [18, 19, 17] ont proposé un algorithme d'allocation de débit s'appuyant sur un modèle empirique des données, algorithme qui donnerait des meilleurs perfomances que celui du TMN.8. Mallat et Falzon ont développé un algorithme d'allocation de débit pour la compression d'images fixes. L'application qui a été donnée à cet algorithme est celle de la transmission d'images satellitaires, qui ne forment en fait qu'une image de hauteur infinie. On retrouve ici la contrainte locale qui est correspond généralement plutôt à la vidéo. L'objet de ce chapitre est de formaliser une adaptation de ce dernier algorithme à la vidéo, puis d'effectuer une comparaison entre les trois méthodes.

5.2. Compression par codage avec quantification

5.2.1. Compression et taux de distortion

Nous rappelons ici dans cette section des bases théoriques sur lesquelles s'appuient les algorithmes d'allocations de débit présentés (TMN.8 et Mallat-Falzon).

Dans un cadre très général, nous pouvons considérer qu'un signal est la réalisation d'un processus aléatoire $F[k]$ de taille N. Le codage par transformée va exprimer ce signal sur une base orthonormée $\mathcal{B} = \{g_k\}_{0 \leq k < N}$:

$$F = \sum_{k=0}^{N-1} F_{\mathcal{B}}[k] g_k.$$

Chaque coefficient $F_{\mathcal{B}}[k]$ peut être vu comme une variable aléatoire définie par :

$$F_{\mathcal{B}}[k] = \langle F, g_k \rangle = \sum_{l=0}^{N-1} F[l] g_l^*[l],$$

où $\mathcal{B}^* = \{g_k^*\}_{0 \leq k < N}$ est la base duale de $\mathcal{B}$. Quitte à noter les valeurs de $E(F_{\mathcal{B}}[k]$ et à l'enlever de $F_{\mathcal{B}}[k]$, on suppose que $F_{\mathcal{B}}[k]$ est à moyenne nulle.

Les valeurs prises par les coefficients sont généralement réelles. La précision du support de calcul limite naturellement la taille du code nécessaire pour représenter ces valeurs, mais il faut quantifier pour réduire véritablement la taille moyenne du code représentant chacun des coefficients. La quantification transforme chaque coefficient $F_{\mathcal{B}}[k]$ par un variable quantifiée $\bar{F}_{\mathcal{B}}[k]$, chaque coefficient étant traîté indépendamment (nous nous plaçons ici dans le cas d'une quantification scalaire). Après quantification, le signal reconstruit est :

$$\bar{F} = \sum_{k=0}^{N-1} \bar{F}_{\mathcal{B}}[k] g_k.$$

La distorsion que provoque la quantification sur un jeu de coefficients peut être mesurée par l'erreur quadratique moyenne :

$$d = E(\|F - \bar{F}\|) = \sum_{k=0}^{N-1} E(|F_{\mathcal{B}}[k] - \bar{F}_{\mathcal{B}}[k]|^2).$$

En moyenne, le PSNR de l'image représenté par les coefficients vaut donc :

$$PSNR = -10 \log_1 0 \frac{d}{NV_{max}^2},$$

où V_{max} est la valeur maximale de l'image. Le calcul sur les coefficients transformés est possible puisque nous supposons que nous exprimons les coefficients sur une base orthonormée, comme la base canonique. Si nous notons R_k le nombre de bits utilisés pour coder $\bar{F}_{\mathcal{B}}[k]$, la distorsion dépend du nombre total de bits :

$$R = \sum_{k=0}^{N-1} R_k.$$

$d(R)$ est appelé le taux de distorsion. Tout l'art de la compression avec quantification se trouve dans la réalisation d'une minimisation du taux de distorsion à débit constant, c'est à dire avec un nombre fixe de bits utilisés pour représenter les coefficients.

5.2.2. Quantification haute-résolution

Étudions ce qui se passe pour un des $F_{\mathcal{B}}[k]$ que l'on notera variable X. Un quantificateur scalaire Q va découper la droite des réels en intervalles de la forme $]y_k, y_{k+1}]$, nommés boîtes de quantifications, et approchera tout $x \in]y_k, y_{k+1}]$ par x_k :

$$\forall x \in]y_k, y_{k+1}],\ Q(x) = x_k.$$

Soit $p(x)$ la densité de probabilité de X. L'erreur de quantification est alors :

$$d = E((X - \bar{X})^2) = \int_{-\infty}^{+\infty} (x - Q(x))\, p(x) dx.$$

L'hypothèse de haute-résolution assume que sur chaque boîte de quantification $]y_k, y_{k+1}]$, $p(x)$ est approximativement constant, égal à p_k. Si nous notons Δ_k les tailles des boîtes de quantifications :

$$\Delta_k = y_{k+1} - y_k,$$

alors

$$\forall x \in]y_k, y_{k+1}], \; p(x) = \frac{p_k}{\Delta_k}$$

Cette hypothèse est vérifiée dans le cas où les boîtes de quantification sont suffisamment petites par rapport au taux de variation de $p(x)$, si bien que nous pouvons alors négliger ses variations au sein d'une boîte. Dans ce cas, le taux de distortion est minimisé si x_k est le milieu de l'intervalle $[y_k, y_{k+1}]$. Nous aurons alors :

$$d = \frac{1}{12} \sum p_k \Delta_k$$

On appelle quantificateur uniforme le quantificateur où toutes les boîtes de quantification sont de la même taille :

$$\forall k \in \mathbf{Z}, \; y_{k+1} - y_k = \Delta.$$

Le taux de distortion devient alors :

$$d = \frac{\Delta}{12}.$$

L'entropie différentielle de X est définie par :

$$\mathcal{H}(X) = -\int_{-\infty}^{+\infty} p(x) \log_2 p(x) dx.$$

Dans le cadre de l'hypothèse de quantification haute-résolution, nous relions l'entropie différentielle de X et l'entropie de $\bar{X}$ en montrant que

$$\mathcal{H}(\bar{X}) \geq \mathcal{H}_d(X) - \frac{1}{2} \log_2(12d)$$

où l'inégalité est une égalité si et seulement si Q est un quantificateur uniforme. L'entropie de $\bar{X}$ étant aussi R_X le nombre de bits moyen minimal nécessaire pour coder les réalisations de $\bar{X}$, on en déduit alors que

$$R_X = \mathcal{H}(\bar{H}) = \mathcal{H}_d(X) - \log_2(12d).$$

En inversant cette équation, nous obtenons la relation :

$$d(R_X) = \frac{1}{12} 2^{2\mathcal{H}_d(X)} 2^{-2R_X}.$$

Revenons aux $F_{\mathcal{B}}[k]$. Il est possible de montrer que pour un taux de distortion fixé, la quantification optimale quantifie chacun des coefficients identiquement. On a alors :

$$\Delta_k = \frac{12d}{N} \qquad \text{pour } 0 \leq k < N$$
$$d(\bar{R}) = \frac{N}{12} 2^{2\bar{\mathcal{H}}_d} 2^{-2\bar{R}},$$

où $\bar{\mathcal{H}}_d$ est la moyenne des entropies différentielles de chacun des coefficients.

Nous avons donc dans ce cas-là un lien clairement exprimable entre le pas de quantification à utiliser et le débit que nous cherchons à atteindre. Cependant, l'étude jusqu'ici a posé plusieurs contraintes qui ne seront pas toujours valables en pratique :

- L'indépendance des coefficients n'est pas établie, et les schémas de codage sont conçus pour tirer partie de cette corrélation.
- L'hypothèse de quantification haute-résolution impose a priori que nous travaillions dans un régime de faible distorsion. En réalité, nous sommes souvent plus intéressés par le cas où le débit est faible, et donc où la distorsion sera élevée. Dans ces domaines, l'hypothèse ne se vérifie pas complètement.

Néanmoins, les algorithmes existants s'appuient sur des considérations similaires à celles que nous venons de faire.

5.3. Des algorithmes de contrôle de débit pour un standard de compression bien connu

Il existe un grand nombre d'algorithmes de contrôle de débit pour les standards existants. Nous présentons ici celui qui fait référence pour H.263, ainsi qu'un autre algorithme, développé par He et Kim [18], dont les résultats ont été salués. Ensuite, nous donnons les explications sur le modèle de contrôle de débit pour la compression d'images fixes de Mallat et Falzon [29], qui présente des similarités avec celui de He et Kim, mais qui s'appuie sur des considérations plus théoriques.

5.3.1. TMN.8 (H.263)

L'algorithme de contrôle de débit le plus couramment utilisé pour les tests est celui intégré au sein du codeur expérimental TMN.8 pour H.263 [39]. Nous l'appellerons dans la suite "algorithme du TMN.8", indépendamment du fait que ce codeur expérimental présentait d'autres avancées par rapport aux codeurs précédents. Cet algorithme agit à deux échelles distinctes : celui de l'allocation de débit pour chaque trame, et celui de l'allocation de débit au sein d'une trame.

5.3.1.1. Contrôle à l'échelle de la trame

La première étape de l'algorithme du TMN.8 est de déterminer le débit disponible pour l'image à coder.

Mémoire-tampon On suppose l'existence d'une petite mémoire de stockage, la mémoire-tampon, ou tampon, qui nous permet de ne pas réaliser le débit cible pour chaque image, mais de s'en écarter légèrement de temps à autre. Comme il est difficile de savoir à l'avance quel sera le type de matériel sur lequel fonctionnera le décodeur ou le codeur, l'encodeur expérimental simule cette mémoire-tampon. La taille théorique de celui-ci est généralement égale à l'espace moyen pris par une trame au débit cible. Cela permet de ne pas avoir trop de fluctuations de fréquence au cours de la séquence. Il est possible que l'algorithme d'allocation de débit agisse de façon incorrecte. Dans ce cas, il est possible que le tampon "déborde" à un instant donné. Pour vider le tampon et pouvoir rependre le cours de la vidéo avec le temps de latence désiré, il faut abandonner (sauter) certaines trames, autant qu'il est besoin pour éliminer le débordement du tampon.

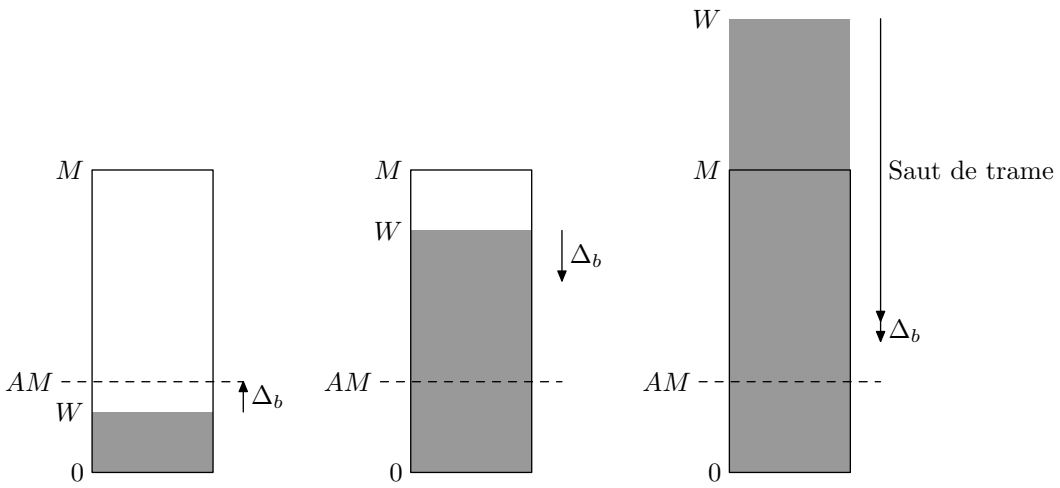
Notations Notons :

- D le débit cible en bits par seconde
- H le nombre de bits utilisés lors de l'encodage de la trame précédente
- R le budget de bits pour l'encodage de la prochaine trame
- G le nombre d'image par seconde dans la séquence originale
- F le nombre d'image par seconde que l'on désire avoir dans la séquence une fois codée
- M la taille virtuelle du tampon en bits, qui sera le sueil pour le saut d'une image lors de l'encodage.
- A la fraction d'équilibre du tampon. L'objectif est que le tampon soit rempli en moyenne de AM bits. Il est possible de voir AM comme le retard de visualisation dû à l'utilisation du tampon
- W le nombre de bits actuellement dans le tampon
- Δ_d la différence entre le budget moyen R/F et le budget réel pour la prochaine trame B

Dans une situation idéale, nous avons :

$$\begin{aligned} R &= D/F \\ \Delta_d &= 0 \\ H &= R_{prec} \\ W &= AM \end{aligned}$$

Algorithme Les images sont sautées si le nombre de bits accumulés dans le tampon par le codage de l'image précédente est trop important. Plus précisément, si jamais le remplissage du tampon est supérieur à la taille de celui-ci (i.e. $W > M$), nous sautons une trame et vidons le tampon de l'objectif de taille d'une trame (D/F), et ce, jusqu'à avoir éliminé le sur-remplissage du tampon.



(a) Niveau du tampon au-dessous du niveau cible

(b) Niveau du tampon au-dessus du niveau cible

(c) Débordement du tampon

FIG. 5.1.: Schéma de la mémoire-tampon. Sa taille est M, et son niveau cible de remplissage est AM. Si $W < AM$, nous bénéficions d'un surplus de budget pour revenir à niveau cible. Si $W > AM$, le budget est limité, avec un amortissement du contrecoup du dépassement sur quelques images. Enfin, s'il y a débordement du tampon, on saute des trames jusqu'à ce qu'enlever le budget de ces trames ait éliminé le débordement du tampon

Le budget pour l'image est alors déterminé de façon à se rapprocher du niveau cible du tampon. Si le tampon est moins rempli que la cible ($W < AM$), la différence entre les deux niveaux ($AM - W$) se reporte intégralement sur le nouveau budget. En revanche, au cas où le remplissage du tampon dépasse le niveau cible ($W > AM$), le manque n'est pas intégralement perdu, mais sa compensation, W/F, est répartie sur un certain nombre de trames suivantes, afin d'eviter une variation trop brutale du budget et donc de la qualité des images codées[1].

L'algorithme 3 récapitule avec les formules correspondantes le mécanisme mis en oeuvre. La sous-boucle prépare le saut d'image pour permettre au tampon de se vider de son dépassement, comme nous l'avons mentionné précédemment.

Algorithme 3 Algorithme du TMN.8 pour l'allocation de débit inter-trame

répéter
  $W \leftarrow \max(W + H - D/F, 0)$
  $skip \leftarrow 1$
  tant que $W > M$ **faire**
    $W \leftarrow \max(W - D/F, 0)$
    $skip \leftarrow skip + 1$
  fin tant que
  l'encodage des $skip.G/F - 1$ trames suivantes est sauté.
  si $W > AM$ **alors**
    $\Delta_d \leftarrow W/F$
  sinon
    $\Delta_d \leftarrow W - AM$
  fin si
  $B \leftarrow D/F - \Delta_d$
  encodage de la trame courante avec un budget de bits R
jusqu'à il ne reste plus de trames à encoder

5.3.1.2. Contrôle de débit à l'intérieur d'une trame

Dans le standard H.263, l'image est découpée en macroblocs (carrés de 16 pixels de côté) qui sont codés les uns après les autres. Il est possible d'indiquer au début du codage de chaque macrobloc si l'on désire faire évoluer le paramètre QP du pas de quantification, au sein d'un petit intervalle autour du paramètre du pas de quantification précédent QP_{prec}. Le paramètre du pas de quantification est un entier qui vaut la moitié du pas de quantification effectivement utilisé.

Une fois déterminé le nombre de bits disponible pour le codage de l'image, on utilise un système de contrôle au niveau des macroblocs afin de choisir les valeurs optimales pour le paramètre de quantification au sein de chacun d'entre eux.

[1] W/F n'est pas homogène à un nombre de bits, mais il y a une multiplication implicite par une durée d'une seconde qui correspond au temps visé d'amortissement du dépassement par rapport au niveau cible.

L'algorithme détermine dans l'image les valeurs de pas de quantification à utiliser macrobloc par macrobloc, dans l'ordre de codage des blocs (de gauche à droite et de haut en bas).

Nous notons ici :

- N_{pix} le nombre de pixels au sein d'un macrobloc (256 pour un macrobloc de 16 pixels par 16).
- R_i le nombre de bits utilisés par le codage du i-ème macrobloc
- σ_i l'écart-type du i-ème macrobloc de la trame d'erreur.
- Δ_i est le pas de quantification pour le i-ème macrobloc.
- N_{mb} est le nombre de macroblocs dans l'image. Le nombre total de coefficients est donc $N = N_{pix}N_{mb}$.
- C le "débit d'en-tête par pixel". Ceci correspond à la répartition du coût des vecteurs de mouvement et des bits de syntaxe sur chacun des pixels.
- K un paramètre du modèle à déterminer selon la distribution des coefficients dans la trame

L'algorithme du TMN.8 s'appuie sur le modèle de débit suivant :

$$R_i = N_{pix}(K\frac{\sigma_i^2}{\Delta_i^2} + C).$$

Ce modèle dérive de l'approximation de l'entropie de $\bar{X}_i$, variable aléatoire issu de la quantification avec un pas Δ_i de la variable aléatoire X de distribution laplacienne. Cette approximation, dans le cas où $\frac{\sigma^2}{\Delta_i^2} < \frac{1}{2e}$ est donnée par [40] :

$$\mathcal{H}(\bar{X}_i) = \frac{e}{ln2}\frac{\sigma_i^2}{\Delta_i^2}. \tag{5.1}$$

La nécessité de prendre K différent de $\frac{e}{ln2}$ vient du fait que les coefficients ne possède pas exactement une distribution laplacienne d'une part, et d'autre part de la capacité des codeurs à exploiter la corrélation spatiale entre coefficients.

Les valeurs de K et de C seront déterminées de façon empirique au fur et à mesure du codage des macroblocs, puisqu'il est possible de les calculer pour les macroblocs déjà codés. La valeur utilisée pour le i-ème macrobloc sera la moyenne constatée sur les $i-1$ premiers macroblocs.

Dans la mesure où changer le pas de quantification pour un nouveau macrobloc à un coût, faible mais pas nécessairement négligeable, le modèle attribue à la distorsion potentielle de chaque macrobloc un poids α_i défini par :

$$\alpha_i = \begin{cases} 2\frac{R}{N}(1-\sigma_i) + \sigma_i & \text{si } R < 0.5N \\ 1 & \text{sinon} \end{cases}$$

Notons maintenant :

- r_i le budget restant pour coder les macroblocs de i à N_{mb}.
- n_i le nombre de macroblocs restant à coder, y compris le i-ème :

$$n_i = N_{mb} + 1 - i.$$

- L_i le nombre de bits potentiellement restant pour coder les coefficients :

$$L_i = r_i - N_{pix} n_i C.$$

- S_i la somme pondérée des variances des macroblocs d'erreur restants :

$$S_i = \sum_{j=i}^{N_{mb}} \alpha_i \sigma_i.$$

Les pas de quantification pour chacun des macroblocs sont alors calculés successivement de la façon suivante :

- Les valeurs des σ_i sont calculées pour chacun des macroblocs.
- Si $L_i > 0$, c'est-à-dire s'il reste un budget de bits pour les coefficients, alors on prend :

$$\Delta_i^* = \sqrt{\frac{N_{pix} K_i \sigma_i}{L_i \alpha_i} S_i},$$

  et l'on choisit le paramètre du pas de quantification dans l'intervalle de variation possible qui donne le pas de quantification le plus proche de Δ_i^*. Sinon, on ne fait qu'augmenter le pas de quantification.
- Comme indiqué précédemment, on met à jour les paramètres du modèle C et K à partir du résultat effectif du codage, avant de passer au macrobloc suivant.

Ribas et Lei donnent dans [39] une description complète et détaillée de l'algorithme.

5.3.2. He et Kim (H.263)

He et Kim ont développé dans [18] un algorithme de contrôle du débit agissant à l'intérieur du codage d'une image (et qui se substitue donc seulement à la deuxième partie de l'agorithme décrit en 5.3.1).

5.3.2.1. Aspect théorique

Le modèle de He et Kim tire parti du lien empirique qui existe entre le débit, et la proportion ρ de coefficients quantifiés à zéro. Il décompose le débit R entre la partie R_z utilisée pour coder les coefficients nuls et celle R_{nz} utilisée pour les coefficients non-nuls.

D'autre part, leur méthode s'appuie sur une étude du pré-encodage des coefficients à l'aide de *run-length* : dans un macrobloc, les coefficients sont parcourus dans un ordre déterminé. Plutôt que de coder chaque valeur séparément, la technique de pré-encodage

suppose que la proportion de coefficients quantifiés à zéro est élevé, et transforme la suite des coefficients quantifiés en une suite de paires de nombres $(a[k], r[k])$:

$$(\bar{F}_{\mathcal{B}}[k])_{1\leq k\leq N_{pix}} \mapsto ((a[k], r[k]))_{1\leq k\leq m}$$

Le premier élément de la paire, $a[k]$ donne l'amplitude du k-ième coefficient non-nul, et le second, $r[k]$, représente le nombre de coefficients nuls qui suivent ce coefficients non-nul. L'idée est que m est petit devant N_{pix} et qu'ainsi la représentation des données à coder est plus compacte.

Nous avons alors :

$$\rho = \frac{1}{N}\sum_{i=0}^{N_{mb}}\sum_{k=0}^{m_i} r[k],$$

le nombre de coefficients signicatifs étant lui égal à :

$$\nu = \frac{1}{N}\sum_{i=0}^{N_{mb}} m_i$$

He et Kim définissent Q_{nz} comme la somme du nombre de bits de la magnitude de chaque coefficient significatif dans l'image, rapporté au nombre de pixels dans l'image.

$$Q_{nz} = \frac{1}{N}\sum_{i=0}^{N_{mb}}\sum_{k=0}^{m_i}\lceil \log_2 |a[k]| \rceil + 1.$$

Q_z est défini comme la somme du nombre de bits des *run-length* entre coefficients significatifs, rapportée au nombre de pixels dans l'image :

$$Q_z = \frac{1}{nN}\sum_{i=0}^{N_{mb}}\sum_{k=0}^{m_i}\lceil \log_2 |r[k]| \rceil + 1.$$

Q_{nz} et Q_z sont des fonctions de la valeur du pas de quantification Δ. Or celui-ci est directement lié, connaissant l'ensemble des valeurs des coefficients, au nombre de coefficients significatifs ou inversement à la proportion de coefficients quatnfiés à zéro, c'est-à-dire à ρ.

Au terme de considérations empiriques, les auteurs tirent les conclusions suivantes :

- La relation entre Q_{nz} et ρ est quasi-linéaire, sous la forme

  $$Q_{nz} = \vartheta(1-\rho). \tag{5.2a}$$

  ϑ est un paramètre du modèle dépendant de l'image à coder.
- Une fois ϑ obtenu, il existe une corrélation entre Q_z, ϑ et ρ. Celle-ci peut s'écrire approximativement sous la forme

  $$Q_z = P_\rho(\vartheta) \tag{5.2b}$$

  où P_ρ est un polynôme du troisième degré dont les coefficients sont des fonctions de ρ, indépendantes de l'image, et donc déterminées une fois pour toutes.

– He et Kim constatent ensuite que le débit peut être approxiché par une fonction de la forme :

$$R(\rho) = \xi_1(\rho)Q_{nz}(\rho) + \xi_2(\rho)Q_z(\rho) + \xi_3(\rho). \tag{5.2c}$$

ξ_1, ξ_2 et ξ_3 étant des fonctions indépendantes de l'image.

5.3.2.2. Aspect pratique

En réalité, P_ρ, ξ_1, ξ_2 et ξ_3 ne sont déterminés que pour un nombre limité de valeurs de ρ. Néanmoins, il est possible de calculer, à l'aide des équations (5.2), les valeurs de R pour ce jeu de valeurs de ρ. Les autres valeurs de R sont interpolées linéairement à partir de ces valeurs.

R doit logiquement décroître quand ρ augmente : il est donc possible de trouver le ρ adéquat pour obtenir un débit donné. D'autre part, en utilisant l'histogramme des coefficients, il est aussi aisé de trouver la taille T de la boîte de quantification zéro associée à une valeur de ρ donnée et donc d'obtenir la valeur Δ^*, le rapport $\theta = T/\Delta$ étant fixe. La relation $\Delta^* \mapsto \rho$ est strictement croissante - plus on quantifie durement, plus le nombre de coefficients nuls est élevé -, ce qui permet d'obtenir la mise en correspondance.

5.3.2.3. Algorithme

L'algorithme est alors le suivant :

– Nous calculons les distributions des coefficients, inter-codés et intra-codés.

– Nous choisissons un pas de quantification fournissant une valeur de ρ telle que

$$0.9 < \rho < 0.95.$$

Pour cette valeur du pas de quantification, nous calculons Q_{nz}. L'équation (5.2a) nous donne alors la valeur de ϑ.

– À partir de la valeur de ϑ et de l'équation (5.2b), nous calculons la valeur de Q_z pour un jeu fixe de valeurs de ρ, puis les valeurs de R.

– Avec l'interpolation linéaire, les valeurs calculées nous donnent la courbe $\rho \mapsto R$. Celle-ci étant censée être décroissante, nous pouvons l'inverser pour obtenir la courbe $R \mapsto \rho$.

– Nous déterminons alors ρ, puis la valeur du paramètre de quantification QP_0 correspondant.

En pratique, il n'est pas toujours possible de trouver une valeur entière de QP réalisant ρ. Pour résoudre ce problème, les auteurs mettent en place un mécanisme de contrôle visant à réaliser en moyenne l'objectif sur l'image entière, en faisant varier légèrement le paramètre de quantification au sein de l'image. Notons ρ^+ la porportion de zéro obtenu pour les macroblocs déjà codés (ce qui est une valeur exacte), et ρ^- la proportion de zéro

à obtenir pour les macroblocs restant à coder (déduit à partir du modèle appliqué sur l'image entière). Posons alors :

$$\kappa = \frac{\rho^-}{\rho^+},$$

Selon la valeur de ce rapport, le pas de quantification est choisi au sein des valeurs entre $QP_0 - 3$ et $QP_0 + 3$. Dans le cas où κ est inférieur à 1, ce qui signifie que l'on a obtenu trop de zéros par rapport à l'objectif, le pas de quantification est diminué, tandis qu'il est augmenté si κ est supérieur à 1, ce qui signifie que nous n'avons pas obtenu suffisamment de zéros par rapport à l'objectif, et donc que nous n'avons pas quantifié assez durement jusqu'à présent.

5.3.3. Algorithme de Mallat et Falzon

5.3.3.1. Aspects théoriques

Les travaux de Mallat et Falzon [29] s'appuient sur une analyse similaire à celle de He et Kim. le débit estimé pour le codage d'une image est décomposé entre la partie R_z utilisée pour coder les coefficients nuls et celle R_{nz} utilisée pour les coefficients non-nuls, résultant de la transformation effectuée préalablement sur l'image.

Lien entre ρ et R_z Nous pouvons imaginer un schéma simple pour coder la carte des coefficients non-nuls, c'est-à-dire pour coder les coefficients nuls. Nous nous fixons un parcours, et si $F_{\mathcal{B}}[k]$ est quantifié à zéro, nous codons $s[k]$ comme 0, et 1 sinon :

$$s[k] = \begin{cases} 0 & \text{si} \bar{F}_{\mathcal{B}}[k] = 0 \\ 1 & \text{si} \bar{F}_{\mathcal{B}}[k] \neq 0 \end{cases}$$

Dans ces conditions, le nombre moyen de bits utilisés pour coder un de ces coefficients est l'entropie d'une source binaire S de probabilité ρ d'être égale à 0 (et donc $\nu = 1 - \rho$ d'être égale à 1) :

$$\mathcal{H}(S) = -\rho \log_2 \rho - \nu \log_2 \nu.$$

Notons r_z le coût moyen par coefficient significatif du codage des coefficients nuls :

$$r_z = \frac{R_z}{\nu N}$$

Nous n'avons pas ici exploité les corrélations spatiales existantes entre les coefficients : l'entropie de S est donc une borne supérieure de r_z, et donc :

$$\frac{R_z}{N} \leq -\rho \log_2 \rho - \nu \log_2 \nu.$$

Comme pour $x \in [0, 1[$, nous avons $-x \log_2 x \leq (1 - x) \log_2 e$, nous en déduisons que :

$$r_z = \frac{R_z}{\nu N} \leq \log_2 e - \log_2 \nu$$

Il nous faut ajouter en réalité à cette borne le coût de codage du paramètre ρ, mais d'une part ce coût est constant, et d'autre part, en pratique, l'utilisation des *run-length*donne une valeur de r_z nettement plus basse que cette borne supérieure. De plus, les expériences numériques effectuées par Mallat et Falzon montrent que r_z est à peu près constant pour une large catégorie d'images. Nous considérons donc dans la suite que r_z est constant.

Approximation non-linéaire À basse résolution, l'hypothèse de quantification haute-résolution n'est plus valide : en effet, autour de 0, la distribution des coefficients est trop "piquée" pour que nous puissons supposer qu'elle est constante sur cette boîte. Nous allons donc traiter séparément la boîte centrale, qui annule les coefficients, et les autres boîtes.

Sullivan [48] a montré que pour une distribution laplacienne, le quantificateur optimal était quasi-uniforme, avec simplement la boîte de quantification zéro d'une taille T différente de la taille Δ des autres boîtes. Le rapport $\theta = T/\Delta$ optimal est fixe. En pratique, même si les distributions ne sont pas laplaciennes, c'est effectivement un tel schéma de quantification qui est utilisé.

Une autre façon d'envisager ce type de quantification est de considérer que nous ne codons que les νN coefficients de plus grande amplitude, en les quantifiant, et que nous abandonnons les autres. Nous nous plaçons donc dans le cadre de l'approximation non-linéaire.

Notons r_{nz} le coût moyen par coefficient significatif du codage de ces mêmes coefficients :

$$r_{nz} = \frac{R_{nz}}{\nu N}$$

Si $\nu N \gg 1$, les νN coefficients signicatifs ont un histogramme normalisé que nous pouvons interpoler continûment par :

$$p_T(x) = \frac{1}{\nu} p(x) \mathbf{1}_{\{|x|>T\}}(x).$$

Notons X_T la variable aléatoire dont p_T est la distribution. Comme nous supposons que l'hypothèse de quantification haute-résolution est valable sur les coefficients significatifs, nous avons donc :

$$r_{nz} = \mathcal{H}_d X_T - \log_2 \Delta$$

Décroissance des coefficients Dans cette partie, nous utilisons la relation $\sim$ dans le sens suivant :

$$f \sim g \iff g = O(f) \text{ et } f = O(g)$$

Nous supposons aussi que $\nu N > \frac{1}{\epsilon}$ et $\nu < \epsilon$ pour un "petit" ϵ.

Nous notons $F^r_{\mathcal{B}}[k]$ les coefficients $F_{\mathcal{B}}[k]$ réordonnés par ordre d'amplitude décroissante, c'est-à-dire :

$$k < k' \Rightarrow F^r_{\mathcal{B}}[k] \geq F^r_{\mathcal{B}}[k']$$

Mallat et Falzon font l'hypothèse que la décroissance des coefficients peut s'exprimer de la manière suivante :

$$|F_{\mathcal{B}}^r[k]| \sim k^{-\gamma(k)}, \tag{5.3}$$

Dans [29], les auteurs supposent que γ est une fonction de $\frac{k}{N}$ et en contraignant sa forme, montre que r_{nz} est une fonction de $\gamma(\nu)$ et de θ. En pratique, $\gamma(\nu)$ est à peu près constant dans le domaine de compression, et donc r_{nz} est constant.

Donnons ici la démonstration plus simple dans le cas où γ est constant et $\gamma > 0.5$ [28]. Dans le cas des images à variation totale bornée, une telle hypothèse est vérifiée, avec $\gamma = 1$. Pour des images plus irrégulières, la relation peut se vérifier approximativement avec $1/2 < \gamma \leq 1$. Le coefficient γ est lié à l'indice de régularité des espaces de Besov [28].

Comme nous mettons à zéro tous les coefficients inférieurs à T, nous déduisons de (5.3) que :

$$\nu N \sim T^{-\gamma}.$$

Dans la mesure où T et Δ sont proportionnels, il vient :

$$\nu N \sim \Delta^{-\gamma}. \tag{5.4}$$

Décomposons maintenant $R_{nz} = R_a + R_s$, où R_a est le nombre de bits utilisés pour coder les amplitudes des coefficients significatifs, et R_s leur signe. Le signe étant une information binaire, son coût de codage ne peut pas dépasser en moyenne par élément 1 bit. Nous avons donc clairement :

$$0 \leq R_s \leq \nu N.$$

Notons maintenant p_j la proportion des coefficients significatifs dans l'amplitude est quantifiée à $(j + 1/2)\Delta + T$. Si nous pouvons choisir le code de longueur optimal pour chacun des coefficients, de longueur $l_j = -\log_2 p_j$, notre budget en bits pour coder les amplitudes serait :

$$\mathcal{H}_a = -\nu N \sum_{j=0}^{+\infty} p_j \log_2 p_j. \tag{5.5}$$

Soit $n_j = \nu N p_j$ le nombre de coefficients tombant dans la boîte j. Si $|F_{\mathcal{B}}^r[k]|$ appartient à cette boîte, alors $|F_{\mathcal{B}}^r[k]| \in [(j+\theta)\Delta, (j+1+\theta)\Delta[$. En utilisant l'équation (5.3), nous obtenons :

$$n_j \sim ((j+\theta)\Delta)^{-\frac{1}{\gamma}} - ((j+\theta+1)\Delta)^{-\frac{1}{\gamma}}.$$

En vertu de l'équation (5.4), ceci signifie que :

$$p_j = \frac{n_j}{\nu N} \sim (j+\theta)^{-\frac{1}{\gamma}} - (j+\theta+1)^{-\frac{1}{\gamma}}.$$

L'expression $\sum_{j=0}^{+\infty} p_j \log_2 p_j$ est donc indépendante de νN, et l'équation (5.5) se simplifie en :

$$\mathcal{H}_a \sim \nu N$$

En réalité, nous ne connaissons pas a priori la valeur de s, qui dépend de la structure de l'image. Mais rien ne nous empêche de choisir pour définir les codes un s prédéfini (par exemple $s = 1/2$). Cela introduira une légère inefficacité, mais la relation entre R_a et νN restera approximativement linéaire. En conséquence, nous avons :

$$R_{nz} = R_a + R_s \sim \nu N,$$

ce qui signifie aussi que r_{nz} est constant par rapport à la quantification de l'image (tant que nous restons dans le bas débit).

Pour résumer : Mallat et Falzon ont découpé le débit entre R_z, le nombre de bits nécessaires pour coder les coefficients nuls après quantification, et R_{nz}, celui nécessaire pour coder les coefficients significatifs. Si nous notons respectivement r_z et r_nz ces débits rapportés au nombre de coefficients significatifs, nous constatons que ces deux valeurs sont constantes pour une image donnée.

Notons $r^* = r_z + r_{nz}$. Nous avons alors :

$$\frac{R}{N} = \frac{R_z + R_{nz}}{N} \tag{5.6}$$

$$= r_z\nu + r_{nz}\nu \tag{5.7}$$

$$= r^*(1 - \rho) \tag{5.8}$$

Le débit est donc une fonction simple de ρ, plus simple que dans le modèle de He-Kim.

Dans le cas $\gamma = 1/2$, l'équation (5.8) devient, en utilisant l'équivalence (5.4) :

$$\frac{R}{N} \sim K\frac{1}{\Delta^2},$$

où K est une constante, à comparer avec le modèle de débit sur lequel s'appuie l'algorithme du TMN.8, donné par l'équation (5.1).

5.3.3.2. Aspects pratiques

En pratique, le paramètre r^* dépend surtout du schéma de compression utilisé (type de transformée, méthode d'encodage des coefficients) mais aussi légèrement du contenu de l'image. On s'autorise à adapter la valeur de r^* en fonction des portions déjà codées de l'image, par un mécanisme de relaxation. On pourra, dans le cas de la compression vidéo, adapter la valeur de r^* en fonction de la valeur réelle pour l'image précédente.

Le modèle permet en tout cas d'obtenir très simplement, à partir de l'histogramme des coefficients, la valeur du pas de quantification nécessaire pour obtenir un débit donné. Qui plus est, cet algorithme atteint une extrêmement bonne précision de prédiction.

Il est donc vraisemblablement intéressant de voir dans quelle mesure cette précision peut apporter à la compression vidéo, dans la mesure où un algorithme similaire, mais au modèle plus pauvre (le présent modèle n'étant pas présenté dans toute son étendue théorique), bien que potentiellement encore affinable, semble donner des résultats relativement probants.

5.4. Mise en oeuvre

5.4.1. Choix du codeur

Les algorithmes du TMN.8 et de He-Kim ont été développés et testés par leurs créateurs pour le standard H.263. Malgré l'âge du codeur de H.263, le TMN.8 datant de 1997, nous avons donc adapté l'algorithme de Mallat et Falzon au sein de ce codeur.

Nous pourrions ensuite envisager d'adapter l'algorithme à un codeur plus récent, comme le codeur JM de H.264. La transposition n'est cependant pas évidente, la structure du JM est plus complexe que celle du TMN.8

5.4.2. Algorithme de He-Kim

L'implémentation de l'algorithme décrit dans 5.3.2 ne pose pas de difficulté particulière d'un point de vue structurel, si ce n'est la nécessité de réordonner certaines opérations, ou plus simplement, dans le cadre des tests, répéter certaines opérations, en particulier le calcul de DCT. En effet, à aucun moment, le calcul de l'image complète des coefficents de DCT n'est effectuée.

L'article de He-Kim laisse cependant sous silence un certain nombre de questions ou de difficultés :

- l'utilisation ou non des échantillons de chrominance
- les incertitudes quant au coût de codage du mouvement
- la différence du mécanisme de retour au niveau cible décrit dans leur article par rapport au TMN.8

5.4.2.1. Choix de l'ensemble d'échantillons

Tout d'abord, He et Kim n'expliquent pas clairement si les échantillons utilisées pour construire les histogrammes correspondent à l'ensemble des échantillons de luminance et de chrominance, ou si seuls les échantillons de luminance sont considérés. La dynamique des deux types de coefficients n'est pas nécessairement identique ; il est donc possible que, même si Q_{nz} et Q_z sont normalisés par le nombre de coefficients, que les résultats ne soient pas similaires selon le choix effectué. Les deux solutions ont été implémentées, par prudence.

5.4.2.2. Estimation du coût de codage du champ de mouvement

Ensuite, l'article de He et Kim suppose que le coût du mouvement est entièrement connu avant la quantification des coefficients. Ceci n'est pas vrai en pratique. En effet, dans le cas où le mouvement est nul sur un bloc, et qu'après quantification, tous les coefficients sont nuls, le bloc est considéré comme sauté, ce qui est un cas particulier de la syntaxe. Que faire ? Il n'est pas possible de savoir a priori quels sont les blocs qui seront sautés, et faire une estimation selon la valeur du mouvement et des coefficients avant quantification reste relativement hasardeux.

Dans la mesure où le pas de quantification s'adapte dans l'algorithme de He-Kim, en fonction du budget de bits restants, la solution adoptée est la suivante : dans un premier temps, on travaille comme si le mode non-codé n'existait pas, ce qui donne une première estimation du budget de bits, en pratique donc éventuellement sous-évaluée ; ensuite, au fur et à mesure que les blocs sont parcourus pour le codage, on rajoute au budget le coût de codage des vecteurs qui n'ont finalement pas été codés.

Une nouvelle fois, par mesure de précaution, la solution où le budget n'est pas réajusté au cours du codage a été elle aussi mise en oeuvre.

5.4.2.3. Contrôle du tampon

Il y a une différence de stratégie entre du mécanisme de contrôle du niveau du tampon à l'échelle des trames tel qu'il est décrit dans l'article de He-Kim et le mécanisme effectivement implémenté dans les sources utilisées, qui correspond à celui décrit en 5.3.1. En effet, l'article de He-Kim suppose que le retour au taux de remplissage cible du tampon se fait sans amortissement (c'est à dire $\Delta = W - AD/F$ quelque soit la valeur de W par rapport à AM). En pratique, nous avons préféré gardé le système en place dans les sources, qui est celui que l'on retrouve dans la littérature.

5.4.3. Algorithme de Mallat et Falzon

Les mêmes questions d'implémentation (choix des coefficients à utiliser pour l'histogramme, estimation du mouvement) se posent pour la mise en oeuvre de cet algorithme que pour celui de He-Kim, dans la mesure où notre objectif est de simplement remplacer le modèle $\rho \mapsto R$ de He-Kim par celui de Mallat et Falzon. Cependant, comme nous sommes libres d'adapter les valeurs des paramètres généraux du modèle, seuls les solutions qui nous ont paru les plus judicieuses ont été expérimentées, c'est-à-dire la prise en compte de l'ensemble des coefficients (luminance et chrominance) pour la construction des histogrammes, et la mise à jour au fur et à mesure du codage du budget de bits en fonction de la correction de l'erreur d'estimation du coût de codage du champ de mouvement ont été systématiquement utilisés.

Afin d'analyser les performances du modèle de Mallat et Falzon, deux options ont été mises en oeuvre.

5.4.3.1. Pas de quantification fixe ou variable sur une trame

Afin de déterminer l'intérêt du mécanisme de variation du pas de quantification, il est possible d'utiliser un pas de quantification constant sur l'image, plutot que de tenter de mettre à jour celui-ci en fonction des zéros déjà obtenus et du budget de bits restant. En effet, il ne faut pas oublier que le changement de pas de quantification au sein d'une trame demande un supplément d'information à transmettre. Il n'est donc pas sûr a priori qu'il soit particulièrement bénéfique au codage, et il s'agit dans l'absolu de déterminer ce que celui-ci apporte.

5.4.3.2. Variation du paramètre r^*

Comme mentionné en 5.3.3, il est possible d'essayer d'adapter le paramètre r^* en fonction de l'image. Une fois qu'une image est codée, il est facile de déterminer exactement la valeur correcte de r^* pour cette image. Le mécanisme de contrôle de r^* est alors le suivant. Supposons qu'on connait les valeurs correctes de r^* pour les images $t-1$, $t-2$ et $t-3$, notées r_1^*, r_2^*, r_3^*. On utilise alors comme valeur pour r^* dans l'image t :

$$\tilde{r}^* = 0.6r_1^* + 0.3r_2^* + 0.1r_3^*.$$

Cette fonction peut être désactivée afin de diminuer le temps de calcul. Cependant, la qualité du codage dépend donc ensuite essentiellement de la qualité de la valeur initiale de $r^* = r_0^*$.

On dispose donc de 4 sous-modes pour l'algorithme de Mallat et Falzon : QP constant + r^* constant, QP constant + r^* variable, QP variable + r^* constant, QP variable + r^* variable.

5.4.4. Complexité de calcul

Nous notons ici L le nombre de valeurs absolues possibles pour les coefficients de DCT [environ 2000]. Rappelons afin de fixer un ordre d'idée que pour une image QCIF, le nombre de pixels de l'image est $N = 25344$ et le nombre de macroblocs est $N_{mb} = 99$.

5.4.4.1. TMN.8

Le principal coût de l'algorithme de contrôle de débit du TMN.8 se trouve dans le calcul de la moyenne et de la variance sur chaque bloc de l'image, et de paramètres simples en découlant. Si l'on décompose de façon plus complète :

- Initialisation : calcul de la variance et de la moyenne. Coût : $\frac{3}{2}Nc_1 + N_{mb}c_2$
- Calcul du pas de quantification. Coût : $N_{mb}c_3$
- Mise à jour du système de contrôle. Coût : $N_{mb}c_4$.

Soit un coût sous la forme NC_{tmn}

5.4.4.2. He-Kim

Il y a trois éléments importants dans le mécanisme de contrôle de débit de He-Kim : la construction de l'histogramme des coefficients d'une part, le calcul de la table de correspondance $QP \mapsto \rho$ et la détermination du paramètre ϑ. En réalité, il y a non pas un, mais 4 histogrammes à gérer, selon deux caractéristiques des coefficients : inter/intra et luma/chroma, ce qui impose un test supplémentaire lors de la construction des histogrammes, quatre fois plus d'opérations pour la détermination de la table et quatre fois plus d'opérations lors de la détermination du paramètre ϑ, puisque cela signifie qu'il faut calculer Q_nz à partir de quatre histogrammes différents, pour lesquels les quantifications ne sont pas nécessairement identiques. Cependant, on peut raisonnablement affirmer que - dans la mesure où l'on ne prend pas en compte les opérations redondantes effectuées

dans le programme de test, et facilement éliminables dans une version optimisée - la complexité n'est pas significativement différente de celle du TMN.8, relativement à l'ensemble de l'algorithme de compression.

En pratique, cela donne :

- Estimation du coût du mouvement. Coût : $N_{mb}h_1$
- Construction des histogrammes. Coût : $\frac{3}{2}Nh_2$
- Construction des fonctions de répartition. Coût : Lh_3
- Construction de la table $\rho - R$. Coût : $32h_4$
- Obtention de ϑ. Coût : Lh_5
- Construction de la courbe $R - QP$ et obtention de QP_0. Coût : h_6
- Mise à jour du système de contrôle : $N_{mb}h_7$
- Calcul du pas de quantification : $N_{mb}h_8$

Soit un coût de la forme $NC_{h1} + LC_{h2}$.

5.4.4.3. Mallat-Falzon

L'adaptation du modèle de Mallat et Falzon a une structure relativement similaire à celle de l'algorithme, mais l'étape de détermination du paramètre θ a cependant été éliminée. D'autre part, le calcul d'un point sur la courbe $\rho \mapsto R$ lors de la détermination du pas de quantification et de sa mise à jour est plus simple que dans le cas du modèle de He-Kim. Il s'agit en effet simplement de trouver un point sur une droite, plutot que de le trouver sur une courbe linéaire par morceau (ce qui impose de déterminer préalablement le morceau correct). La complexité de l'algorithme est donc plus faible que celle de He-Kim, essentiellement, grace à la diminution des opérations sur les histogrammes, et au fait que l'estimation de θ n'est pas à faire, qui est assez coûteuse. En pratique, cela donne :

- Estimation du coût du mouvement. Coût : $N_{mb}h_1$
- Construction des histogrammes. Coût : $\frac{3}{2}Nh_2$
- Construction des fonctions de répartition. Coût : Lh_3
- Construction de la table $\rho - QP$. Coût : $32h_4$
- Construction de la courbe $\rho - R$ et obtention de QP_0. Coût : a_6
- Mise à jour du système de contrôle : $N_{mb}a_7$
- Calcul du pas de quantification : $N_{mb}a_8$

Avec $a_6 < h_6$, $a_7 < h_7$ et $a_8 < h_8$. Soit un coût de la forme $NC_{a1} + LC_{a2}$, avec $C_{a1} < C_{h1}$ et $C_{a2} < C_{h2}$.

5.4.4.4. Comparaison

En partant de la dernière remarque du paragraphe précédent, il est clair que l'algorithme de Mallat et Falzon a une complexité plus faible que celui de He-Kim. La comparaison entre la complexité du TMN.8 et de celle des algorithmes de He-Kim et de Mallat et Falon est plus difficile. Il est cependant vraisemblable que la construction des 4 histogrammes, de la table de correspondance entre ρ et le pas de quantification soit finalement plus coûteuse que le calcul de variance de l'image. D'autre part, le mécanisme de mise à jour du pas de quantification au sein de la trame est lui aussi plus coûteux.

Quoiqu'il en soit, ces coûts restent marginaux comparés aux coûts de la DCT, et surtout de la recherche du mouvement.

5.4.5. Conclusion

La mise en oeuvre de l'algorithme de Mallat et Falzon, qui est en grande partie neuve, avec des composantes simples, est relativement aisée. En revanche, celle de l'algorithme de He-Kim est plus complexe suite aux imprécisions quant au contenu du modèle, et à certains flous difficiles à résoudre. Par prudence, un certain nombre d'options ont été rendues possibles afin d'essayer de déterminer la meilleure combinaison, plutôt que de se fixer sur une seule a priori.

5.5. Résultats et Analyse

5.5.1. TMN.8

Avant toute chose, il est nécessaire de constater que les résultats obtenus par le TMN.8 utilisé ne correspondent pas à ceux mentionnés dans l'article de He-Kim, et ceci, quelque soit le jeu d'option utilisé. Cela pourrait être vraiment inquiétant si la littérature ne donnait pas l'impression qu'il y a autant de résultats différents que d'articles sur le sujet. En particulier, l'article décrivant le système de contrôle de débit du TMN.8 [39] donne des résultats différents que ceux donnés dans [18] et de ceux obtenus avec les sources dont nous nous sommes servies.

Nous avons choisi de travailler uniquement avec les valeurs que nous avons obtenues par nos séries de test, sans essayer de comparer avec les résultats d'autres articles.

5.5.2. Tests effectués

5.5.2.1. Jeux d'options du standard H.263

Partant du principe que les jeux d'options ne nous étaient pas imposés, vu les constations précédentes, nous avons choisi d'utiliser le moins possible d'options dans l'utilisation du codeur H263.

	Correction du coût des vecteurs	
	Avec	Sans
Luma+Chroma	HeKim0	HeKim4
Luma seulement	HeKim8	HeKim12

TAB. 5.1.: Table de paramètres pour l'algorithme de He-Kim

5.5.2.2. Séquences

Cependant, afin de nous rapprocher des conditions expérimentales utilisées par He-Kim, nous avons utilisé les mêmes séquences, avec les mêmes débits cibles. Les séquences sont toutes au format QCIF@30Hz et comportent 300 images. Il s'agit de (avec le débit cible à l'encodage) :

- Mother and Daughter (M. & D.) à 16 kbps,
- Carphone à 32 kbps,
- Container à 32 kbps,
- Salesman à 32 kbps,
- Coastguard à 32 kbps,
- News à 48 kbps,
- Foreman à 48 et 64 kbps

Les séquences sont toutes codées à une fréquence cible de 10 Hz, ce qui signifie que le programme n'essaie de coder qu'une trame sur trois de la séquence originale. En pratique, cette fréquence pourra être plus faible au cas où certaines images sont sautées par le système lorsque le tampon est trop rempli. Seule la première image est codée dans un mode intra, toutes les autres étant codées en inter.

5.5.2.3. He-Kim

Pour l'algorithme de He-Kim, nous avons deux paramètres à choisir (cf. 5.4.2) :

- correction du coût des vecteurs au fur et à mesure du codage, ou pas
- utilisation des coefficients de chrominance, ou pas

Les 4 combinaisons ont été expérimentées, selon la table 5.1.

5.5.2.4. Mallat-Falzon

Pour l'algorithme de Mallat et Falzon, nous avons deux paramètres à choisir (cf. 5.4.3) :

- l'utilisation du paramètre de quantification variable au sein de la trame (QP variable), ou non
- l'utilisation d'une boucle d'adaptation de r^*, afin de se rapprocher au mieux des données de l'image, ou non

Les 4 combinaisons ont été expérimentées, selon la table 5.2.
D'autre part, nous avons essayé comme valeur de r_0^* 6,7 et 8.

	QP variable	QP constant
r^* constant	MF0	MF1
r^* variable	MF2	MF3

TAB. 5.2.: Table de paramètres pour l'algorithme de Mallat-Falzon adapté

5.5.2.5. Présentation des résultats

La qualité d'un mécanisme de planification de débit pour la vidéo peut être évaluée à l'aide des éléments suivants :

- nombre de trames perdues
- PSNR
- réalisation de la consigne de débit
- remplissage du tampon

PSNR Parmi les données extraites, la mesure de la qualité est une des plus importantes. Le PSNR donné correspond à la moyenne sur les images inter effectivement codées. La première image, codée nécessairement en intra, ainsi que les trames sautées ne sont pas prises en compte.

Images non-codées Dans le cadre d'un système de contrôle de débit, on s'intéresse nécesairement à la qualité de contrôle afin qu'il n'y ait pas de débordement trop courant du tampon. Cette qualité peut être mesurée par le dénombrement des images non-codées.

On distinguera le nombre total d'images non-codées, du nombre d'images non-codées après la première image inter (qui sera noté entre parenthèse). En effet, les images intra, qui ne peuvent utiliser de redondance temporelle, sont par nature plus coûteuses à encoder que les images inter à qualité égale. Cela rend plus élevés les risques de débordement suite au codage d'une trame intra.

D'autre part, le TMN.8 ne dispose pas d'un mécanisme de contrôle de débit pour les trames intra, contrairement à l'algorithme de He-Kim et de celui de Mallat et Falzon. Il y a donc toutes les chances pour que le nombre d'images perdues suite à la première image soit plus élevé dans le TMN.8.

On souhaite donc pouvoir juger plus équitablement de la qualité de contrôle sur les trames les plus courantes, c'est à dire les inter.

Il faut aussi noter que lorsque le nombre d'images non-codées augmente, le PNSR a toutes les chances d'augmenter lui aussi dans la mesure où le même nombre de bits est alloué au codage d'un nombre plus faible de trames. Il s'agit donc de ne pas se laisser abuser par les résultats en terme de PSNR et de considérer globablement les deux éléments, PSNR et nombre de trames non-codées.

Débit réalisé Si l'on souhaite éviter les débordements du tampon, on souhaite aussi éviter que le système de contrôle de débit sous-utilise la bande passante à sa disposition. La capacité du système à réaliser ceci peut être mesurée en notant le débit effectivement

Séquence	débit cible	PSNR				
		TMN8	HeKim0	HeKim4	HeKim8	HeKim12
Foreman	48	30.29	30.30	30.21	30.31	30.19
Foreman	64	31.38	31.21	31.47	31.24	31.45
M. & D.	16	31.45	32.23	32.00	32.31	31.95
Salesman	32	33.10	32.59	33.02	32.59	33.00
Carphone	32	31.09	31.01	30.79	30.95	30.79
Coastguard	32	27.30	27.59	27.38	27.61	27.34
News	48	33.16	32.46	33.38	32.48	33.39
Container	32	32.88	32.54	33.53	32.38	33.50

TAB. 5.3.: Algorithme d'He-Kim : PSNR

Séquence	débit cible	Images non-codees				
		TMN8	HeKim0	HeKim4	HeKim8	HeKim12
Foreman	48	4(2)	1(1)	1(1)	1(1)	1(1)
Foreman	64	2(1)	0(0)	0(0)	0(0)	0(0)
M. & D.	16	9(2)	29(26)	20(17)	32(29)	19(16)
Salesman	32	4(0)	6(5)	10(9)	6(5)	11(10)
Carphone	32	4(0)	1(0)	1(0)	1(0)	1(0)
Coastguard	32	5(2)	3(2)	2(1)	3(2)	3(2)
News	48	3(0)	1(0)	1(0)	1(0)	1(0)
Container	32	5(0)	2(0)	2(0)	2(0)	5(3)

TAB. 5.4.: Algorithme d'He-Kim : Images non-codées

atteint lors de la compression. Celui-ci, s'il peut être légérement plus élevé que le débit cible à cause d'un remplissage du tampon plus important en fin de séquence qu'au départ, sera inférieur en cas de vide temporaire du tampon. Le débit atteint en comptant toutes les trames et celui en comptant seulement la compression des trames à partir de la première trame inter seront donnés, toujours afin de relativiser l'influence de la trame intra initiale.

Courbes de remplissage du tampon Les courbes de remplissage du tampon permettent de constater dans quelle mesure le système de contrôle de débit est capable de suivre correctement la consigne qui lui a été fournie pour une trame, ou bien s'il se "trompe" régulièrement de sorte qu'il est obligé de compenser les trames suivantes. Qui plus est, cela permet de repérer l'existence de trames difficiles que tous les systèmes ont du mal à comprimer en respectant la consigne.

5.5.3. He-Kim

5.5.3.1. Comparaison avec le TMN.8

Les résultats obtenus sont les suivants :

Il est difficile d'éclaircir à l'aide des résultats les incertitudes mentionnées en 5.4.2.1, puisque les résultats sont alternativement plus ou moins bons selon les séquences pour

Séquence	débit cible	Débit réel				
		TMN8	HeKim0	HeKim4	HeKim8	HeKim12
Foreman	48	48.47 (46.63)	46.24 (45.30)	45.60 (44.66)	46.43 (45.49)	45.29 (44.35)
Foreman	64	64.24 (62.40)	59.50 (58.57)	62.50 (61.57)	59.69 (58.75)	62.27 (61.34)
M. & D.	16	16.02 (14.62)	16.03 (15.29)	16.01 (15.27)	16.10 (15.35)	16.06 (15.31)
Salesman	32	31.22 (29.51)	29.63 (28.82)	30.71 (29.90)	29.60 (28.80)	30.59 (29.77)
Carphone	32	32.02 (30.29)	30.38 (29.49)	29.20 (28.31)	30.04 (29.15)	29.13 (28.23)
Coastguard	32	31.82 (30.25)	30.66 (29.87)	29.49 (28.71)	30.64 (29.85)	29.76 (28.97)
News	48	46.60 (44.44)	39.33 (38.27)	46.25 (45.19)	39.66 (38.60)	46.44 (45.38)
Container	32	26.71 (24.61)	26.13 (25.10)	31.42 (30.39)	25.30 (24.27)	31.11 (30.08)

TAB. 5.5.: Algorithme d'He-Kim : Débit réel

l'une ou l'autre variante.

Les résultats des séquences *Mother and Daughter* et *Salesman* sont médiocres, au vu du nombre d'images sautées. Le comportement erratique du mécanisme de contrôle est particulièrement visible sur les courbes de remplissage du tampon (5.4) pour ces deux séquences. La présence des pics correspond généralement à une image perdue. Pour ce même critère d'évaluation et pour toutes les autres séquences, l'algorithme de He-Kim fait mieux que celui du TMN.8. Si l'on met de côté les deux séquences mentionnées précédemment, pour entre 4 séquences sur 6 pour les moins bonnes variantes et 2 sur 6 pour la meilleure variante, les résultats en terme de PSNR sont moins bons que pour le TMN.8, même si le nombre d'images non-codées est constamment inférieur dans l'algorithme de He-Kim pour les 6 séquences.

5.5.3.2. Considérations sur le modèle de He-Kim

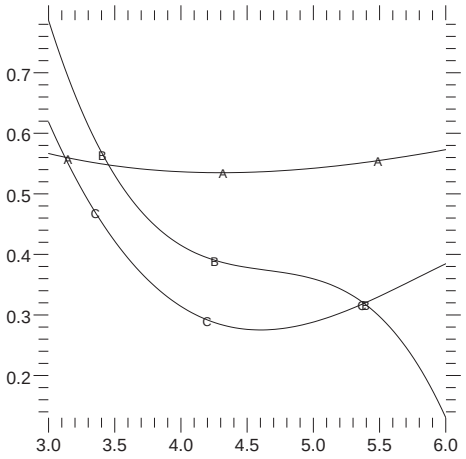


FIG. 5.2.: Courbe $R(\vartheta, \rho)$ pour A : $\rho = 0.96$, B : $\rho = 0.97$, C : $\rho = 0.98$

Si nous traçons les valeurs de R pour les 9 points du modèle de He-Kim en fonction du paramètre ϑ, nous pouvons constater le fait suivant : ce modèle n'est valide que dans le

Séquence	Débit cible	PSNR				
		TMN8	MF0	MF1	MF2	MF3
Foreman	48	30.29	30.71	30.70	30.56	30.56
Foreman	64	31.38	31.76	31.78	31.69	31.71
M. & D.	16	31.45	31.80	31.79	31.47	31.43
Salesman	32	33.10	32.83	32.70	32.42	32.37
Carphone	32	31.09	31.40	31.36	31.19	31.15
Coastguard	32	27.30	27.79	27.79	27.62	27.60
News	48	33.16	33.40	33.37	33.31	33.27
Container	32	32.88	33.21	33.10	32.96	32.69

TAB. 5.6.: Algorithme de Mallat-Falzon ($r_0^* = 6.7$) : PSNR

Séquence	Débit cible	Images non-codées				
		TMN8	MF0	MF1	MF2	MF3
Foreman	48	4(2)	4(3)	5(4)	2(1)	2(1)
Foreman	64	2(1)	1(1)	1(1)	1(1)	1(1)
M. & D.	16	9(2)	8(5)	9(6)	4(1)	4(1)
Salesman	32	4(0)	1(0)	2(1)	1(0)	1(0)
Carphone	32	4(0)	4(3)	3(2)	1(0)	1(0)
Coastguard	32	5(2)	11(10)	9(8)	2(1)	2(1)
News	48	3(0)	1(0)	1(0)	1(0)	1(0)
Container	32	5(0)	2(0)	2(0)	2(0)	2(0)

TAB. 5.7.: Algorithme de Mallat-Falzon ($r_0^* = 6.7$) : Images non-codées

cas où $3.5 \lessapprox \vartheta \lessapprox 5.4$. Hors de ces bornes, les courbes $\rho \mapsto R$ ne sont plus décroissantes, ce qui est absurde.

En effet, le fait que ρ augmente signifie que l'on simplifie l'information contenue dans l'image. Il s'ensuit logiquement que R devrait diminuer quand ρ augmente.

Sur le graphique 5.2, cela signifie que la courbe A devrait se trouver au-dessus de la courbe B, et celle-ci au-dessus de la courbe C. En réalité, ce n'est le cas que dans l'intervalle de valeurs mentionné précédemment. En observant la répartition des valeurs de ϑ utilisées lors de la compression, nous nous apercevons que nombre d'entre elles ne sont pas dans l'intervalle où le modèle est correct. Il y a donc clairement un problème à ce niveau-là qui, malgré toute l'attention portée, ne semble pas venir d'un problème de programmation.

5.5.4. Mallat-Falzon

Les résultats obtenus sont données dans les tables 5.5.4 à 5.5.4

Séquence	Débit cible	Débit réel				
		TMN8	MF0	MF1	MF2	MF3
Foreman	48	48.47 (46.63)	48.27 (47.30)	48.30 (47.33)	48.26 (47.29)	47.63 (46.66)
Foreman	64	64.24 (62.40)	64.51 (63.36)	64.57 (63.41)	64.17 (63.02)	63.54 (62.38)
M. & D.	16	16.02 (14.62)	16.11 (15.36)	16.10 (15.34)	14.90 (14.14)	14.82 (14.07)
Salesman	32	31.22 (29.51)	29.84 (29.01)	29.12 (28.30)	27.48 (26.65)	27.28 (26.46)
Carphone	32	32.02 (30.29)	32.02 (31.12)	32.01 (31.12)	31.77 (30.87)	31.50 (30.60)
Coastguard	32	31.82 (30.25)	31.92 (31.04)	31.73 (30.86)	31.13 (30.26)	31.06 (30.19)
News	48	46.60 (44.44)	47.45 (46.38)	47.24 (46.17)	46.44 (45.37)	46.16 (45.10)
Container	32	26.71 (24.61)	30.11 (29.08)	29.28 (28.25)	28.52 (27.49)	27.13 (26.10)

TAB. 5.8.: Algorithme de Mallat-Falzon ($r_0^* = 6.7$) : Débit réel

Séquence	Débit cible	PSNR				
		TMN8	MF0	MF1	MF2	MF3
Foreman	48	30.29	30.51	30.47	30.56	30.54
Foreman	64	31.38	31.56	31.52	31.73	31.71
M. & D.	16	31.45	31.63	31.63	31.47	31.43
Salesman	32	33.10	32.36	32.33	32.42	32.41
Carphone	32	31.09	31.15	31.14	31.19	31.15
Coastguard	32	27.30	27.63	27.64	27.61	27.57
News	48	33.16	32.85	32.65	33.31	33.27
Container	32	32.88	32.65	32.60	32.96	32.69

TAB. 5.9.: Algorithme de Mallat-Falzon ($r_0^* = 8.0$) : PSNR

Séquence	Débit cible	Images non-codées				
		TMN8	MF0	MF1	MF2	MF3
Foreman	48	4(2)	1(1)	1(1)	2(2)	1(1)
Foreman	64	2(1)	0(0)	0(0)	1(1)	1(1)
M. & D.	16	9(2)	4(1)	4(1)	4(1)	4(1)
Salesman	32	4(0)	1(0)	1(0)	1(0)	1(0)
Carphone	32	4(0)	1(0)	1(0)	1(0)	1(0)
Coastguard	32	5(2)	2(1)	2(1)	2(1)	2(1)
News	48	3(0)	1(0)	1(0)	1(0)	1(0)
Container	32	5(0)	2(0)	2(0)	2(0)	2(0)

TAB. 5.10.: Algorithme de Mallat-Falzon ($r_0^* = 8.0$) : Images non-codées

Séquence	Débit cible	Débit réel				
		TMN8	MF0	MF1	MF2	MF3
Foreman	48	48.47 (46.63)	47.97 (47.04)	47.36 (46.43)	47.84 (46.91)	47.47 (46.54)
Foreman	64	64.24 (62.40)	63.47 (62.42)	61.50 (60.45)	64.11 (63.07)	63.43 (62.37)
M. & D.	16	16.02 (14.62)	15.91 (15.16)	15.81 (15.05)	14.90 (14.14)	14.82 (14.07)
Salesman	32	31.22 (29.51)	26.89 (26.08)	26.74 (25.92)	27.57 (26.75)	27.80 (26.99)
Carphone	32	32.02 (30.29)	31.54 (30.64)	31.35 (30.45)	31.77 (30.87)	31.50 (30.60)
Coastguard	32	31.82 (30.25)	31.69 (30.87)	31.57 (30.75)	31.20 (30.39)	30.64 (29.82)
News	48	46.60 (44.44)	42.90 (41.84)	41.54 (40.48)	46.44 (45.37)	46.16 (45.10)
Container	32	26.71 (24.61)	26.38 (25.35)	26.11 (25.08)	28.52 (27.49)	27.13 (26.10)

TAB. 5.11.: Algorithme de Mallat-Falzon ($r_0^* = 8.0$) : Débit réel

5.5.4.1. Comparaison entre les différentes variantes

5.5.4.2. Valeur initiale de r^*

Comme on peut s'y attendre, le choix du paramètre initial r_0^* n'influe pas énormément sur les résultats dans le cas où l'on est capable de faire varier r^*. En revanche, dans le cas où celui-ci est fixe, on peut noter une franche différence : l'équilibre entre nombre d'images codées et PSNR se déplace en faveur du premier lorsqu'on augmente la valeur de r^*. Cela est intéressant quand le nombre d'images non-codées est important, mais dans le cas où celui-ci est faible, le nombre d'images perdues ne peut plus nécessairement être diminué, et on aurait donc préféré avoir un meilleur PSNR.

On peut voir sur les courbes de remplissage du tampon l'effet d'un r^* trop faible, en particulier pour Coastguard : le mécanisme de contrôle a tendance à toujours sous-estimer le coût de l'image, et donc à utiliser plus de bits que le budget alloué. On a donc régulièrement un cycle remplissage du tampon-saut d'image (et donc vidage partiel du tampon).

5.5.4.3. r^* variable ou constant

L'utisation de la variante ou l'on fait évoluer r^* permet d'essayer d'atteindre un compromis entre les différents cas où l'on se fixe une valeur de r^* pour toute la séquence. En particulier, on est à peu près sûr d'obtenir un nombre faible d'images non-codés (dans 7 séquences sur 8, le nombre de trames perdues est le même qu'avec la valeur haute de r_0^* lorsque ce paramètre est constant sur la séquence), tout en améliorant dans la plupart des cas le PSNR par rapport à un r^* constant qui nous aurait donné le même nombre de trames non-codées (amélioration du PSNR dans 6 séquences sur 8).

En pratique, cela se traduit aussi par une amélioration du suivi de la consigne, puisque on ne se trouve que rarement, à la fois en débordement du tampon (perte de trame) et en assèchement de celui-ci (sous-utilisation de la bande-passante). Cela se voit aussi sur les courbes de remplissage du tampon. Les courbes correspondant aux modes MF2 et MF3 (donc avec un r^* variable) sont celles qui suivent le mieux la consigne, évitant trop souvent d'être au-dessus de la consigne, aussi bien qu'en underflow.

5.5.4.4. Pas de quantification variable ou constant

Le mécanisme de variation du pas de quantification joue peu dans la conservation des trames. Les résultats sont en effet quasiment similaires entre les deux variantes, sauf dans le cas où le paramètre r^* est fixé et faible. Dans ce dernier cas, il est difficile de conclure, puisque pour 2 séquences, il y a amélioration lors de l'introduction du système et, dans 2 autres, dégradation.

Si ce système n'a que peu d'influence sur le nombre d'images perdues, il a en revanche une sensiblement positive pour ce qui concerne le PSNR. La différence est généralement assez faible, de l'ordre de 0.05 dB, et dans quelques rares cas, est même en faveur du pas de quantification constant[2]. Il peut arriver qu'elle monte jusqu'à 0.27 dB, ce qui reste appréciable, puisque le système de variation du pas de quantification n'augmente pas démesurément la complexité.

Si l'on regarde les tables de débits effectivement réalisés, on peut noter que l'utilisation du système de variation du pas de quantification se traduit par une meilleure utilisation de la bande passante : les débits réalisés sont plus proches des consignes lorsque l'on permet une légère variation du pas de quantification au sein d'une même trame.

5.5.4.5. Choix de la variante de référence

Il semble donc que le meilleur mode soit celui où l'on choisit un paramètre r^* plutôt élevé initialement, mais variable, tout en activant le mécanisme du pas de quantification au sein d'une même trame. Cela correspond au mode MF2, avec $r_0^* = 8.0$. Pour la suite des comparaisons, on considérera ce mode comme mode de référence.

5.5.4.6. Comparaison avec le TMN.8

L'algorithme basé sur le modèle de Mallat et Falzon donne toujours un résultat au moins aussi bon que le TMN.8 en terme de conservation des trames, y compris en terme de nombre de trames sautées après la première image inter. D'autre part, dans 7 séquences sur 8, cela se combine avec une amélioration de la qualité de la vidéo compressée, en terme de PSNR. Cependant, la perte de qualité dans la dernière séquence (*Salesman*) est assez importante.

Nous pouvons néanmoins considérer que les résultats du modèle de Mallat et Falzon sont probants vis à vis de ceux du TMN.8.

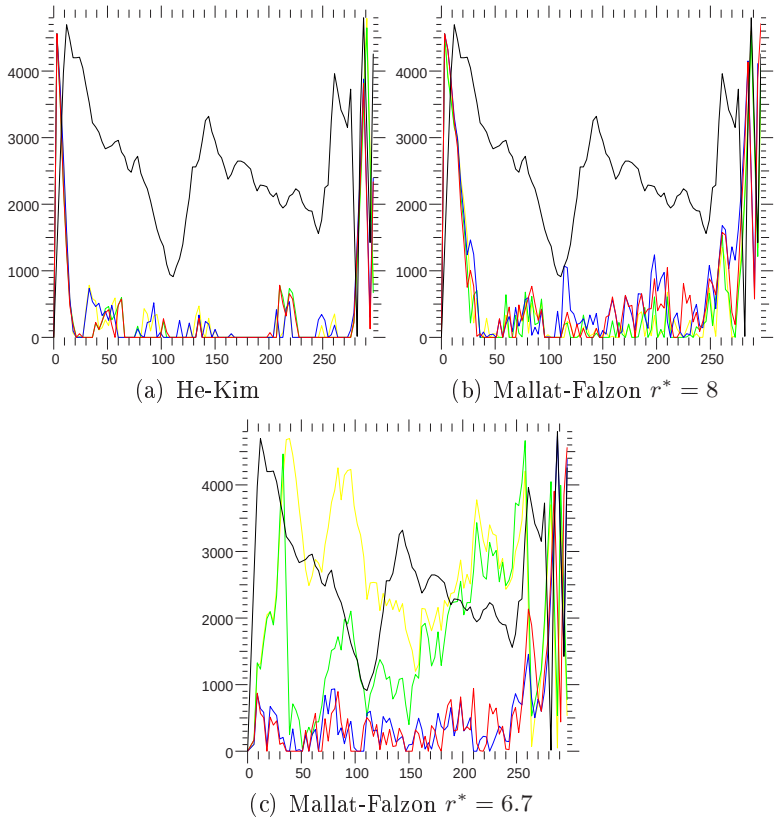
Si nous étudions les courbes de remplissage du tampon, nous voyons que, dans certains cas (figure 5.3), l'algorithme de Mallat et Falzon n'utilise pas le tampon aussi bien que le TMN.8. Il est donc peut-être possible d'introduire des améliorations pour rendre l'algorithme de Mallat et Falzon encore plus performant. En revanche, dans le cas de la séquence *Salesman* (figure 5.4), une des raisons du manque de compétitivité du TMN.8 est justement le fait qu'il asséche le tampon.

[2]Ce qui n'est pas complètement étonnant, puisque faire varier le pas de quantification a un coût

5.5.4.7. Comparaison avec le modèle de He-Kim

En terme de trames perdues, l'adaptation de l'algorithme de Mallat et Falzon est toujours meilleur que l'algorithme de He-Kim, mis à part pour la séquence *Foreman*. En terme de PSNR, la comparaison est là toujours en faveur de l'algorithme de Mallat-Falzon, mis à part dans les séquences où l'algorithme de He-Kim produit un nombre élevé de trames perdues..

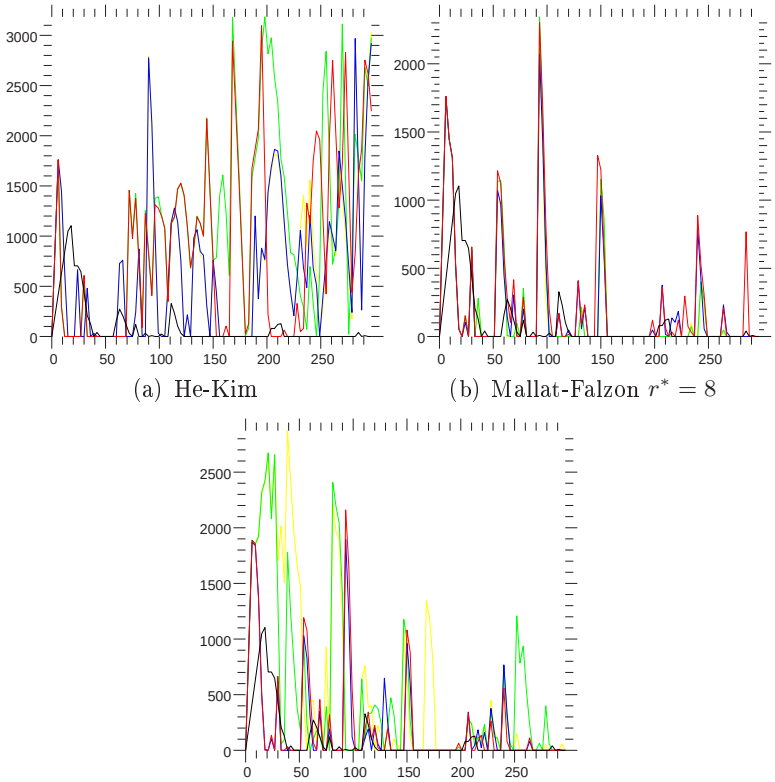
Lorsque nous regardons les courbes de remplissage du tampon, nous voyons que l'algorithme de He-Kim a un comportement plus extrême que celui de Mallat et Falzon : il a soit plus tendance à faire déborder le tampon (figure 5.4) soit plus tendance à l'assécher (5.3)



(a) He-Kim (b) Mallat-Falzon $r^* = 8$

(c) Mallat-Falzon $r^* = 6.7$

FIG. 5.3.: Remplissage du tampon pour *Foreman*@48kbps. Légende pour He-Kim : Noir : TMN.8, Jaune : avec Chrominance et correction des coûts des vecteurs ; Vert : avec Chrominance et sans correction du coût des vecteurs ; Bleu : sans Chrominance et avec correction du coût des vecteurs ; Rouge : sans Chrominance ni correction du coût des vecteurs ; Légende pour Mallat-Falzon : Noir : TMN.8, Jaune : r^* et QP constants ; Vert : r^* constant, QP variable ; Bleu : r^* variable, QP constant ; Rouge : r^* et QP variables



(a) He-Kim (b) Mallat-Falzon $r^* = 8$

(c) Mallat-Falzon $r^* = 6.7$

FIG. 5.4.: Remplissage du tampon pour *Salesman*@32kbps. Légende pour He-Kim : Noir : TMN.8, Jaune : avec Chrominance et correction des coûts des vecteurs ; Vert : avec Chrominance et sans correction du coût des vecteurs ; Bleu : sans Chrominance et avec correction du coût des vecteurs ; Rouge : sans Chrominance ni correction du coût des vecteurs ; Légende pour Mallat-Falzon : Noir : TMN.8, Jaune : r^* et QP constants ; Vert : r^* constant, QP variable ; Bleu : r^* variable, QP constant ; Rouge : r^* et QP variables

5.5.5. Conclusion

Suite aux tests effectués, il semble que l'algorithme de Mallat et Falzon s'adapte correctement à la vidéo et à la compression de trame d'erreur. Les résultats obtenus sont clairement meilleurs à la fois que ceux de l'algorithme du TMN.8 et que ceux de l'algorithme de He-Kim. Néanmoins, les incertitudes quant aux détails pratiques de l'implémentation de ce dernier font qu'il est difficile de dire si la modélisation de Mallat et Falzon est supérieure à celle de He et Kim. Les résultats fournis par ces derniers dans leur article semblent meilleurs que ceux obtenus, en terme de maintien du remplissage du tampon autour de son niveau cible, de trames perdues et de PSNR. L'algorithme de Mallat et Falzon a dans tous les cas l'avantage d'avoir une mise en place plus simple.

5.6. Perspectives

5.6.1. Adaptation de l'algorithme dans H264

Le monde de la compression vidéo est en permanente évolution, et de nouveaux standards émergent. Il faut donc s'interroger sur les posssibilités d'exploiter l'algorithme développé au sein du standard le plus récent, en l'occurrence H.264, aussi connu sous le nom de MPEG4-AVC.

La transposition directe de l'algorithme n'est pas possible dans le codeur de test, car la structure, bien que globalement assez similaire, présente néanmoins quelques différences entre les codeurs pour H.263 et H.264. En effet, de nouveaux outils ont été mis en place, en particulier des outils d'optimisation débit-distorsion. Ces derniers utilisent comme paramètre le pas de quantification pour équilibrer la contrainte entre le coût de codage de l'erreur et le coût de codage du mouvement. Ils opèrent bloc après bloc pour obtenir les vecteurs de mouvement, en minimisant :

$$S(v) = \sum |\text{coeff}_{err}| + \lambda(QP)C(v),$$

où λ est une fonction croissante du paramètre de quantification, et où $C(v)$ est le coût de codage prédictif du vecteur à tester. L'algorithme utilisé dans le TMN8 et celui de Mallat et Falzon supposent eux une connaissance complète de l'image à coder.

Comment donc adapter le système ? L'idée la plus simple venant à l'esprit est de supposer que le pas de quantification moyen ne varie pas brutalement d'une trame à l'autre. On pourrait donc prendre le pas de quantification utilisé dans l'image précédente comme paramètre pour les outils d'optimisation débit-distorsion, puis, connaissant désormais l'image d'erreur, coder effectivement les blocs à l'aide du pas de quantification calculé selon le modèle de Mallat et Falzon.

On peut raffiner le système en insérant dans le modèle de l'image précédente les blocs pour lesquels le mouvement a déjà été obtenu, et obtenir un résultat légèrement plus fin.

5.6.1.1. Adaptation de la précision du champ de vecteur

Dans H.263, aucun contrôle n'est effectué sur la recherche des vecteurs de mouvements par l'algorithme de contrôle de débit. Dans H.264, il est possible d'influer sur le choix des

vecteurs de mouvements, comme le fait le mécanisme d'optimisation débit-distorsion. On peut donc contrôler plus ou moins la façon dont le débit est réparti entre le mouvement et les coefficients.

Cependant, le système actuel revient à aplanir le champ de mouvement de sorte qu'il soit le plus lisse possible. Plus précisément, la fonction $C(v)$ croît strictement avec l'amplitude de la différence entre le vecteur de mouvement v et sa prédiction à partir de ses voisins $\bar{v}$. Cela est logique quand la fonction de codage du vecteur de mouvement est entièrement déterminée et fixe. Elle correspond alors à une certaine répartition des probabilités d'apparition des vecteurs de mouvements.

Il se trouve que le standard H264 met en oeuvre un mode de codage arithmétique adaptatif selon des contextes. Cela signifie que la fonction de codage s'adapte aux probabilités d'apparitions effectivement constatées. On peut donc essayer de modeler les probabilités afin de pouvoir minimiser l'entropie des valeurs prises par la différence entre le vecteur de mouvement et sa prédiction tout en évitant de trop écraser le champ.

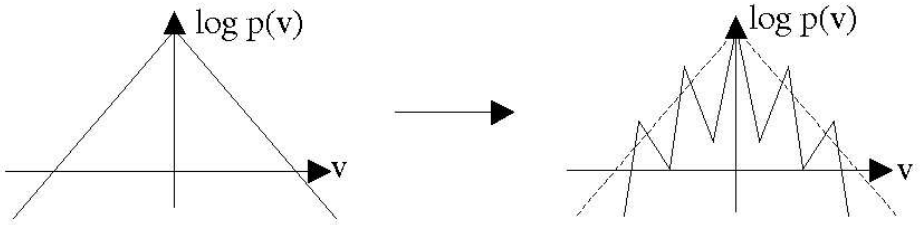


FIG. 5.5.: Modification de la courbe de probabilités d'apparition des valeurs du gradient du vecteur de mouvement pour diminuer l'entropie du champ.

L'idée est de favoriser les positions fractionnaires à dénominateur faible en comparaison de celles à dénominateur élevé, c'est-à-dire, les positions demi-entières par rapport à celles au quart de pixels, et celles au quart de pixel par rapport à celles aux huitièmes de pixel, tout en conservant la préférence pour une variation faible du champ de mouvement. La figure 5.5 montre une telle évolution. Celle-ci a toutes les chances de mener à des valeurs plus correctes du champ de mouvement, dans la mesure où ce n'est pas parce qu'on a moins de bits à utiliser pour coder l'image que le champ va, comme par miracle, évoluer plus lentement. Il faudrait bien évidemment déterminer une forme correcte de la fonction de coût, avec comme paramètre soit le budget de bits pour la trame, soit le paramètre de quantification estimé a priori pour la trame, ce qui semble plus raisonnable dans un premier temps.

Nous avons implémenté une telle approche sur un codeur JM pour H.264, en s'assurant que le flux comprimé reste décodable par un décodeur H.264 standard. Le codeur a été modifié en changeant la forme de la fonction de coût asociée au champ de mouvement, de manière à ce qu'elle suppose une distribution "en dents de scie" des vecteurs de mouvement. Cette fonction pénalise en particulier les précisions élevées dans la représentation du champ de mouvement. Le comportement de ce codeur a été testé sur la plupart des séquences d'évaluation classiques. Il ressort de ces expériences que les gains sont marginaux ou négatifs.

Une des faiblesses de l'approche est que les contextes s'initialisent au début de chaque image et que cette initialisation, dans le codage et le décodage H.264, suppose que la distribution du mouvement est laplacienne. Imposer une distribution différente crée une inefficacité à cause du temps mis par le codeur arithmétique à contexte à apprendre notre nouvelle distribution. Ici, le temps d'apprentissage est trop important par rapport au nombre de vecteurs à coder dans une trame. Il faudrait donc pouvoir débuter avec une autre distribution que la distribution laplacienne, et donc casser la compatibilité avec les décodeurs standard H.264.

5.6.2. Conclusion

Nous avons, dans ce chapitre, adapté un algorithme d'allocation de débit pour les images fixes au cas de la vidéo. Cet algorithme, bien qu'ayant une structure simple, donne des résultats meilleurs que les deux références que nous nous étions fixées.

L'évolution des standards, en fournissant de nouvelles stratégies de codage, remet néanmoins en question régulièrement les mécanismes d'allocation de débit déjà existants. Il faut donc se demander comment adapter le modèle de Mallat et Falzon dans les standards les plus récents.

Chapitre 6.

Transformation jointe des deux composantes du champ de vecteurs

Dans les chapitres 3 et 4, nous avons estimé des champs de mouvement pour lesquels nous donnons un vecteur par bloc de 2 pixels sur 2 (dans le cas de le la méthode de Bernard régularisée) ou par bloc de 4 pixels par 4 ou 8 pixels par 8 (dans le cas du modèle bilinéaire par morceau). Dans le cadre d'une utilisation au sein d'un codeur vidéo, de tels champs se doivent d'être représentés le plus efficacement possible. L'idée naturelle est de traiter chacune des composantes du champ de mouvement comme une image et de compresser celle-ci, en particulier à l'aide d'une transformation spatiale de type ondelettes ou DCT. Néanmoins, nous pouvons nous interroger sur l'information que chaque composante peut nous apporter sur l'autre pour savoir si nous ne pourrions pas exploiter cette caractéristique particulière du champ de mouvement. Dans ce chapitre, nous verrons qu'il est possible de montrer qu'une telle information existe, au moins dans certains cas, et nous proposons un nouveau schéma de *lifting* visant à l'exploiter.

6.1. Entropie et information mutuelle

6.1.1. Quelques définitions

Lorsqu'on s'intéresse à la compression de données, il est naturel de voir ce que nous dit la théorie de l'information. Celle-ci s'efforce de définir quelle est la quantité d'information contenue dans un objet (comme par exemple une variable aléatoire) ou, ce qui revient approximativement au même, quelle est la quantité d'information nécessaire pour décrire un objet donné.

Un de des concepts les plus importants est l'entropie, déja présentée rapidement en 3.1.3, qui mesure l'incertitude d'une variable aléatoire. Rappelons que l'entropie $\mathcal{H}(X)$ d'une variable aléatoire discrète est définie par :

$$\mathcal{H}(X) = -\sum_{x \in \mathcal{X}} p(x) \log p(x) \tag{6.1}$$

où $\mathcal{X}$ est l'alphabet de la variable X et p est sa distribution.

On montre que l'entropie d'une variable aléatoire est la limite inférieure de la moyenne du nombre de bits nécessaires pour coder une réalisation de la variable connaissant la distribution p. Cette limite peut, de plus, être atteinte asymptotiquement à l'aide du

codage arithmétique. Le lien est donc fort entre l'entropie et la quantité d'information que représente la connaissance de la distribution d'une variable aléatoire.

Dans le cas d'une paire de variables aléatoires, on utilise l'entropie jointe. L'entropie jointe $\mathcal{H}(X,Y)$ d'une paire de variables aléatoires discrète (X,Y) dont la distribution est $p(x,y)$ est définie par :

$$\mathcal{H}(X,Y) = -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x,y)\log p(x,y) \tag{6.2}$$

avec des conventions de notation similaires.

Il est naturel ensuite d'essayer de quantifier l'incertitude sur une variable aléatoire lorsqu'on connait une réalisation de l'autre variable aléatoire de la paire. C'est ce que mesure l'entropie conditionnelle $H(Y|X)$, qui est définie par :

$$\mathcal{H}(X,Y) = -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x,y)\log p(y|x) \tag{6.3}$$

Il faut noter qu'en général, $\mathcal{H}(X|Y) \neq \mathcal{H}(Y|X)$: si par exemple $X = \lfloor Y/5 \rfloor$, il est évident que la connaissance de Y permet de supprimer toute incertitude sur X, tandis que l'inverse n'est pas vrai.

On peut aussi essayer de définir une distance entre deux distributions, du point de vue de l'information. L'entropie relative (ou distance de Kullback-Leibler) entre deux distributions p et q est définie par :

$$D(p \parallel q) = \sum_{x\in\mathcal{X}} p(x)\log\frac{p(x)}{q(x)} \tag{6.4}$$

En ce qui concerne l'information, il est possible de représenter cette dernière comme le coût supplémentaire dû à l'utilisation d'une distribution q pour définir les symboles utilisés pour représenter une réalisation de X, plutôt que la véritable distribution p. Il faut aussi noter qu'il ne s'agit pas d'une véritable distance, puisqu'elle n'est pas symétrique et ne vérifie pas l'inégalité triangulaire.

La dernière définition à poser est celle de l'information mutuelle qui indique la réduction d'incertitude sur une des variables aléatoires connaissant la réalisation de l'autre. L'information mutuelle $I(X;Y)$ est l'entropie relative entre la distribution jointe du couple de variables aléatoires et le produit de leur distribution respective :

$$\mathcal{I}(X;Y) = \mathcal{I}(Y;X) = \sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x,y)\log\frac{p(x,y)}{p(x)p(y)} \tag{6.5}$$

On a immédiatement $\mathcal{I}(X;Y) = \mathcal{H}(X) - \mathcal{H}(X|Y) = \mathcal{H}(Y) - \mathcal{H}(Y|X) = \mathcal{H}(X) + \mathcal{H}(Y) - \mathcal{H}(X,Y)$. L'information mutuelle nous permet d'évaluer ce qu'il est possible de gagner en codant conjointement les deux variables par rapport au cas où elles sont codées séparément.

6.1.2. Information d'une composante de mouvement sur l'autre

6.1.2.1. Information mutuelle des vecteurs

Un champ de vecteurs de mouvement peut être aisément vu comme n réalisations indépendantes d'une paire de variables aléatoires $(V_1^i, V_2^i)_{0<i\leq n}$. Il est alors bien naturel d'essayer de s'intéresser à la réduction du coût de codage que nous sommes susceptibles d'obtenir en codant chacune des variables conjointement plutôt que séparément. La fraction du coût de codage qu'il est possible de gagner si nous codons simplement les valeurs du champ est de l'ordre de $g_1(V_1, V_2) = \mathcal{I}(V_1, V_2)/(\mathcal{H}(V_1) + \mathcal{H}(V_2))$.

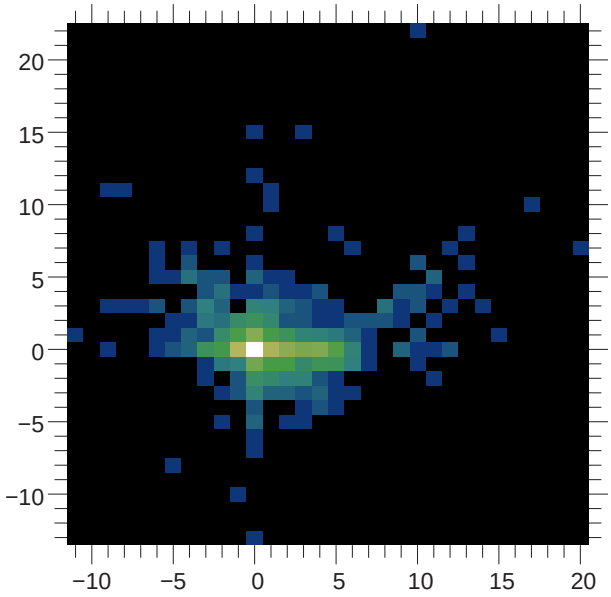
Le rapport $g_1(V_1, V_2)$ a été calculé pour des champs issus de chacune des deux méthodes d'estimation de mouvement présentées précédemment, quantifiées à différents pas. Les résultats montrent que les valeurs de $g_1(V_1, V_2)$ se situent majoritairement entre 0.05 et 0.15. Nous pourrions donc espérer gagner environ 10% du coût en bits du codage des vecteurs en codant la paire de composantes au lieu de coder chaque composante séparément, si nous codons chaque vecteur sans transformation. Ce gain potentiel s'explique entre autres par l'anisotropie du champ de mouvement, comme le montre la figure 6.1. Malheureusement, exploiter ce gain nécessite que le codeur et le décodeur connaissent la distribution des paires (v_1^i, v_2^i). Il faudrait donc transmettre une information sur cette distribution, mais il est à craindre que le surcoût de cette information soit supérieur au gain qu'on peut espérer en tirer.

Plutôt que de transmettre le profil de la distribution des données, les codeurs vidéos actuels utilisent un codage arithmétique adaptatif. Ce type de codage consiste à effectuer un codage arithmétique s'appuyant sur un profil de distribution des données se mettant à jour au fur et à mesure de l'encodage de celles-ci. Il n'est malheureusement pas garanti que l'adaptativité soit ici suffisamment efficace pour tirer parti du type de distribution des données. Mais plutôt que de chercher à exploiter l'information mutuelle entre les deux composantes du champ au moment du codage des coefficients en passant d'une codage scalaire à un codage vectoriel, il est possible d'utiliser cette information à d'autres étapes de la compression.

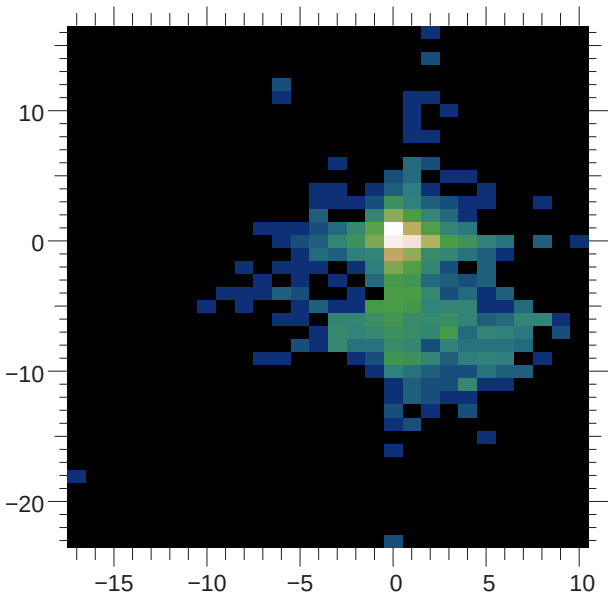
6.1.2.2. Co-occurrence des gradients de forte amplitude

Nous avons supposé que ces variables avaient la même distribution, et qu'elles étaient indépendantes. Il est pourtant logique de penser qu'il existe une dépendance entre les variables du fait de leur position géographique : deux points voisins ont plus de chances d'appartenir au même objet de la scène et donc d'avoir le même mouvement. Il y aura donc un lien entre deux vecteurs positionnés sur des points voisins. Ceci est d'autant plus vrai pour la méthode d'estimation de mouvement décrite au chapitre 3. En effet, la méthode fournissant un champ dense (un vecteur par bloc de 2 pixels sur 2), deux vecteurs associés à des blocs voisins seront plus proches que si les blocs étaient de plus grande taille, comme c'est le cas pour la méthode d'estimation décrite au chapitre 4. D'autre part, la méthode impose une contrainte de régularité sur le champ de mouvement, et donc la corrélation spatiale des vecteurs.

Ce lien est généralement exploité en utilisant une transformation spatiale (ondelettes,



(a) Co-histogramme du premier champ de mouvement de la séquence *Paris*, obtenu par la méthode d'estimation décrite au chapitre 4



(b) Co-histogramme du premier champ de mouvement de la séquence *Mobile*, obtenu par la méthode d'estimation décrite au chapitre 3

FIG. 6.1.: Co-histogramme de champs de mouvement. Les vecteurs sont arrondis au demi-pixel. L'échelle est exprimée en demi-pixels.

DCT, différence avec une prédiction médiane des vecteurs déjà codés, etc.). Cette transformation spatiale peut-elle tirer parti de l'information que chaque composante porte sur l'autre, et si oui, comment ?

Dans un modèle simple de mouvement, les objets de la scène sont en translation uniforme les uns par rapport aux autres. Les discontinuités du champ apparaissent aux frontières des objets. Dans la majorité des cas, ces discontinuités concernent les deux coordonnées du mouvement. La figure 6.2 montre pour le champ de mouvement de la séquence *Bus*, estimé par la méthode de Bernard régularisée, les zones où le gradient du champ est de forte amplitude. Ce sont approximativement les mêmes sur les composantes v_x et v_y. À l'opposé, la même étude sur certaines autres séquences ne met pas en lumière les mêmes caractéristiques. Le propos de la suite de l'étude présentée dans ce chapitre fait néanmoins la supposition que pour les vecteurs d'un champ de mouvement suffisamment dense, cette caractéristique est présente.

6.2. Schéma de *lifting* conjoint

L'hypothèse de travail sur laquelle nous nous appuyons est donc que les discontinuités du champ de mouvement ont tendance à se retrouver d'une composante sur l'autre, ce qui est le cas quand la discontinuité en question n'est ni purement verticale, ni purement horizontale. Dans le cadre de notre hypothèse, cela signifie qu'il doit être possible d'utiliser la connaissance des discontinuités d'une des composantes pour aider à la compression de l'autre composante. La connaissance de la disposition des irrégularités peut nous permettre de définir des schémas de *lifting* dont le support des filtres s'adapte de façon à ne pas les franchir.

Comme d'habitude avec le *lifting* adaptatif, le souci est d'être capable de effectuer la transformation inverse sans difficulté. Il faut aussi que les distorsions induites sur le signal soient continues par rapport à la distorsion des coefficients dans le cas où une quantification est effectuée entre l'analyse et la synthèse. Nous séparons donc le cas avec quantification, et le cas sans quantification.

6.2.1. *lifting* joint pour données entières

6.2.1.1. *lifting* joint avec interpolateurs de Deslauriers-Dubuc

Aucune quantification n'est effectuée quand le signal doit être compressé sans perte, ce qui n'a de sens que pour un signal qui peut être représenté avec des entiers[1]. Dans ces conditions, on utilise aussi des ondelettes entières (1.2.4.1). Nous avons vu que n'importe quel schéma de *lifting* pouvait être rendu entier en ajoutant des parties entières. Nous disposons, à partir des ondelettes de Cohen-Daubechies-Fauveau, d'une série complète d'ondelettes entières s'appuyant sur les interpolateurs de Deslauriers-Dubuc, à tous les ordres impairs. Nous pouvons donc essayer de choisir parmi ces différents filtres pour notre étape de *lifting* à un point donné. Comment effectuer ce choix ? Nous avons supposé que les discontinuités se trouvaient à la même position dans nos deux séries d'échantillons.

[1] ce qui inclut les signaux à précision fractionnaire fixe.

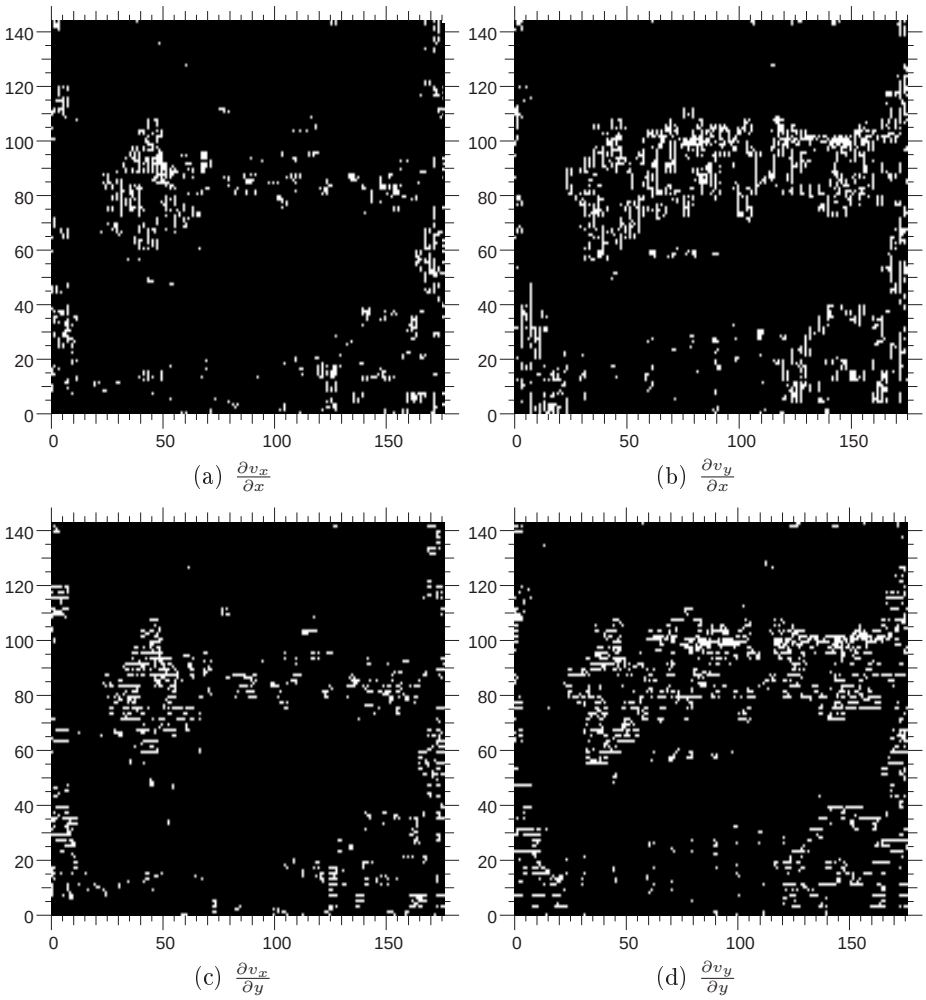


FIG. 6.2.: Comparaison des différentes composantes de $\nabla \mathbf{v}$ pour le champ de mouvement entre les trames 5 et 6 de la séquence *Bus* estimé par la méthode de Bernard régularisée. En blanc, les zones où l'amplitude de la composante est supérieure à 2 pixels.

Nous pouvons tester nos différents prédicteurs sur un de nos deux signaux, afin de choisir celui que nous allons choisir pour l'autre.

Soit deux séries d'échantillons de même taille, $(x^k_{1,n})_{0\leq n\leq N}$ et $(x^k_{2,n})_{0\leq n\leq N}$. Choisissons $(i,j)\in\{(1,2),(2,1)\}$ et calculons :

$$\tilde{x}^k_{i,n,\alpha} = |x^k_{i,2n+1} - P_\alpha((x^k_{i,2n}))|,$$

où les P_α sont les différents prédicteurs possibles. Posons :

$$\alpha_l = \operatorname*{argmin}_\alpha \tilde{x}_{i,l,\alpha}.$$

Le prédicteur pour un point $2l+1$ donné dans $(x^k_{j,n})_{0\leq n\leq N}$ est alors P_{α_l}. Comme opérateur de mise à jour, nous pourrons prendre l'opérateur de mise à jour U_{α_l} traditionnellement associé à P_{α_l}.

Dans la mesure où le choix s'effectue à partir d'autres données, il n'y a pas de problème pour ce qui est de la synthèse : en supposant que nous avons réussi à reconstituer $(x^k_{i,n})_{0\leq n\leq N}$, la reconstruction de $(x^k_{j,n})_{0\leq n\leq N}$ est immédiate. D'autre part, comme il n'y a pas de quantification, la reconstruction parfaite est assurée.

Il s'agit ensuite d'opérer une transformation en ondelettes par *lifting* pour la série d'échantillons qui nous a servi de référence. Nous pouvons utiliser le même procédé, mais il faut tenir compte du fait que la série déjà transformée, qui devient maintenant notre série de référence, dispose d'échantillons deux fois moins denses.

Nous calculons donc :

$$\tilde{x}_{j,l,\beta} = |x^{k+1}_{j,l} - P_\beta((x^{k+1}_{j,l}))|,$$

P_β opérant sur tous les voisins du point considéré. D'autre part, β parcourt l'ensemble des prédicteurs à tous les ordres, y compris les ordres pairs. Posons ensuite :

$$\gamma_l = 1 + 2\operatorname*{argmin}_\beta \tilde{x}_{j,l,\beta}.$$

Le prédicteur utilisé pour le point $2l+1$ est alors P_{γ_l} et pour l'opérateur de mise à jour, nous pouvons utiliser son pendant, U_{γ_l}.

La deuxième partie du *lifting* n'est cependant pas satisfaisante. En effet, si nous sommes au bord d'une discontinuité, il est impossible de savoir de quel côté de celle-ci le point que nous aurons à prédire effectivement se trouve. Il est donc préférable de faire une étape simple utilisant le prédicteur entier classique à un ordre moyen et l'opérateur de mise-à-jour associé..

6.2.1.2. *lifting* joint avec les prédicteurs à gauche, à droite et centré

Il est possible de travailler avec d'autres familles de prédicteurs que ceux de Deslaurier-Dubuc. Définissons P_{-1}^{lin}, P_0^{lin} et P_1^{lin} par :

$$P_{-1}^{lin}((x_{2k})) = \lfloor -\frac{1}{2}(x_{2(k-1)}) + \frac{3}{2}(x_{2k})\rfloor$$
$$P_0^{lin}((x_{2k})) = \lfloor \frac{1}{2}(x_{2k}) + \frac{1}{2}(x_{2(k+1)})\rfloor$$
$$P_1^{lin}((x_{2k})) = \lfloor \frac{3}{2}(x_{2(k+1)} - \frac{1}{2}(x_{2(k+1)})\rfloor$$

Le prédicteur centré P_0^{lin} est le meilleur des trois dans le cas général, mais en présence d'une discontinuité, l'utilisation de P_{-1}^{lin} ou de P_1^{lin} peut être préférable afin de ne pas utiliser pour la prédiction des données de part et d'autre de la discontinuité, ce qui risquerait de ne pas fournir des résultats utiles. Nous pouvons définir des poids $(w_s^t)_{s\in\{-1,0,1\},t\in\{d,b\}}$ qui serviront à privilégier un prédicteur par rapport aux autres, par exemple :

$$w_{-1}^d = w_1^d = 2 \qquad w_0^d = 1$$
$$w_{-1}^b = w_1^b = 1 \qquad w_0^b = \frac{1}{2}$$

Posons :

$$\alpha_l = \underset{\alpha\in\{-1,0,1\}}{\operatorname{argmin}} \; w_\alpha^d |x_{i,2l+1}^k - P_\alpha((x_{i,2n}^k))| + w_\alpha^b$$

La valeur de α_l détermine le prédicteur utilisé au point l pour le pas de prédiction du *lifting* appliqué à la suite $(x_{j,n})$. En mettant un poids w_0 plus faible que les autres, nous avons favorisé dans le choix de α_l le prédicteur central. Ceci permet de n'utiliser un autre prédicteur que le prédicteur centré qu'au cas où la discontinuité est nette.

Le choix de l'opérateur de mise à jour s'adapte aussi suivant la valeur des α_l. Les opérateurs de mise-à-jour à gauche (resp. centré et à droite) U_{-1} (respectivement U_0 et U_1) sont définis par :

$$U_{-1}^{lin}((x_{2k+1})) = \lfloor -\frac{1}{4}(x_{2k-3}) + \frac{3}{4}(x_{2k-1})\rfloor$$
$$U_0^{lin}((x_{2k+1})) = \lfloor \frac{1}{4}(x_{2k-1}) + \frac{1}{4}(x_{2k+1})\rfloor$$
$$U_1^{lin}((x_{2k+1})) = \lfloor \frac{3}{4}(x_{2k+1)} - \frac{1}{4}(x_{2k+3})\rfloor$$

Si $\alpha_l \leq 0$, cela signifie que nous n'avons pas trouvé de discontinuité entre les points $x_{i,2l}^k$ et $x_{i,2l+1}^k$. Le point de coordonnée $2l+1$ peut donc être utilisé sans trop de crainte pour la mise-à-jour. Dans le cas contraire, il vaudrait mieux utiliser le point de coordonnée $2l-3$. De même, si $\alpha_{l-1} \geq 0$, cela signifie que nous n'avons trouvé aucune discontinuité entre les points $x_{i,2l}^k$ et $x_{i,2l-1}^k$. Le point de coordonnée $2l-1$ peut donc être utilisé pour la mise-à-jour, et dans le cas contraire, il vaut mieux utiliser celui de coordonnée $2l+3$.

	$\alpha_l \leq 0$	$\alpha_l > 0$
$\alpha_{l-1} \geq 0$	U_0^{lin}	U_{-1}^{lin}
$\alpha_{l-1} < 0$	U_1^{lin}	U_0^{lin}

TAB. 6.1.: Choix de l'opérateur de mise-à-jour en fonction des (α_l)

Il est possible que nous obtenions $\alpha_l > 0$ et $\alpha_{l-1} < 0$, ce qui signifie que nous avons détecté une discontinuité de chaque côté du point. Dans ce cas, nous utiliserons les points de coordonnées $2l+1$ et $2l-1$. Le choix de l'opérateur de mise-à-jour est récapitulé dans le tableau 6.1.

Pour les mêmes raisons que celles évoquées en 6.2.1.1, on préfèrera une étape de *lifting* entier classique centré pour transformer la suite des $(x_{i,n}^l)$.

6.2.2. *lifting* joint sur des réels

Dans le cas où les données sur lesquelles nous souhaitons opérer sont réelles, le système décrit dans la section précédente présente des discontinuités brutales (dans le choix des opérateurs) qui rendent impossible l'utilisation du procédé décrit. En effet, une quantification des coefficients dont le pas tend vers 0 ne donnera pas alors une distorsion tendant elle aussi vers 0, ce qui n'est pas acceptable.

Si nous voulons pallier le problème, nous avons plusieurs approches possibles : soit nous modifions l'ordre de nos opérations de *lifting*, à la manière de Claypoole et Baraniuk (1.2.4.4), soit nous trouvons une facon de supprimer la discontinuité induite par le choix des opérateurs.

Si nous souhaitons pouvoir à chaque étape de choix disposer des échantillons tels qu'ils seront après quantification et reconstruction, cela signifie que nous devons abandonner l'adaptativité sur une des deux composantes du vecteur de mouvement : nous pouvons par exemple transformer la composante (x_1) sans adaptativité, la quantifier puis la reconstruire, puis ensuite effectuer tous les choix à partir de ces coefficients.

L'autre solution envisageable est de passer continûment d'un prédicteur à l'autre. Une première remarque est de constater que tout barycentre de prédicteurs est lui aussi un prédicteur[2], même s'il ne disposera pas nécessairement des qualités d'approximation de ceux-ci. Nous pouvons donc tenter d'attribuer des poids à chacun des prédicteurs P_α au point n dépendant des erreurs $\tilde{x}_{i,\alpha,n}$. Plus une erreur est faible par rapport aux autres, plus son poids doit être important dans le calcul du barycentre. Si nous prenons des poids ρ_i tels que :

$$\rho_i = \frac{\sum_{\alpha' \neq \alpha} (\tilde{x}_{i,\alpha',n} + \varepsilon_{\alpha'})}{(A-1) \sum_{\alpha'} (\tilde{x}_{i,\alpha',n} + \varepsilon_{\alpha'})},$$

où A est le nombre de prédicteurs dans l'ensemble parmi lesquels nous les choisissons, et où les ε_α sont fixés pour définir les poids lorsque toutes les erreurs sont nulles et assurer

[2] En terme de fonction, cela tombe sous le sens, mais cette affirmation est à prendre en terme d'objectif du prédicteur : fonction qui espère prédire les valeurs d'une fonction d'une certaine classe aussi justement que possible.

ainsi une certaine stabilité du choix autour de l'origine. Néanmoins, une telle solution présente l'inconvénient suivant :

- Si les ε_α sont petits, la transformation sera instable lorsque les erreurs prédites sur la composante de référence seront faibles.
- À l'opposé, si les ε_α sont plus importants, ils pousseront la combinaison des prédicteurs vers un prédicteur particulier barycentre des prédicteurs que l'on teste.Or un tel prédicteur est vraisemblablement moins bon que le prédicteur utilisé en général pour le cas réel simple. Nous perdrons alors en efficacité.

6.2.3. Asymétries du *lifting* joint

Nous avons, jusqu'à présent, passé sous silence un des problèmes du *lifting* joint : l'asymétrie dans le traitement des données. En effet, quelle que soit la variante choisie, les deux composantes vectorielles ont des rôles différents. Dans le cas où l'adaptativité ne porte que sur l'une des deux composantes, cela est évident, mais cela est vrai aussi lorsque les deux composantes bénéficient de l'adaptativité. En effet, comme nous l'avons expliqué, dans un cas, le critère s'appuie sur une comparaison avec l'effet des filtres sur l'autre composante, celle-ci étant de même taille, tandis que, dans l'autre cas, la comparaison s'effectue à partir de filtres de longueur plus réduite, s'appliquant à un signal de taille moitié. Il y a donc aussi une asymétrie entre les composantes.

D'autre part, le schéma de *lifting* 1D sur lequel nous nous appuyons est non-linéaire. Cela signifie que l'ordre de filtrage n'est pas neutre quand nous passons au filtrage 2D. Du fait de la non-linéarité, filtrer verticalement puis horizontalement n'est pas équivalent à effectuer ces mêmes opérations dans l'ordre inverse. Il nous faut donc déterminer quelle est l'ampleur de la différence et s'il existe un ordre plus efficace que l'autre pour effectuer le filtrage.

6.3. Résultats

Nous avons utilisé le schéma de *lifting* décrit en 6.2.1.2 pour transformer d'une part un champ de mouvement synthétique pour valider le principe de notre schéma, et d'autre part des champs de mouvement obtenus par les méthodes décrites dans les chapitres 3 et 4. Ces derniers sont préalablement arrondis à une précision donnée, afin d'être représentés par des entiers, puis transformés par *lifting*. La façon de procéder est ici différente de celle utilisée pour les tests réalisés sur des coefficients d'image classique ou d'image d'erreur. En effet, d'une part, il est encore peu courant de trouver des systèmes de compression utilisant des champs réels compressés avec distorsion. D'autre part, nous avons vu qu'adapter l'idée du *lifting* joint sur des données réelles semble difficile. Nous avons effectué la comparaison par rapport aux ondelettes 5/3 entières, sur les 20 premières images de la séquence.

6.3.1. Critère d'évaluation

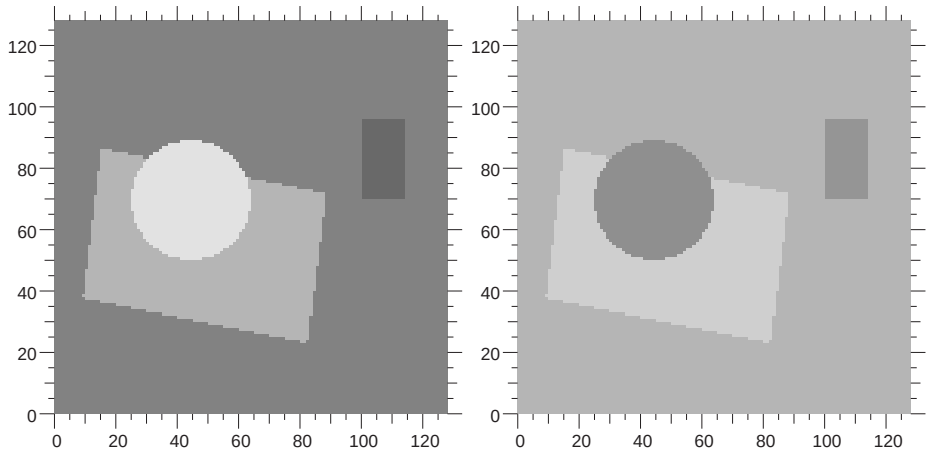
Le coût de codage a été estimé de la façon suivante : pour chaque sous-bande s de la décomposition en ondelettes, nous avons calculé l'entropie $\mathcal{H}_s$ des n_s coefficients de cette sous-bande. Le coût de codage R estimé est alors :

$$R = \sum_s n_s \mathcal{H}_s.$$

Nous comparons alors les coûts de codage entre eux, sachant qu'il n'y a jamais de distorsion introduite dans le champ : en effet, comme nous l'avons mentionné, la transformation est une transformation entière, et nous n'avons pas à quantifier les données après la transformation en ondelettes. Plus précisément, nous notons le pourcentage de gain ou de perte, entre l'utilisation du *lifting* entier joint et celle du *lifting* entier classique.

6.3.2. Test sur un champ synthétique

Le champ de la figure 6.3 représente le mouvement d'objets de forme simple en translation parrallèlement au plan de l'image. Nous avons utilisé un schéma de *lifting* joint utilisant le changement de filtre sur la composante v_x du champ.



(a) v_x (b) v_y

FIG. 6.3.: Composantes du champ de mouvement synthétique

Le coût de codage estimé de cette composante après transformation est de 1.31 bits par vecteur du champ, tandis qu'elle est de 1.42 bits par vecteur dans le cas de l'utilisation d'un schéma de *lifting* standard. Nous avons donc un gain de 7,7% pour le codage de cette composante, et de 4,2% si nous considérons le coût de codage des deux composantes du champ de mouvement. Ces résultats sont satisfaisants : ils montrent la validité de notre raisonnement.

6.3.3. Tests sur les données réelles

6.3.3.1. Composante de référence et première direction de filtrage

En effectuant le *lifting* joint, nous avons deux paramètres à prendre en compte : la composante qui sert de référence pour déterminer le choix des filtres sur l'autre composante, d'une part, et d'autre part la première direction de filtrage. En effet, notre traitement n'étant pas linéaire, filtrer horizontalement puis verticalement n'est pas équivalent à enchaîner filtrage vertical puis filtrage horizontal. En pratique, comme le montrent les résultats donnés dans le tableau 6.2, il est difficile de faire un choix a priori sur les meilleurs paramètres à utiliser, les comparaisons ne donnant pas des résultats cohérents. Pour les résultats suivants, nous donnerons la moyenne calculée sur les 4 combinaisons possibles de ces deux paramètres.

Séquence	Format	Méthode d'estimation	Composante de référence	Première direction	Précision 1/4 pixel	Précision 1/2 pixel
Paris	CIF	BBL	v_y	X	-1.1	-0.8
			v_y	Y	-1.0	-0.6
			v_x	X	-1.5	-0.2
			v_x	Y	-1.7	-0.6
News	QCIF	BR	v_y	X	-0.7	0.7
			v_y	Y	-0.5	0.2
			v_x	X	-0.6	-0.1
			v_x	Y	-1.0	0.0

TAB. 6.2.: Comparaison du choix des composantes et des directions privilégiées. BR : méthode de Bernard régularisée. BBL : modèle d'estimation d'un champ bilinéaire par morceau

6.3.3.2. Résultats d'ensemble

Pour chaque séquence, nous avons noté pour les 20 premiers champs de mouvement, le meilleur pourcentage de gain de codage, le plus faible, et le pourcentage moyen. L'ensemble des résultats se trouve dans le tableau 6.3.

L'objectif que nous nous proposions d'atteindre avec le *lifting* joint n'est clairement pas atteint : en moyenne, l'utilisation de ce nouveau schéma est contre-productif sur toutes les séquences, et aux deux précisions du champ testées, sauf dans un cas. Une des raisons que nous pouvons avancer est que les champs utilisés ne correspondent pas dans leur majorité à notre hypothèse de travail, c'est-à-dire à des champs où les gradients des deux composantes du mouvement sont d'amplitude élevée aux mêmes positions. Dans certains cas, nous pouvons néanmoins noter une légère amélioration du coût de codage, par exemple pour la séquence *News*. Celle-ci présente l'avantage d'être assez nette, et d'avoir des objets bien délimités : présentateurs, vidéo dans une zone du fond. En revanche, dans le cas d'une séquence complexe, comme *Tempete*, où des feuilles volent, les

Séquence	Format	Méthode d'estimation	Précision : 1/4 pixel			Précision : 1/2 pixel		
			Max.	Moy.	Min.	Max.	Moy.	Min.
Paris	CIF	BBL	+0.2	-1.3	-4.7	+1.0	-0.6	-3.3
Tempete	CIF	BBL	-1.4	-4.5	-9.2	+0.2	-3.1	-6.3
Mobile	CIF	BBL	+0.5	-0.6	-2.7	+0.4	-0.3	-2.3
Bus	CIF	BBL	-1.7	-3.8	-6.0	-0.8	-2.4	-4.8
Paris	CIF	BR	+1.0	-1.1	-3.8	+1.8	-0.5	-2.4
Tempete	CIF	BR	-0.7	-3.6	-7.4	-0.2	-2.4	-5.6
Mobile	CIF	BR	+0.3	-0.7	-3.6	+0.2	-0.3	-2.0
Bus	CIF	BR	-1.7	-3.6	-6.6	-0.8	-2.5	-4.7
News	QCIF	BBL	+1.0	-0.3	-4.7	+0.8	-0.1	-1.7
Foreman	QCIF	BBL	+1.1	-0.7	-2.5	+1.3	0.0	-1.3
Silent	QCIF	BBL	+1.8	-2.5	-8.0	+4.0	-1.5	-6.9
News	QCIF	BR	+0.8	-0.7	-3.2	+2.7	+0.2	-1.2
Foreman	QCIF	BR	+0.3	-0.8	-2.4	+0.6	-0.2	-1.6
Silent	QCIF	BR	+0.5	-2.3	-6.3	+2.4	-1.3	-4.8

TAB. 6.3.: Gain de codage pour l'utilisation du *lifting* joint dérivé des ondelettes 5/3 entières. Le gain est exprimé en pourcentage par rapport aux ondelettes 5/3 entières. Min.(resp. Moy., Max.) : gain minimal (resp. moyen, maximal) sur le codage d'un trame. BR : méthode de Bernard régularisée. BBL : méthode d'estimation d'un champ bilinéaire par morceau.

méthodes d'estimations auront du mal à fournir un champ de mouvement où l'on distingue clairement les objets en déplacement. C'est sur cette séquence que nous obtenons les plus mauvais résultats.

L'autre hypothèse que nous pouvons avancer est que le choix des poids dans le critère de sélection du prédicteur n'est pas adéquat. Il est possible qu'en favorisant plus le prédicteur central par rapport aux prédicteurs latéraux, nous améliorions légèrement les résultats. Néanmoins, trop favoriser le prédicteur central aura, à l'inverse, tendance à diminuer la différence entre le schéma de *lifting* joint et le schéma classique. Dans ces conditions, la complexité supplémentaire introduite par ce schéma ne sera-t-elle rédhibitoire par rapport aux gains potentiels ?

6.4. Conclusion

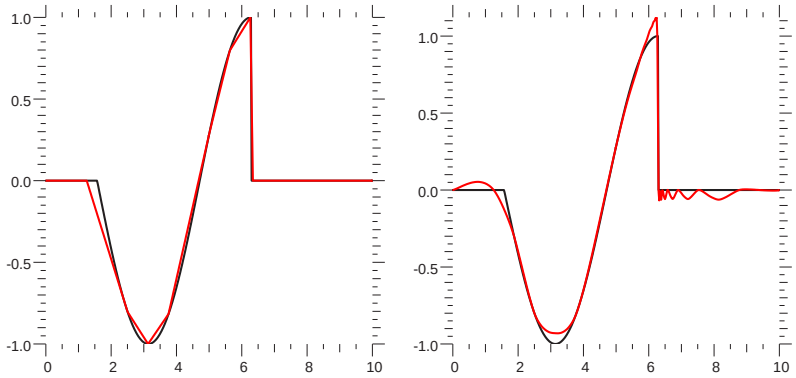
Dans ce chapitre, nous avons essayé de mettre en place un schéma de *lifting* qui fait appel à la connaissance de données associées aux données que nous souhaitons transformer. Dans le cas de champs de mouvement synthétiques, ce nouveau schéma apporte un gain probant. Malheureusement, les hypothèses sur la forme de l'information que chaque jeu de données fournit sur l'autre ne sont pas vérifiées pour nos méthodes d'estimation de mouvement. Les résultats sur ces données n'apportent alors que rarement des gains. De plus amples tests avec d'autres méthodes d'estimation de mouvement seraient nécessaires, d'une part, pour déterminer si notre hypothèse de travail peut se vérifier pour d'autres méthodes de mesure du mouvement, auquel cas le schéma proposé pourra être plus efficace.

Chapitre 7.

Ondelettes sans rebonds

7.1. Motivation

Dans la section 1.2.4, nous avons vu des tentatives de développement d'ondelettes non-linéaires. En particulier, Claypoole et Baraniuk essayent d'adapter l'ordre des ondelettes à la régularité locale du signal. En effet, les ondelettes d'ordre élevé sont intéressantes sur les zones régulières, mais engendrent des oscillations (effet de Gibbs) dans les zones contenant des discontinuités (figure 7.1(b)). À l'opposé, les ondelettes d'ordre faible, si elles ne réussissent pas à exploiter toute la structure des zones fortement régulières, ne sont pas la cause d'oscillations lors de la représentation de discontinuités (figure 7.1(a)).



(a) Ondelette d'ordre 2 (b) Ondelette d'ordre 4

FIG. 7.1.: Interpolation linéaire et interpolation de Deslaurier-Dubuc d'ordre 4.

Malheureusement, la solution proposée présente plusieurs défauts :

- Le passage d'un filtre à l'autre est quelque chose d'essentiellement discret. Cela signifie qu'il existe une véritable discontinuité dans le prédicteur. Si aucun dispositif particulier n'est mis en place, cela signifie que l'erreur entre le signal d'origine et le signal obtenu après perte de compression et décodage peut ne pas tendre vers 0 quand le pas de quantification tend vers 0.
- La synchronisation utilisée pour pallier le problème précédent limite les utilisations possibles du schéma. En effet, le pas de quantification doit être choisi avant de connaître la distribution des coefficients. Dans ces conditions, une compression à débit fixé sera plus difficile à réaliser. Cela exclut aussi ce schéma des systèmes

progressifs pour lesquels il n'est pas sûr que le décodeur reçoive les coefficients exacts.

Nous voulons donc définir un schéma de *lifting* pour lequel :

- le prédicteur tire partie de la régularité du signal ;
- le prédicteur n'engendre pas des rebonds indésirables lors de la reconstruction ;
- le prédicteur est continu par rapport aux données.

7.2. Prédiction encadrée

Pour le début de notre étude, nous nous plaçons dans le cas 1D, et nous nous intéressons essentiellement au prédicteur, en construisant un schéma de *lifting* ne disposant que d'un opérateur de prédiction (et donc d'aucune étape de mise à jour).

7.2.1. Prédicteur de Deslauriers-Dubuc

Les prédicteurs que nous utilisons sont ceux des ondelettes d'interpolation de Deslauriers-Dubuc. Ils consistent à faire une interpolation lagrangienne glissante des échantillons impairs x_{2k+1} à partir des échantillons pairs x_{2k}. Afin de prédire x_{2k+1}, un polynôme de Lagrange P de degré $2p-1$ est calé sur les $2p$ échantillons pairs les plus proches :

$$x_{2k-2p+2}, \ldots, x_{2k}, x_{2k+2}, \ldots, x_{2k+2p}$$

La valeur prédite en $2k+1$ est alors :

$$\widehat{x}_{2k+1} = P(2k+1).$$

La figure 7.2 illustre ce type d'interpolation. Sur le graphe représenté et dans toute cette section, un échantillon se situe au point M_n de coordonnées (n, x_n). De même, un échantillon interpolé se trouve au point $\widehat{M}_n$ de coordonnées $(n, \widehat{x}_n)$.

L'étape de prédiction consiste donc à effectuer le calcul :

$$h_k = x_{2k+1} - \widehat{x}_{2k+1}$$

et donc à remplacer les échantillons impairs par les erreurs de prédiction.

Plus précisément, dans la suite, nous considérerons le cas des ondelettes d'interpolation de Deslauriers-Dubuc d'ordre 4.

7.2.2. Encadrement d'ordre 0

Puisque nous cherchons à éliminer les rebonds, il nous faut d'abord les définir et les localiser. Dans une première et grossière approximation, appelons rebond tout endroit où les valeurs interpolées ne sont pas comprises dans l'intervalle défini par les plus proches voisins $2k$ et $2(k+1)$, $[x_k \wedge x_{2(k+1)}, x_{2k} \vee x_{2(k+1)}]$. Si nous voulons répondre au problème

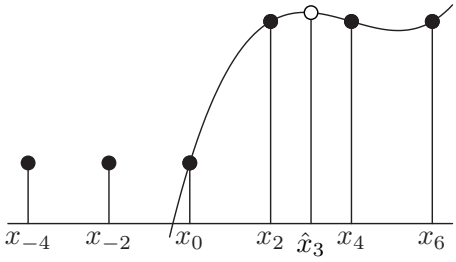


FIG. 7.2.: Interpolation de Deslauriers-Dubuc d'ordre 4

avec cette définition, nous pouvons simplement, partant d'un interpolateur d'ordre n, P_n, l'empêcher de donner des valeurs sortant de l'intervalle admissible, en reprojetant le résultat de l'interpolation dans l'intervalle. On définit alors $P_{n,0}$ par :

$$P_{n,0}(x) = \begin{cases} x_{2k} \wedge x_{2(k+1)} & \text{si } P_n(x) \leq x_{2k} \wedge x_{2(k+1)} \\ x_{2k} \vee x_{2(k+1)} & \text{si } P_n(x) \geq x_{2k} \vee x_{2(k+1)} \\ P_n(x) & \text{sinon,} \end{cases}$$

tout ceci étant valable sur l'intervalle $[2k, 2(k+1)]$.

Nous pouvons donner une définition de $P_{n,0}$ sous une forme plus compacte :

$$\begin{aligned} P_{n,0}(x) &= ((x_{2k} \wedge x_{2(k+1)}) \vee P_n(x)) \wedge (x_{2k} \vee x_{2(k+1)}) \\ &= \text{med}(x_{2k}, P_n(x), x_{2(k+1)}) \end{aligned}$$

L'effet de $P_{n,0}$ peut se voir sur la figure 7.3.

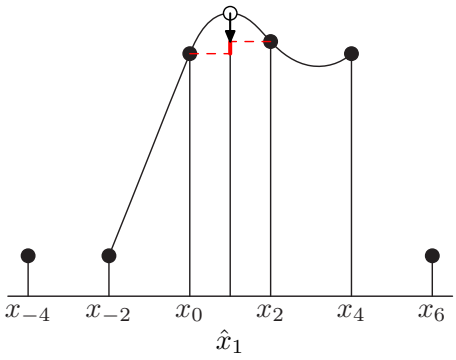


FIG. 7.3.: Exemple de reprojection : $P_{3,0}$

Le fait est que, comme les opérations $\wedge$ et $\vee$ sont continues, $P_{n,0}$ est lui aussi continu. Ce prédicteur élimine les rebonds au sens que nous avons donné et, en l'absence de rebond, il se comporte comme le prédicteur sous-jacent. $P_{n,0}$ correspond donc bien aux critères que nous nous étions fixés.

Il est aussi possible de montrer que cette prédiction permet de conserver la monotonie du signal : si

$$x_{2k}, x_{2k+2}, \ldots, x_{2l}$$

est monotone, alors la séquence interpolée

$$x_{2k}, \widehat{x}_{2k+1}, x_{2k+2}, \ldots, x_{2l}$$

l'est aussi.

7.2.3. Encadrement d'ordre 1

Notre première définition d'un rebond est bien évidemment trop large. Par exemple, elle n'est pas stable lors de l'addition d'une composante linéaire au signal. Néanmoins, nous avons pu dégager une solution adéquate à partir de cette définition : l'encadrement du polynôme d'interpolation par des polynômes d'ordre 0 calés sur les voisins du point recherché.

Partant de cette idée, il est naturel de l'étendre à des polynômes d'ordre supérieur et de rechercher l'invariance par ajout d'une composante polynômiale de même ordre.

Il est possible de définir un schéma simple pour la prédiction encadrée par des polynômes d'ordre 1, représentés par des droites sur un graphe de fonctions. Un échantillon interpolé $\widehat{x}_1$ doit être tel que le point $(1, \widehat{x}_1)$ se situe entre les lignes Δ_1 et Δ_2, où Δ_1 est la ligne parallèle à $(M_{-2}M_2)$ passant par M_0 et où Δ_2 est la droite (M_0, M_2). Nous imposons une contrainte similaire définie symétriquement à l'aide des points M_4, M_2 et M_0 en définissant les droites Δ'_1 et Δ'_2 (figure 7.4).

Plus formellement, notons $P_{k,lm}$ le polynôme d'ordre 1, vérifiant :

$$P_{k;l,m}(2k) = x_{2k} \tag{7.1a}$$

$$P_{k;l,m}(2m) - P_{k;l,m}(2l) = x_{2m} - x_{2l} \tag{7.1b}$$

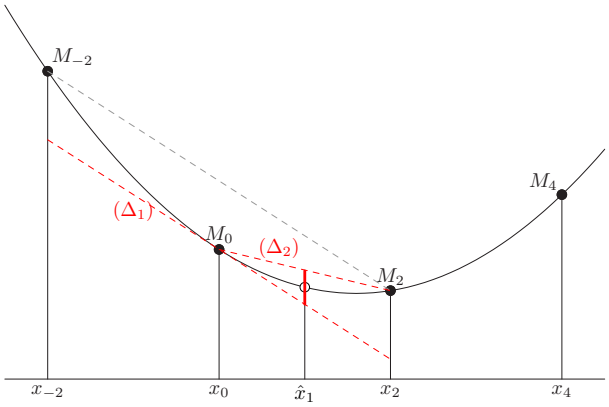
Notre nouveau prédicteur au point $2k+1$ peut s'écrire comme :

$$\begin{aligned} P_{n,1}(x) = &\Big(\big(P_{k;k-1,k+1}(x) \vee P_{k;k,k+1}(x)\big) \wedge \big(P_{k+1;k,k+2}(x) \vee P_{k+1;k,k+1}(x)\big)\Big) \\ &\wedge P_n(x) \vee \\ &\Big(\big(P_{k;k-1,k+1}(x) \wedge P_{k;k,k+1}(x)\big) \vee \big(P_{k+1;k,k+2}(x) \wedge P_{k+1;k,k+1}(x)\big)\Big) \\ = &\operatorname{med}(P_{k;k-1,k+1}(x), \operatorname{med}(P_{k+1;k,k+2}(x), P_n(x), P_{k+1;k,k+1}(x)), P_{k+1;k,k+1}(x)) \end{aligned}$$

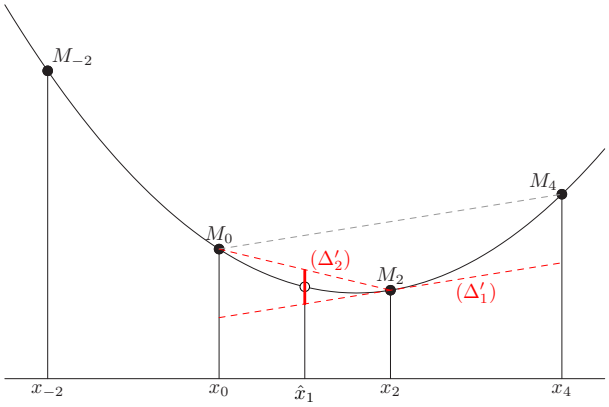
7.2.4. Analyse du comportement des ondelettes sans rebonds

Dans cette section, nous allons essayer d'analyser le comportement des ondelettes sans rebonds d'interpolation, c'est-à-dire construites avec un schéma de lifting avec une étape de prédiction seule et sans étape de mise-à-jour). Nous les analysons dans deux variantes : avec un schéma d'encadrement d'ordre 0, et un schéma d'encadrement d'ordre 1.

Nous supposerons que les fonctions approchées sont régulières par morceaux, et ce sur un nombre fini de morceaux. Nous devrons faire une deuxième hypothèse qui n'est



(a) Contrainte d'encadrement linéaire à gauche



(b) Contrainte d'encadrement linéaire à droite

FIG. 7.4.: Exemple de reprojection : $P_{3,1}$

pas une hypothèse de régularité classique, et qui s'appuiera sur la définition de "fonction k-simple".

Définissons tout d'abord ce que nous appelons un point de changement de signe. Si f une fonction définie d'un intervalle $I \subset \mathbf{R}$ dans $\mathbf{R}$, alors un point $x \in I$ est un point de changement de signe de f si :

$$\forall \varepsilon > 0, \exists x', x'' \in I, \quad |x - x'| < \varepsilon, |x - x''| < \varepsilon \text{ et } f(x') < 0, f(x'') > 0 \tag{7.2}$$

Une fonction f définie d'un intervalle $I \subset \mathbf{R}$ dans $\mathbf{R}$ est alors dite *simple* si le nombre de ses points de changement de signe est fini. Cette définition s'étend alors naturellement : une fonction f définie d'un intervalle $I \subset \mathbf{R}$ dans $\mathbf{R}$ est dite *k-simple* si elle est dérivable $k-1$ fois et que pour $i \in \{0, \ldots, k-1\}$, $f^{(i)}$ est simple.

L'ensemble des points de changement de signe d'une fonction est d'intérieur vide, par définition même de ces points. Cependant, ils ne sont pas nécessairement en nombre fini. Par exemple, la fonction f définie par :

$$f(x) = \begin{cases} 0 & \text{si } x = 0 \\ x^9 \sin \frac{1}{x} & \text{sinon} \end{cases}$$

est 4 fois dérivable de dérivée $4^{\text{ème}}$ continue, mais elle possède un nombre infini de changement de signe, de même que ses quatre premières dérivées. Elle n'est donc pas simple ni encore moins 4-simple.

Pour la suite, nous ferons donc l'hypothèse que la fonction approchée est 1-simple dans le cas d'un schéma d'encadrement d'ordre 0 et 2-simple dans le cas d'un schéma d'encadrement d'ordre 1.

Dans les deux cas, le comportement de ces ondelettes de représentation non-linéaires a les propriétés suivantes :

- près d'une discontinuité, l'approximation d'une fonction par une décomposition en ondelettes sans rebonds tronquée ne fait pas intervenir de rebonds. On ne peut pas parler de convergence uniforme (puisque la fonction vers laquelle l'approximation converge n'est pas continue). En revanche, le graphe de la fonction ne présente pas de rebonds et tend à conserver la discontinuité.
- lorsque nous ne sommes pas à proximité d'une discontinuité ou d'un point de changement de signe de la dérivée $k^{\text{ème}}$ de notre fonction k-simple, cette dérivée est de signe constant. Dans le cas de l'encadrement de degré 0, cela signifie que la fonction est monotone (croissante ou décroissante) et que donc l'opérateur de reprojection est sans effet. De même, dans le cas d'un encadrement de degré 1, la fonction est soit concave soit convexe, et on peut alors montrer que l'opérateur de reprojection $P_{3,1}$ est également sans effet à des échelles suffisamment fines. Dans les deux cas, si la fonction à approcher est au moins $\mathcal{C}^4$, alors l'erreur d'approximation est celle des ondelettes de Deslauriers-Dubuc, donc en $O(2^{-4j})$, si j est l'échelle des ondelettes.
- lorsque nous sommes à proximité d'un point de changement de signe de la dérivée $k^{\text{ème}}$ de notre fonction k-simple, nous introduisons une erreur d'approximation d'ordre moins favorable dépendant de k. L'erreur effectuée est en $O(2^{-jk})$.

Pour résumer : à proximité d'une singularité, le schéma des ondelettes sans rebonds ne fait pas intervenir de rebonds. Pour le reste des positions, l'erreur d'approximation est similaire à celle des ondelettes de Deslauriers-Dubuc, sauf en un nombre fini de zones autour des points de changement de signe de la dérivée $k^{\text{ème}}$ de la fonction approchée.

7.2.5. Formulation par médian d'intervalles

Chacune des contraintes que nous avons posées sur notre prédicteur peut être vue comme la projection p_I de la valeur interpolée sur un intervalle $I = [a, b]$:

$$p_I(x) = \begin{cases} a & \text{si } x < a \\ x & \text{si } a \leq x \leq b \\ b & \text{sinon,} \end{cases}$$

Lorsque nous avons plusieurs contraintes, nous projetons donc plusieurs fois sur des intervalles différents, en espérant trouver *in fine* un encadrement "central". Considérons plus précisément cet objectif en considérant deux intervalles quelconques $I_1 = [a_1, b_1]$ et $I_2 = [a_2, b_2]$. Si $I_1 \cap I_2 \neq \emptyset$, alors notre encadrement central sera clairement $I_1 \cap I_2$, et dans ce cas,

$$I_1 \cap I_2 \neq \emptyset \Longrightarrow p_{I_1} \circ p_{I_2} = p_{I_2} \circ p_{I_1} = p_{I_1 \cap I_2}.$$

Mais que se passe-t-il si $I_1 \cap I_2 = \emptyset$? Dans ce cas précis, effectuer un encadrement par I_1 puis un encadrement par I_2 nous donne une réponse différente de la suite d'encadrements effectués dans l'ordre inverse :

$$I_1 \cap I_2 = \emptyset \Longrightarrow p_{I_1} \circ p_{I_2} \neq p_{I_2} \circ p_{I_1}$$

Ceci est problématique si l'on considère que nos contraintes sont d'importances équivalentes. De plus, nous avons, dans un cas comme dans l'autre, un résultat qui est fortement excentré par rapport à l'ensemble des deux intervalles.

Si nous voulons pallier ces inconvénients, nous avons intérêt à rechercher notre valeur contrainte, non plus dans I_1 ou I_2, mais dans l'intervalle qui se situe entre ces deux intervalles. Définissons ceci de manière plus formelle. Le filtre R_k de rang k associe à une liste non-ordonnée sa k-ième plus petite valeur. L'intervalle que nous recherchons, noté $\text{intmed}(I_1, I_2)$, est alors :

$$\text{intmed}(I_1, I_2) = [R_1(a_1, b_1, a_2, b_2), R_2(a_1, b_1, a_2, b_2)].$$

Synthétiser les deux contraintes d'encadrement par I_1 et I_2 par la contrainte d'encadrement par $\text{intmed}(I_1, I_2)$ peut paraître surprenant. En effet, dans le cas où les deux intervalles sont disjoints, aucune des deux contraintes initiales n'est finalement respectée, et nous avons effectivement :

$$\text{intmed}(I_1, I_2) \cap I_1 = \emptyset$$
$$\text{intmed}(I_1, I_2) \cap I_2 = \emptyset$$

Néanmoins, puisque, dans ce cas, il est impossible de satisfaire aux deux contraintes simultanément, cet encadrement permet d'avoir une contrainte qui se présente comme un intermédiaire entre les deux contraintes initiales.

Nous pouvons par ailleurs étendre directement notre définition au médian de k intervalles, par la formule suivante :

$$\text{intmed}(([a_j, b_j])_{1 \leq j \leq k}) = [R_k(a_1, \ldots, a_k, b_1, \ldots, b_k), R_{k+1}(a_1, \ldots, a_k, b_1, \ldots, b_k)].$$

D'autre part, notre définition peut s'étendre aussi simplement aux intervalles de la forme $]-\infty, b]$ ou $[a, +\infty[$, en prenant simplement soin d'ouvrir la borne de l'intervalle si celle-ci est un infini.

Enfin, il est clair que le médian d'intervalles est continu par rapport à chaque intervalle, puisque ses bornes sont calculées par des fonctions continues des bornes des intervalles.

Notons $I_{k;d}(x)$ l'intervalle de bornes $P_{k;k-d,k+d}(x)$ et $P_{k;k,k+d}(x)$. Nous pouvons alors réécrire $P_{n,1}(x)$ pour le point $2k+1$ comme :

$$P_{n,1}(x) = p_{\text{intmed}(I_{k,1}(x), I_{k+1,-1}(x))}(P_n(x))$$

7.2.6. Résultats

7.2.6.1. Signaux synthétiques

Nous avons testé notre schéma de prédiction non-linéaire dans le cas d'une approximation à N termes de signaux synthétiques simples.

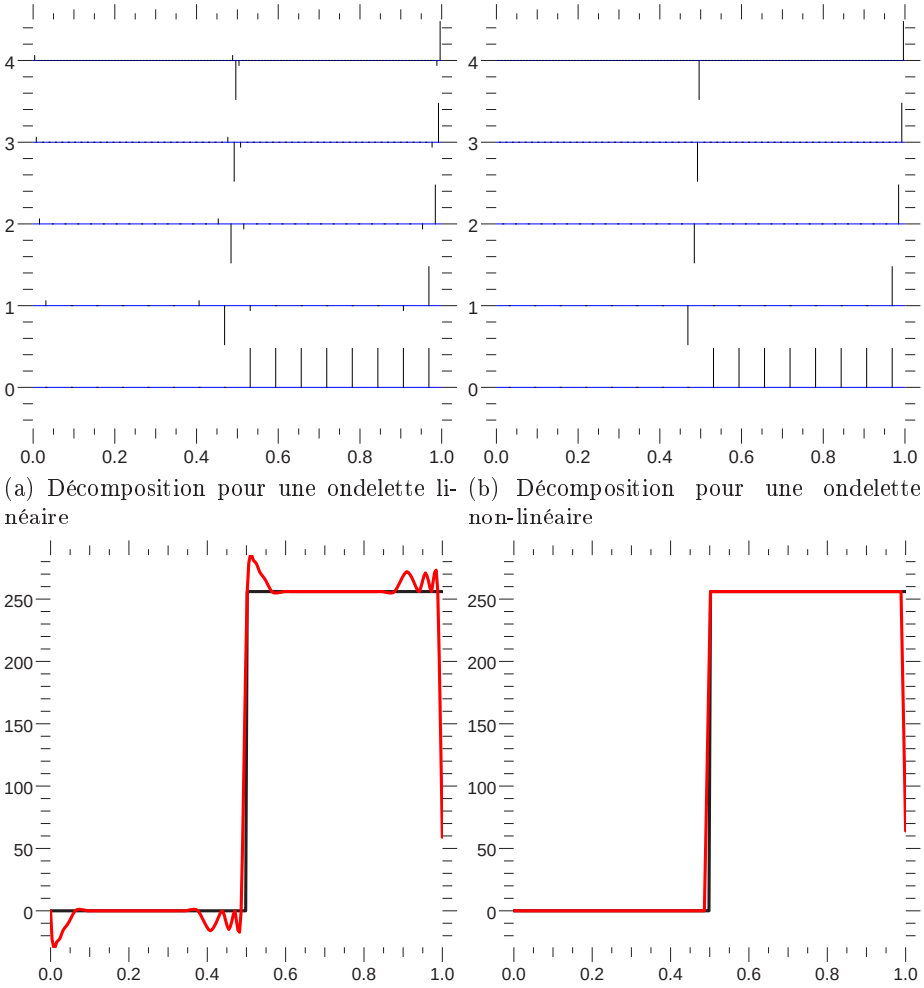
Dans le cas d'un simple créneau (figure 7.5), nous pouvons voir que notre schéma non-linéaire n'engendre pas de petits coefficients autour des coefficients importants, contrairement au cas linéaire. Lors de l'approximation à N termes, la quantification à zéro de ces petits coefficients dans le cas linéaire provoque des rebonds. Notre nouveau schéma, lui, n'est pas pénalisé.

De la même façon, notre schéma se comporte selon nos attentes dans le cas d'un signal continu par morceau. L'absence de petits coefficients qui se situent tout autour des discontinuités avantage nettement le prédicteur non-linéaire dans le cadre de l'approximation à N termes. En effet, la distorsion des coefficients est alors globalement plus faible, puisqu'il y a moins de coefficients non-nuls.

Nous introduisons ensuite une étape de mise-à-jour pour laquelle nous opérons ce même système d'encadrement. Là encore, la capacité à ne pas créer trop de coefficients significatifs autour des singularités du signal permet à notre schéma d'être nettement plus performant que le schéma linéaire équivalent, que ce soit pour le créneau (figure 7.7) ou pour le signal continu par morceau (figure 7.8).

7.2.6.2. Signaux réels

Nous avons testé notre schéma complet (prédiction et mise-à-jour) sur des bandes de l'image Lena. Nous obtenons une qualité d'approximation à N termes correspondant à la courbe de la figure 7.9.



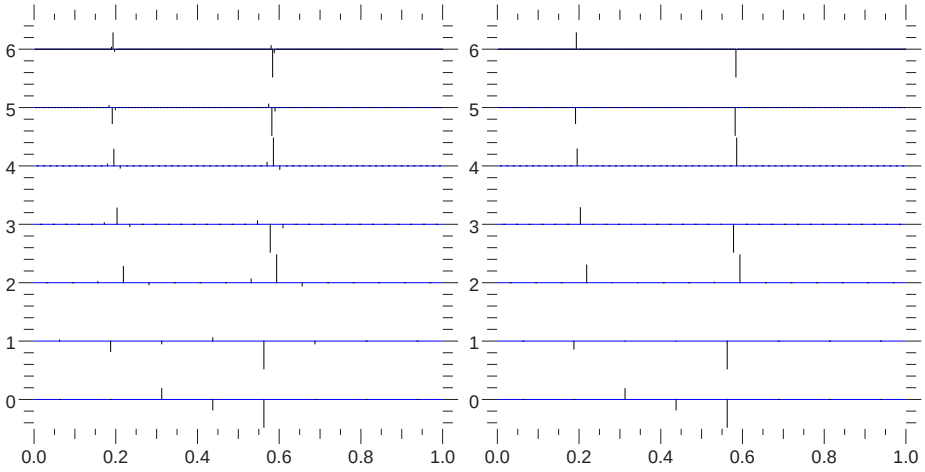
(a) Décomposition pour une ondelette linéaire

(b) Décomposition pour une ondelette non-linéaire

(c) Approximation à N termes pour une ondelette linéaire

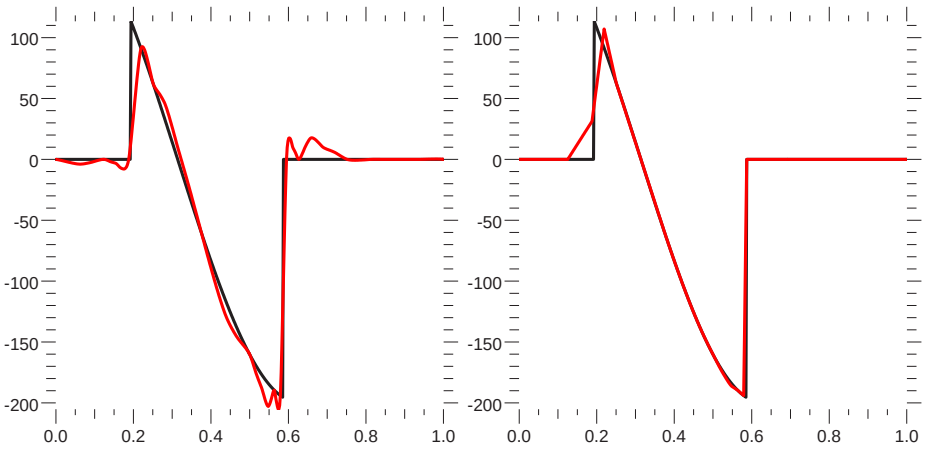
(d) Approximation à N termes pour une ondelette non-linéaire

FIG. 7.5.: Test des ondelettes sans rebonds sur un créneau. Nous n'utilisons ici que les opérations de prédiction. Sur les courbes : en noir, le signal original ; en rouge, le signal approximé.



(a) Décomposition pour une ondelette linéaire

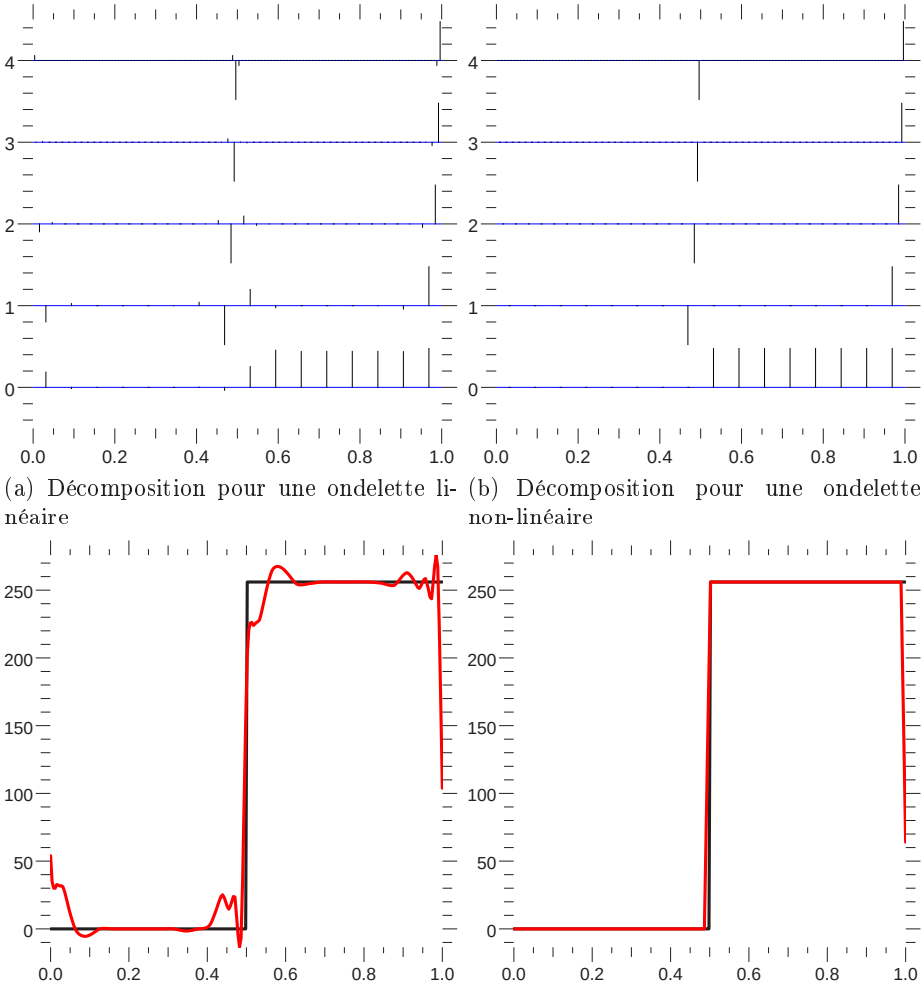
(b) Décomposition pour une ondelette non-linéaire



(c) Approximation à N termes pour une ondelette linéaire

(d) Approximation à N termes pour une ondelette non-linéaire

FIG. 7.6.: Test des ondelettes sans rebonds sur un signal continu par morceau. Nous n'utilisons ici que les opérations de prédiction. Sur les courbes : en noir, le signal original ; en rouge, le signal approximé.



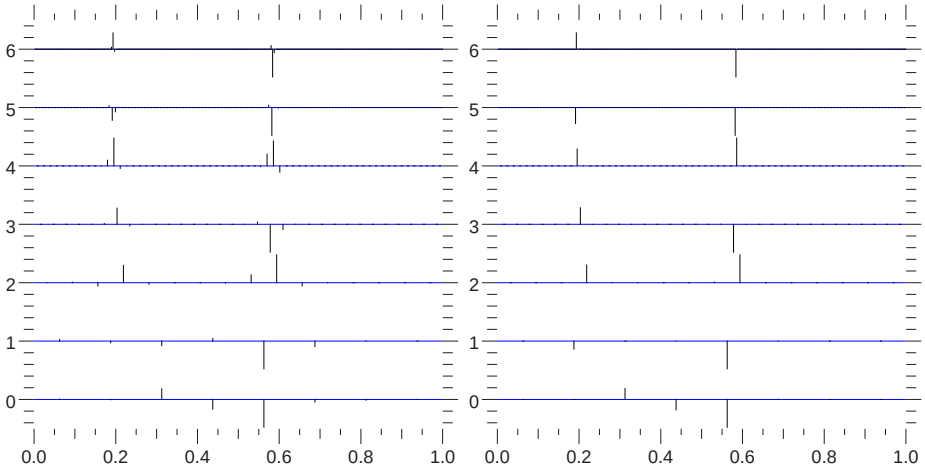
(a) Décomposition pour une ondelette linéaire

(b) Décomposition pour une ondelette non-linéaire

(c) Approximation à N termes pour une ondelette linéaire

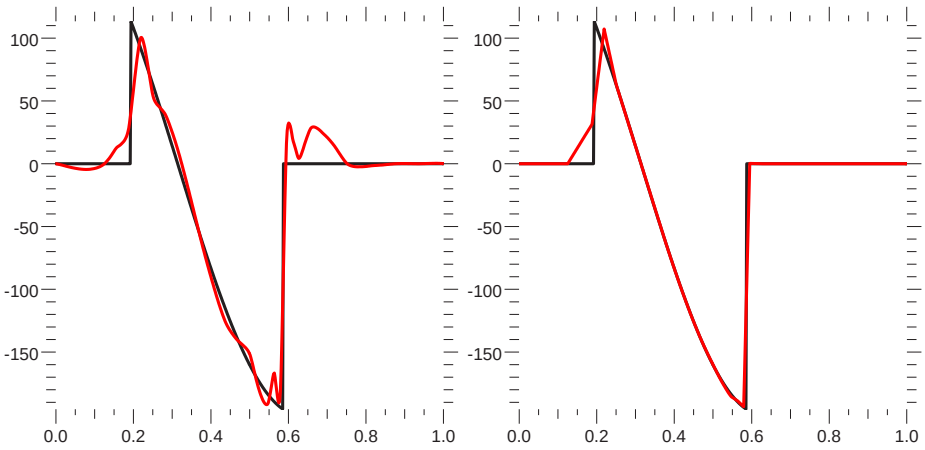
(d) Approximation à N termes pour une ondelette non-linéaire

FIG. 7.7.: Test des ondelettes sans rebonds sur un créneau. Nous utilisons ici les opérations de prédiction et de mise-à-jour. Sur les courbes : en noir, le signal original ; en rouge le signal approximé.



(a) Décomposition pour une ondelette linéaire

(b) Décomposition pour une ondelette non-linéaire



(c) Approximation à N termes pour une ondelette linéaire

(d) Approximation à N termes pour une ondelette non-linéaire

FIG. 7.8.: Test des ondelettes sans rebonds sur un signal continu par morceau. Nous utilisons ici les opérations de prédiction et de mise-à-jour. Sur les courbes : en noir, le signal original ; en rouge, le signal approximé.

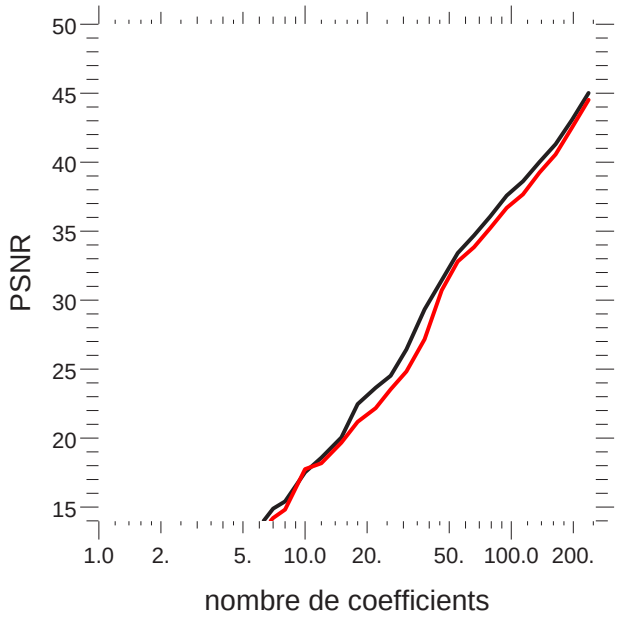


FIG. 7.9.: Test des ondelettes sans rebonds 1D sur des tranches de Lena : qualité de l'approximation à N termes. Courbe noire : ondelette linéaire. Courbe rouge : ondelette sans rebonds.

Pour des images réelles, notre schéma est moins efficace que le schéma linéaire. Nous pouvons tenter d'expliquer ceci à partir de notre étude sur le comportement des ondelettes sans rebonds (7.2.4). En effet, nous avons noté que nous perdions en qualité d'approximation aux environs des points de changement de signe de la dérivée seconde. Si ces points sont effectivement rares dans le cas des signaux synthétiques, en particulier par rapport aux discontinuités du signal, ce n'est pas le cas dans les signaux réels. La perte due au changement de l'ordre d'approximation autour des points de changement de signe compense alors le gain provenant de l'élimination des rebonds.

D'autres tests montrent que pour un schéma en prédiction seule, l'ondelette non-linéaire fournit des résultats légèrement supérieurs à l'ondelette linéaire. Cela mène à penser qu'il doit y avoir une possibilité d'améliorer la conception de l'opérateur de mise-à-jour.

7.3. Ondelettes sans rebonds en 2 dimensions

7.3.1. Conception

7.3.1.1. Incorrection de la définition par produit tensoriel

Nous l'avons mentionné précédemment, l'aspect non-linéaire du design d'une ondelette en une dimension rend difficile, si ce n'est impossible, l'extension habituelle directe en ondelettes séparables en deux dimensions.

Plus précisément, nous pouvons décrire habituellement la construction du schéma de *lifting* 2D linéaire comme sur la figure 7.10, en notant les points de la grille EE, EO, OE ou OO suivant que leurs première ou seconde coordonnées sont paires ou impaires.

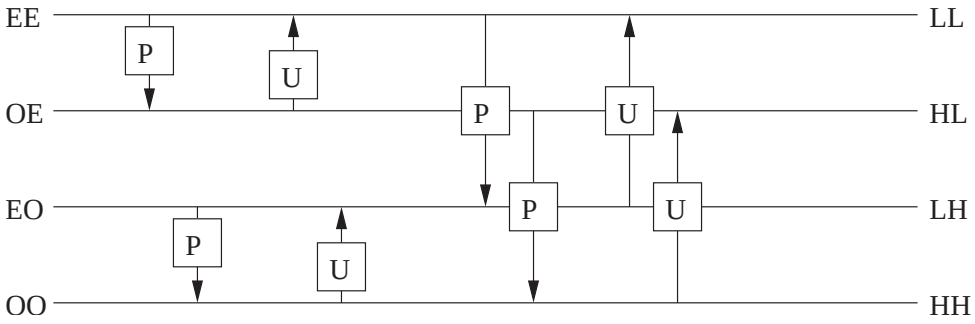


FIG. 7.10.: Schéma de *lifting* 2D linéaire

Il est possible de commuter la mise-à-jour suivant l'axe des x et la prédiction dans l'axe des y, dans la mesure où nous pouvons les écrire sous forme tensorielle comme :

$$\begin{pmatrix} I & U \\ 0 & I \end{pmatrix} \otimes I_2 \quad \text{et} \quad I_2 \otimes \begin{pmatrix} I & 0 \\ P & I \end{pmatrix}$$

Ce qui nous permet de représenter le schéma de *lifting* comme sur la figure 7.11.

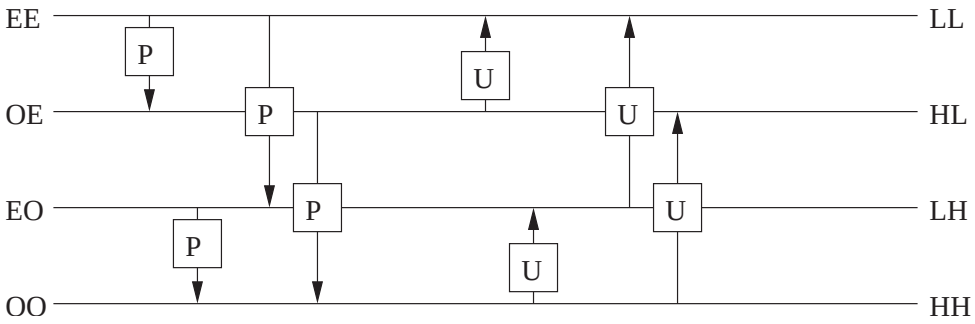


FIG. 7.11.: Schéma de *lifting* 2D linéaire modifié

7.3.1.2. Symétrisation du schéma de *lifting*

Comment pouvons-nous donc nous utiliser l'intuition qui nous a servi pour le cas à une dimension pour passer au cas à deux dimensions ?

Nous pouvons, tout d'abord, modifier l'arrangement des prédictions et des mises-à-jour pour le rendre plus symétrique, tel que sur la figure 7.12.

Ici, le pas de prédiction noté $P/2$ consiste à ne faire que la moitié de la correction par prédiction, c'est-à-dire à effectuer, en 1D :

$$h_{2k+1} = x_{2k+1} - \hat{x}_{2k+1}/2$$

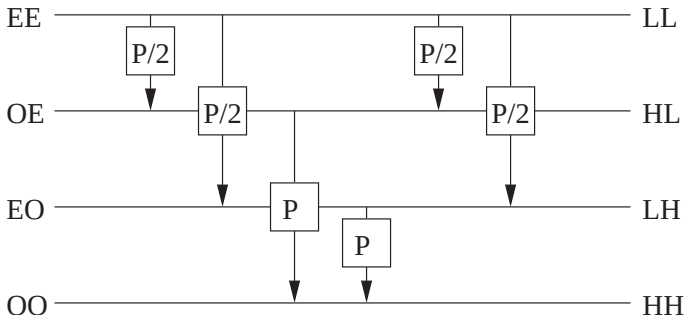


FIG. 7.12.: Schéma de *lifting* 2D linéaire symétrisé (au sein du schéma)

au lieu de

$$h_{2k+1} = x_{2k+1} - \hat{x}_{2k+1}$$

Ceci est bien évidemment directement transposable à l'opération de mise-à-jour.

7.3.1.3. Symétrisation des opérateurs

Nous pouvons aussi changer les prédicteurs eux-mêmes, pour introduire directement la symétrie de rôle entre la direction horizontale et la direction verticale.

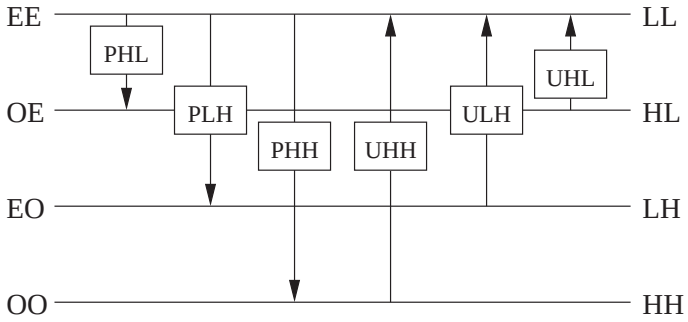


FIG. 7.13.: Schéma de *lifting* 2D linéaire symétrisé (par changement d'opérateurs)

Le prédicteur horizontal et le prédicteur vertical sont semblables à ceux utilisés dans le cas à une dimension. En revanche, le prédicteur diagonal est prévu pour être intrinsèquement à deux dimensions.

Dans le cas de la prédiction d'ordre 0, cela donne les prédicteurs suivants :

$$P_{n,0}^{HL}(x) = \mathrm{med}(x_{2k,2l}, x, x_{2k+2,2l})$$
$$P_{n,0}^{LH}(x) = \mathrm{med}(x_{2k,2l}, x, x_{2k,2l+2})$$
$$P_{n,0}^{HH}(x) = \mathrm{med}(x_{2k,2l}, x_{2k,2l+2}, x, x_{2k+2,2l}, x_{2k+2,2l+2})$$

Il est à noter que, pour le prédicteur diagonal, nous utilisons tous les plus proches voisins du point et pas seulement ceux disposés de part et d'autre sur une seule direction. Ceci nous permet d'avoir une contrainte plus fine et aussi plus cohérente..

Pour la prédiction à l'ordre 1, nous utilisons le médian d'intervalle, qui nous permet de gérer la complexité des encadrements. Pour ces derniers, étendons la notation définie par l'équation (7.1) au cas à 2 dimensions, en remplaçant les index scalaires par des index vectoriels :

$$P_{\mathbf{k};\mathbf{l},\mathbf{m}}(2\mathbf{k}) = x_{2\mathbf{k}} \tag{7.3a}$$

$$P_{\mathbf{k};\mathbf{l},\mathbf{m}}(2\mathbf{m}) - P_{\mathbf{k};\mathbf{l},\mathbf{m}}(2\mathbf{l}) = x_{2\mathbf{m}} - x_{2\mathbf{l}} \tag{7.3b}$$

De même, l'intervalle $I_{\mathbf{k},\Delta}(\mathbf{x})$ est l'intervalle de bornes $P_{\mathbf{k};\mathbf{k}-\Delta,\mathbf{k}+\Delta}(\mathbf{x})$ et $P_{\mathbf{k};\mathbf{k},\mathbf{k}+\Delta}(\mathbf{x})$. Définissons enfin :

$$\Delta_1^{HH} = (1,1) \qquad \Delta_2^{HH} = (-1,1)$$
$$\Delta^{LH} = (0,1) \qquad \Delta^{HL} = (1,0)$$

Nos nouveaux prédicteurs au point $2\mathbf{k} + \Delta_1^{HH}$ sont alors :

$$P_{n,1}^{HL}(\mathbf{x}) = p_{\text{intmed}(I_{\mathbf{k},\Delta^{HL}}(x), I_{\mathbf{k}+\Delta^{HL},-\Delta^{HL}}(x))}(x)$$
$$P_{n,1}^{LH}(\mathbf{x}) = p_{\text{intmed}(I_{\mathbf{k},\Delta^{LH}}(x), I_{\mathbf{k}+\Delta^{LH},-\Delta^{LH}}(x))}(x)$$
$$P_{n,1}^{HH}(\mathbf{x}) = p_{\text{intmed}(I_{\mathbf{k},\Delta_1^{HH}}(x), I_{\mathbf{k}+\Delta_1^{HH},-\Delta_1^{HH}}(x), I_{\mathbf{k},\Delta_2^{HH}}(x), I_{\mathbf{k}+\Delta_2^{HH},-\Delta_2^{HH}}(x))}(x)$$

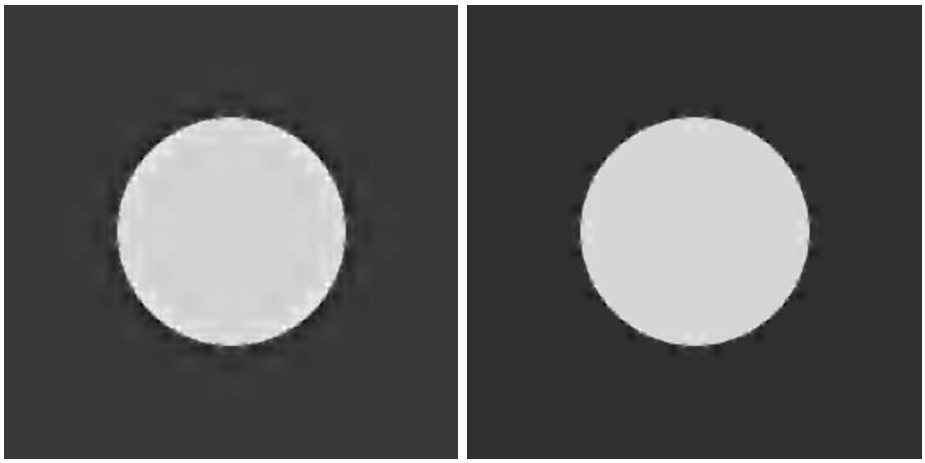
Notre médian d'intervalle devient ici important. En effet, si nous avions formulé nos opérateurs par des encadrements successifs, nous risquions d'obtenir des encadrements incompatibles entre eux. Imaginons que chacune des diagonales passant par le point que nous cherchons à prédire présente un profil linéaire, mais que les deux droites suivant les diagonales n'aient pas de point d'intersection, n'ayant pas la même valeur au point à interpoler. Comme l'encadrement d'ordre 2 proposé force les points déjà disposés en droite de ne prédire des points que sur cette droite, nous aurions deux contraintes se réduisant à un point. La solution proposée permet,elle, de prendre une valeur quelconque au sein de l'intervalle formé par les deux valeurs des profils linéaires au point qui nous intéresse.

Comme dans le cas 1D, l'encadrement proposé respecte la convexité et est invariant par ajout d'une composante linéaire.

7.3.2. Résultats

7.3.2.1. Signaux synthétiques

Nous avons expérimenté le schéma symétrisé afin de coder des images 2D synthétiques. Les résultats correspondent à ce que nous obtenons en 1D : nous avons visuellement moins d'artefacts, comme nous pouvons le constater sur la figure 7.14. Par ailleurs, des expériences de compression nous montrent que, si nous utilisons un schéma complet de prédiction et de mise-à-jour, nous gagnons entre 2 et 3 dB par rapport au cas linéaire. En revanche, si nous n'utilisons que l'opérateur de prédiction, le gain se limite à environ 0.8 dB.



(a) Ondelette linéaire (b) Ondelette sans rebonds

FIG. 7.14.: Test des ondelettes sans rebonds 2D sur un disque. Nous n'utilisons que les opérations de prédiction.

7.3.2.2. Signaux réels

À l'opposé de ce que nous obtenons pour les signaux synthétiques, mais similairement au cas 1D, le schéma utilisant des opérateurs de mise-à-jour et de prédiction non-linéaire ne rivalise pas avec le schéma linéaire. La perte sur l'image Lena est d'environ 0.7 dB. Dans le cas de l'utilisation de l'opérateur de prédiction, l'ondelette sans rebonds fait légèrement mieux que l'ondelette linéaire (de l'ordre de 0.15 dB sur *Lena*).

Par ailleurs, nous avons testé les ondelettes sans rebonds 2D sur des images résultant d'une décomposition temporelle en ondelettes (suivant le mouvement). À chaque échelle, les ondelettes sans rebonds provoquent une perte significative par rapport à l'utilisation des ondelettes 7/9.

7.4. Conclusion

Nous avons développé dans ce chapitre des ondelettes non-linéaires qui présentent l'avantage d'être continues par rapport aux données, ce qui leur garantit un bon comportement vis-à-vis de la quantification. De plus, nous avons montré que sous certaines conditions, elles disposent toujours de bonnes propriétés d'approximation par rapport aux ondelettes dont elles dérivent. Si ces ondelettes donnent des résultats probants sur des signaux synthétiques ad hoc, c'est-à-dire qui vérifient nos conditions, elles ne semblent pas, en l'état, compétitives par rapport aux ondelettes linéaires dans le cas de données réelles pour lesquelles ces conditions sont moins bien respectées.

Dans la mesure où le schéma n'utilisant que les opérateurs de prédiction semble être efficace par rapport au schéma linéaire équivalent, il est probable qu'il faille apporter des éléments d'amélioration à l'opérateur de mise-à-jour.

D'autre part, nous avons développé en (7.3.1.3) une formulation intrinséquement à 2 dimensions des ondelettes sans rebonds, à l'aide du médian d'intervalle. Ce schéma reste à tester pour le comparer à celui utilisé pour les expériences déjà effectuées et au schéma linéaire.

Quatrième partie .

Filtrage morphologique

Chapitre 8.

Morphologie Mathématique et Vidéo

La morphologie mathématique est le fondement de nombreux outils de segmentation et de filtrage. Ces outils sont particulièrement bien connus pour les images 2D, et l'extension aux images 3D est naturelle pour la plupart d'entre eux [15]. De nombreuses méthodes ont été proposées pour utiliser la morphologie mathématique en vidéo, en particulier à des fins de segmentations, ou encore d'appariement de séquences. Dans le cadre de la segmentation, la plupart des méthodes s'appuient sur une différentiation au niveau du traitement des axes spatiaux et temporel : la segmentation est effectuée uniquement dans le domaine spatial, et ses résultats sont propagés au travers de l'axe temporel.

Certains auteurs [42, 36] proposent de segmenter une première image, par exemple à l'aide de marqueurs fournis par un utilisateur, puis utilisent le résultat de leur segmentation pour créer un nouveau jeu de marqueurs qui seront projetés dans l'image suivante pour effectuer une nouvelle segmentation.

D'autres auteurs [2, 31, 14] segmentent indépendamment les images de la séquence, puis tentent d'apparier les régions obtenues.

Les deux méthodes souffrent de l'apparition de nouveaux objets dans le cadre de l'image au cours de la séquence, puisque ce nouvel objet, soit va troubler l'appariement, soit sera rattaché à l'une des régions existantes.

Par ailleurs, dans ces exemples, la corrélation temporelle entre les images est faiblement exploitée au sein de l'étape de segmentation, et les frontières des objets sont soumises à des oscillations.

Une autre solution [52] est d'utiliser un traitement 3D en accumulant les images les unes contre les autres en un grand parallépipède. Cette approche est plus prometteuse dans le sens où elle permet de prendre mieux en compte la corrélation temporelle bien qu'elle soit plus gourmande en terme de mémoire.

Malheureusement, cette deuxième approche (et certaines méthodes de la première catégorie) ne prend pas en compte le problème de l'aliasage temporel. En effet, l'échantillonage de la séquence au cours du temps fait que, lorsque de grands mouvements (par rapport à la taille de l'objet à considérer) ont lieu d'une image à l'autre, les pixels appartenant à un même objet ne sont plus connectés correctement si l'on considère que le voisin temporel d'un pixel est à la même position spatiale que celui-ci.

8.1. Éléments structurants suivant le mouvement (ESM)

8.1.1. Des idées venant d'autres domaines

Dans le domaine de la compression vidéo, nous avons vu que la méthode la plus couramment employée pour gérer le mouvement au cours de la séquence est d'utiliser la compensation de mouvement. En particulier, dans les schémas de filtrage temporel compensé par le mouvement (*Motion-Compensated Time Filtering* - MCTF), le mouvement est utilisé de facon à ce que cette utilisation soit indépendante du type de schéma de lifting adopté. L'introduction du mouvement est effectuée au niveau le plus bas, le niveau pixellique, afin de mettre en correspondance les pixels sur lesquels un filtre sera appliqué.

L'envie vient donc naturellement d'appliquer le même type de principe à la morphologie mathématique [24, 25].

8.1.1.1. L'élément structurant

La brique de base de la morphologie mathématique est l'élément structurant. Appliqué à un point donné de l'espace, il lui associe un voisinage de ce point. On utilise généralement un élément structurant fixe, défini par un ensemble des décalages par rapport au point dont on cherche le voisinage, S.

$$N : x \mapsto x + S$$

Néanmoins, Serra [44] a introduit le formalisme des fonctions structurantes : la fonction structurante fait varier le jeu des décalages en fonction du point étudié :

$$N : x \mapsto x + S_D(x)$$

La fonction structurante elle-même peut dépendre des données D : soit de l'image directement, soit des mesures dérivées sur celle-ci [26], soit encore de n'importe quelle méta-donnée imaginable qui nous donnerait une idée de la forme que doit avoir chaque voisinage. Dans la mesure où les éléments structurants fixes sont des cas particuliers des fonctions structurantes, alors constantes, nous appellerons celles-ci éléments structurants dans la suite.

L'idée de voisinage est justement celle que nous recherchons dans ce chapitre : nous souhaitons rendre "voisins" des éléments qui sont effectivement corrélés. Dans la mesure où l'aliasiage spatial est généralement faible, il est rarement utile de définir un voisinage spatial qui se distingue notablement des plus proches voisins dans une n-connexité (par exemple 4-, 6- ou 8-connexité dans le cas d'un espace discret à deux dimensions). En revanche, il devient intéressant de définir différemment le voisinage dans le cas de la direction temporelle. Il est effectivement naturel de considérer comme premier voisin temporel d'un point la position du point de l'objet auquel il appartient à l'image suivante (ou précédente, selon la direction du voisinage considéré). Agnus [2] définit un élément structurant purement temporel, orienté selon le mouvement, qui lui sert à créer un opérateur connexe. Guichard [16], au sein du formalisme des équations aux dérivées partielles (EDP), introduit lui aussi le mouvement pour effectuer des opérations là aussi

purement temporelles. Dans les deux cas, le mouvement est supposé constant au travers de la trame du point considéré, ce qui rend l'élément structurant rigide. Cela pose des problèmes lorsque la trajectoire de l'objet est courbe.

8.1.2. Définition des éléments structurants suivant le mouvement

8.1.2.1. Opérateurs de compensation

Formalisons notre idée. Notons v^+ le champ de mouvement de la séquence dans la direction avant, et v^- celui dans la direction arrière. Dans un premier temps, nous considèrerons que les champs de mouvement sont à précision entière, et donc qu'à chaque position de pixel, ils associent une autre position exacte de pixel. Ces deux champs de mouvement permettent de définir 2 opérateurs de compensation, M^+ et M^-, comme suit :

$$M^+(t,x,y) = (t+1, x + v_x^+(t,x,y), y + v_y^+(t,x,y)) \tag{8.1}$$

$$M^-(t,x,y) = (t-1, x + v_x^-(t,x,y), y + v_y^-(t,x,y)) \tag{8.2}$$

En généralisant la notation, nous arrivons à :

$$M^p = \begin{cases} Id & \text{si } p = 0 \\ (M^+)^p & \text{si } p > 0 \\ (M^-)^{-p} & \text{si } p < 0 \end{cases} \tag{8.3}$$

En complément, nous définissons l'opérateur de translation T par :

$$T_{(a,b)}(t,x,y) = (t, x+a, y+b).$$

Notons aussi M_0 l'opérateur de compensation pour un champ nul (ce qui revient à effectuer une translation dans le sens de l'axe temporel).

8.1.2.2. Modification des voisinages

Supposons maintenant que nous disposons d'un élément structurant à 3 dimensions, S qui, par définition, associe à tout point $\mathbf{x}$ son voisinage

$$V(\mathbf{x}) = \{\mathbf{x}\} + S(\mathbf{x}) \tag{8.4}$$

soit encore

$$V(\mathbf{x}) = \{(t+\delta t, x+\delta x, y+\delta y) | (\delta t, \delta x, \delta y) \in S(\mathbf{x})\} \tag{8.5}$$

Éléments S-T Nous pouvons alors construire un nouveau voisinage par :

$$V_M(\mathbf{x}) = \{M^{\delta t}(T_{(\delta x, \delta y)}(t,x,y)) | (\delta t, \delta x, \delta y) \in S(\mathbf{x})\} \tag{8.6}$$

Au lieu de suivre l'axe temporel, le voisinage suit pour chaque point son "fil" de mouvement (figure 8.2(a) comparée à la figure 8.1(a)). Si le champ de mouvement est idéalement

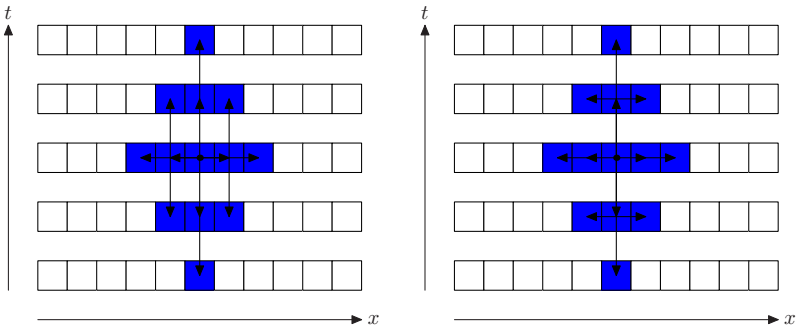
correct, cela signifie que nous avons bien rendu à nouveau connexe l'objet que nous suivons en dépit de l'aliasage.

Il est à noter que cela ne définit pas non plus une seule composante connexe à l'objet dans tout le volume spatio-temporel. Si la définition de notre nouveau voisinage permet de passer outre l'aliasage temporel, elle ne résoud en rien - pour autant qu'il aurait fallu la résoudre - la séparation de l'objet en plusieurs composantes connexes dans le volume spatio-temporel du fait d'occlusions momentanées.

Éléments T-S Dans les éléments de type S-T, nous avons d'abord appliqué le décalage spatial puis, ensuite seulement, le décalage temporel. La déformation n'est pas identique à celle effectuée dans l'autre ordre, entre autres du fait de la possible convergence du champ (cf. 8.1.3.2). Nous pouvons donc aussi bâtir notre nouveau voisinage comme

$$W_M(\mathbf{x}) = \{T_{(\delta x,\delta y)}(M^{\delta t}(t,x,y))|(\delta t,\delta x,\delta y) \in S(\mathbf{x})\}, \tag{8.7}$$

c'est-à-dire en suivant d'abord le fil de mouvement du centre du voisinage, puis en nous déplaçant dans l'espace (figure 8.2(b) comparé à la figure 8.1(b). Ces deux variantes de voisinage définissant ces éléments structurants suivant le mouvement (ESM) ne possèdent pas les mêmes propriétés et peuvent être chacune intéressantes selon le type d'opération à effectuer.



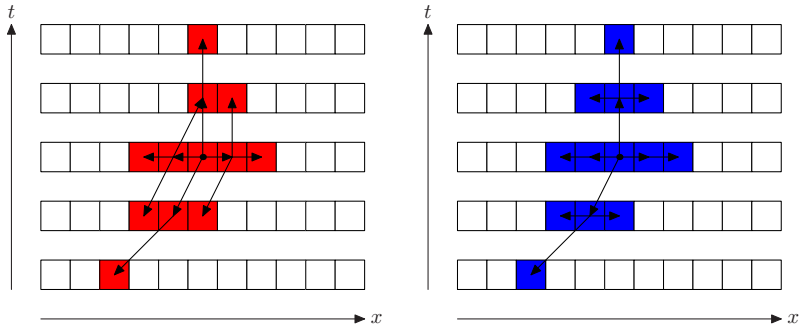
(a) $S-T$ (b) $T-S$

FIG. 8.1.: Voisinage "classique", correspondant à un voisinage d'ESM avec un champ de mouvement nul. À gauche, les voisins sont obtenus en appliquant d'abord le déplacement spatial, puis ensuite le déplacement temporel (S-T). À droite, les voisins sont obtenus en appliquant d'abord le déplacement temporel, puis ensuite le déplacement spatial (T-S). Les deux constructions sont équivalentes ici.

8.1.3. Quelques considérations pratiques

8.1.3.1. Disparition et occlusion des objets

Mais de fait, que se passe-t-il en cas de disparition de l'objet sur lequel notre point se trouve ? Bien évidemment, cela dépend de la facon dont notre champ de mouvement



(a) ST (b) TS

FIG. 8.2.: Voisinage issu d'un ESM, avec un champ non nul. À gauche, les voisins sont obtenus en appliquant d'abord le déplacement spatial, puis ensuite le déplacement temporel (S-T). À droite, les voisins sont obtenus en appliquant d'abord le déplacement temporel, puis ensuite le déplacement spatial (T-S). Les deux constructions ne sont pas identiques.

gère cette situation. Certains modèles se permettent de pointer hors de l'image, ou de signaler qu'un correspondant n'a pas été trouvé pour le point ou la région considéré.

Une première possibilité pour traiter cette question est de poser par convention :

$$v(t, x, y) = (+\infty, +\infty).$$

Pour notre élément structurant, cette situation revient à se trouver sur un bord : le point est projeté à l'infini, et notre traitement, qui s'effectue sur l'image, subit, dans le cadre pratique, l'intersection du voisinage avec l'image :

$$V_{M,I}(t, x, y) = V_M(t, x, t) \cap I.$$

Il est à noter que, de facon similaire au cas de l'élément structurant purement spatial au bord, l'intersection avec le cadre change la cardinalité de l'élément structurant. Il faut donc être prudent dans le cas d'utilisation de filtres de rang par exemple. Il s'agit bien d'un inconvénient puisqu'il y a toutes les chances que le nombre de voisinages n'ayant pas la cardinalité maximale augmente par rapport au traitement purement spatial.

L'autre possibilité est que le modèle donne un vecteur de mouvement incorrect qui pointe néanmoins approximativement vers le lieu de la disparition de l'objet. Dans ce cas, nous en revenons au cas d'une frontière de notre objet : de même qu'il n'est pas possible, ni raisonnable, de déformer un élément structurant dans l'espace de manière à complètement éviter les frontières des objets, il n'est pas anormal que, lorsqu'un objet disparaît, ses voisins temporels ne lui appartiennent plus. Cela permet par ailleurs que les objets puissent être reconnectés entre eux dans le cas d'une ligne de partage des eaux avec marqueurs ou d'une cascade.

Dans le cas des éléments T-S, il est aussi possible que l'image du centre par l'opérateur de compensation soit un pixel à la bordure de l'image. Les voisins de l'image du centre

seront donc éventuellement hors de l'image. A l'inverse, l'image par l'opérateur de compensation d'un point du bord n'est pas nécessairement elle-même au bord. Dans ce cas, les voisins directs du point image seront tous à l'intérieur de l'image.

8.1.3.2. Convergence et divergence du champ de mouvement

Des points en moins, des trous en plus Sur certains points, le champ de mouvement converge. Cela signifie qu'il est possible que deux points distincts dans un élément structurant 3D se retrouvent associés au même point dans un élément structurant de type S-T après utilisation de l'opérateur de compensation (point B sur la figure 8.3). Il en résulte donc un changement de cardinalité du voisinage par rapport au voisinage obtenu par l'élément structurant 3D classique. À l'opposé, à certains endroits, le champ de mouvement diverge. À ces endroits-là, il peut arriver que deux points, initialement voisins spatialement à l'intérieur d'un voisinage, ne le soient plus après utilisation de l'opérateur de compensation (point A sur la figure 8.3). Les deux effets conjugués mènent à un risque de changement de forme du voisinage entre celui obtenu par l'élément structurant 3D et l'élément structurant de type S-T.

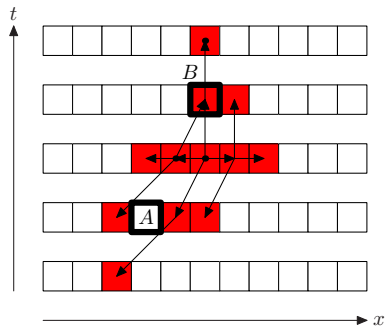


FIG. 8.3.: Illustration du problème de convergence/divergence du mouvement pour un ESM de type S-T. A est un nouveau trou dans une tranche spatiale du voisinage. B est la position commune de plusieurs points après application de la déformation.

Les éléments structurants de type T-S ne posent pas ce type de problème : comme le déplacement spatial s'effectue après le déplacement temporel, la forme de l'élément structurant sur un plan temporel donné est indépendante du mouvement, seule sa position varie alors. Par conséquent, des pixels distincts sur un même plan image dans l'espace 3D le seront toujours dans l'espace déformé 2D+t.

Cardinalité Une des questions que soulève la convergence du champ de mouvement est celle de la cardinalité du voisinage de chaque point. En effet, La logique de la définition de certains opérateurs dépend de cette cardinalité. C'est le cas par exemple des filtres de rang et du filtre médian. Rappelons la définition du filtre de rang k, R_k

$$R_k(f)(x) = f(y_{x,k})$$

où $\{y_{x,i}\}$ est le voisinage de x et où $f(y_{x,i}) \leq f(y_{x,j})$ si $i < j$; le filtre de rang remplace la valeur de chaque point par la k-ième plus petite valeur de son voisinage. Le filtre médian, quant à lui, prend la valeur médiane sur le voisinage :

$$\mathrm{med}(f)(x) = \operatorname*{med}_{y \in V(x)} f(y)$$

La solution utilisée pour pallier ce problème dans le cas des bords est de définir les filtres de rang non plus par le rang exact mais par un quantile $\alpha \in [0, 1]$. Le filtre prend alors la k-ième plus petite valeur dans le voisinage, avec k défini par :

$$k = \lfloor \alpha \operatorname{card} V(x) \rfloor.$$

Cette définition peut être réutilisée telle quelle ici.

Néanmoins, il est possible d'envisager une autre solution, plus satisfaisante. Celle-ci consiste à maintenir le nombre d'éléments au sein du voisinage par ESM, en attribuant à chaque point du voisinage un poids proportionnel au nombre de points du voisinage classique qui sont envoyés sur lui après déformation : si deux points sont déformés sur le même point, alors la valeur de l'image à ce point est comptée deux fois, et ainsi de suite. Cela revient en pratique à considérer un voisinage non plus comme un ensemble de points, mais comme une liste, où un même élément peut apparaître plusieurs fois. Ceci posséde une certaine logique : dans la mesure où le champ de mouvement réel n'est pas en réalité à valeur entière, supposer que les points sont confondus relève généralement plus d'une vue de l'esprit que de la réalité ; nous utilisons simplement le point considéré comme approximation du point de coordonnées non-entières qui est le résultat de la compensation. Cela ne doit donc pas masquer le fait que les deux points images sont généralement distincts, et qu'il est donc naturel de compter chacun de ceux-ci pour le calcul d'un filtre.

Cette solution peut donner des résultats sensiblement différents, à la fois du cas où le problème de changement de cardinalité n'est pas pris en compte, et à la fois du cas où l'on utilise les quantiles pour définir les filtres de rang. Ceci est illustré sur la figure 8.4. Nous recherchons la valeur résultant de l'application d'un filtre médian sur le point en rouge, lorsqu'on utilise un voisinage issu d'une 8-connexité subissant une transformation S-T.

- Dans un voisinage classique, il y a 9 éléments (8 voisins plus le centre). Le nombre impair de valeurs nous garantit que le médian est correctement défini.
- Nous n'avons plus ici que 8 voisins. Certaines définitions du médian, dans le cas où celui-ci s'applique sur un nombre pair d'éléments, lui font prendre comme valeur la moyenne des deux valeurs centrales. Le résultat serait donc ici 5.5.
- Si nous définissons le médian comme le filtre de quantile 0.5, nous prendrons donc la 4ème plus petite valeur des voisins, c'est-à-dire 4.
- Si nous tenons compte de la multiplicité des points convergents, le 9 compte 2 fois, et la valeur médiane est alors 7.

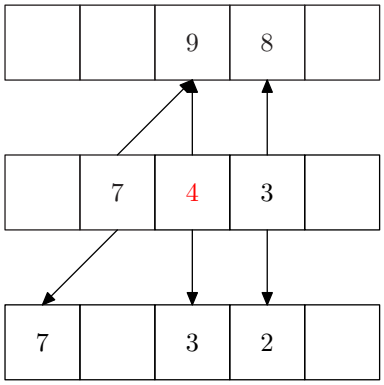


FIG. 8.4.: Illustration du problème de convergence sur la cardinalité du voisinage. Avec un voisinage "classique" correspondant à une 8-connexité, le médian du point en rouge est : 4 pour un filtre médian exprimé en quantile ($\alpha = 0.5$), 5.5 pour un filtre médian où le médian est pris comme moyenne des deux éléments centraux dans le cas d'un nombre pair d'éléments, 7 si la multiplicité des points est prise en compte.

Nous constatons donc ici, sur un exemple particulier, combien les différentes approches mènent à des résultats bien distincts. Utiliser la multiplicité des points déformés nous semble la solution la plus correcte.

D'autres solutions, plus complexes à mettre en œuvre, car utilisant la composante sous-pixellique des vecteurs de mouvement, sont proposées en (8.5) pour pallier le problème du maintien de la cardinalité.

Transformation en tout-ou-rien L'autre problème engendré par la convergence et divergence du champ dans le cas des éléments S-T se trouve dans l'application de la transformation en tout-ou-rien. Dans cette transformation, deux éléments structurants sont utilisés conjointement. Soit S_1 et S_2 deux éléments structurants. Alors, la transformation en tout-ou-rien associé est définie par :

$$\mathrm{HOM}(X) = \{x | S_1(x) \in X \text{ et } S_2(x) \cap X = \emptyset\} \tag{8.8}$$

Afin que l'opération ait véritablement un sens, il est préférable d'assurer que :

$$S_1(x) \cap S_2(x) = \emptyset \qquad \forall x \in \mathcal{I}, \tag{8.9}$$

sinon il est clair que le point ne pourra appartenir au résultat, quelle que soit la configuration véritable dans laquelle se trouve le point. Si ceci est simple à assurer dans le cadre habituel, la déformation qu'impose le mouvement aux ESM de type S-T ne permet pas de garantir *a priori* cette propriété du fait de la possible convergence du mouvement.

Plusieurs solutions sont envisageables pour résoudre ce problème. Tout d'abord, il est possible de décider que l'un des éléments a plus de poids que l'autre. Si l'on souhaite par

exemple donner plus de poids à l'intérieur de l'objet qu'à l'extérieur, on peut utiliser, à la place de $S_{2,M}$, l'élément structurant $S'_{2,M}$ défini par :

$$S'_{2,M}(x) = S_{2,M}(x) \setminus S_{1,M}(x).$$

A l'inverse, on peut considérer que les points appartenants aux deux éléments structurants sont des points indéterminés, et donc remplacer $S_{1,M}$ et $S_{2,M}$ par $S'_{1,M}$ et $S'_{2,M}$ définis chacun comme ci-dessus. Mais ces solutions ne sont pas satisfaisantes. En effet, comme on peut le voir sur la figure 8.5, la déformation de type S-T peut changer l'homotopie associée aux deux éléments structurants. Utiliser des différences d'éléments structurants ne peut ici qu'aggraver le problème.

- Dans la situation classique, c'est-à-dire sans mouvement (figure 8.5(a)), la transformation en tout-ou-rien nous permet de trouver certains bords diagonaux. L'expression "bord diagonal" est ici utilisée abusivement, dans la mesure où l'on ne se situe pas pleinement dans des dimensions spatiales. Nous trouverons plutôt les situations d'occlusion de la forme par un autre objet sur la droite.
- Après déformation selon le mouvement (figure 8.5(b)), deux problèmes émergent : la convergence des points de la forme et de ceux du fond en un même point, d'une part, et d'autre part l'apparition d'un trou dans une des tranches spatiales, séparant en deux morceaux ce qui n'en était qu'un auparavant.
- Si l'on privilégie la forme, la transformation trouvera aussi tout l'intérieur de celle-ci.
- Si l'on privilégie le fond, on détectera aussi toutes les zones de disparition de l'objet.
- Si l'on remplace les points problèmatiques par des indéterminations, nous aurons les deux types de faux positifs par rapport à notre objectif initial : intérieur spatial de l'objet, et occlusions venant de toutes les directions
- Dans tous les cas, aucune solution n'est apportée à l'apparition du trou spatial dans $S_{1,M}$.

Il est important de souligner que ce problème ne concerne que les éléments de type S-T ; en effet, les éléments de type T-S respectent mieux cette contrainte d'homotopie, en maintenant la forme respective des deux éléments structurants de la transformation. C'est donc eux qui devraient être utilisés pour les opérations de type transformation en tout-ou-rien.

La convergence comme avantage Si la convergence du champ peut sembler problèmatique dans le cas des éléments de type S-T, comme nous venons de la voir pour les filtres de rang et les transformations en tout-ou-rien, leur respect de celle-ci leur offre tout de même un avantage net : une des raisons pour lesquelles nous souhaitions déformer notre voisinage était de pouvoir considérer comme étant voisin d'un point ce qui lui correspondait de la manière la plus juste possible, et d'éviter donc de "mélanger les pommes et les poires", c'est-à-dire des points appartenant selon toute vraisemblance à des objets différents. C'est justement en tenant compte de la convergence que l'on peut

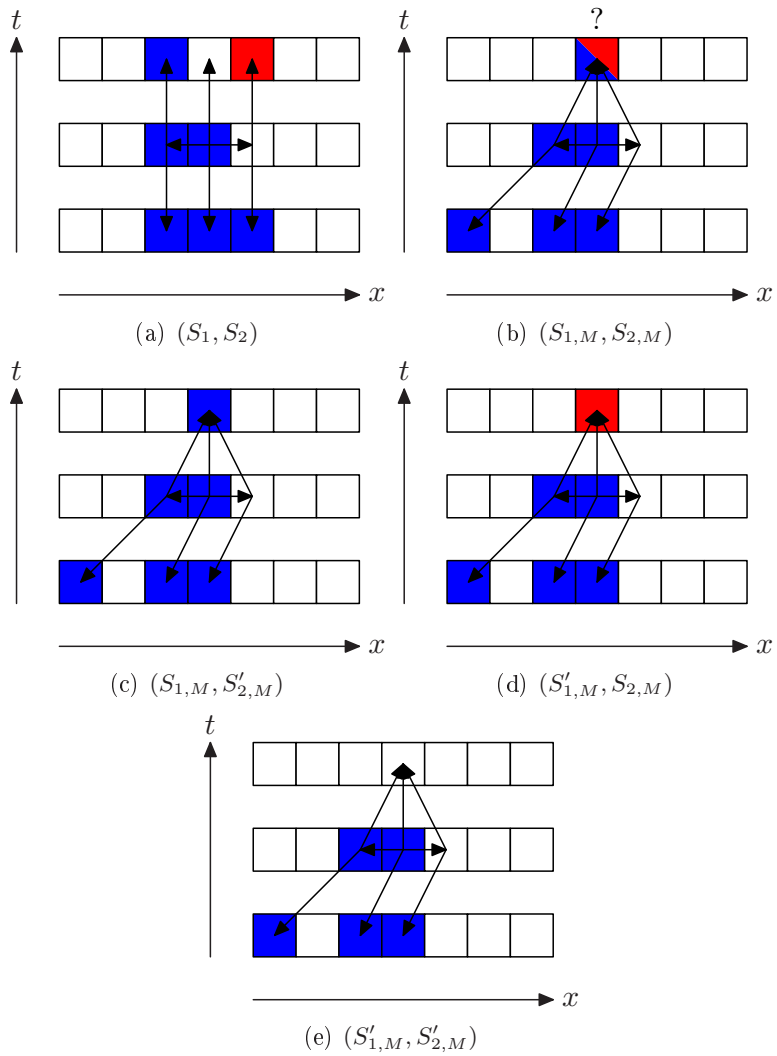


FIG. 8.5.: Illustration du problème de la transformation en tout-ou-rien combinée avec les ESM : en bleu, les points du voisinage devant appartenir à la forme ; en rouge, ceux devant appartenir au fond.

tenir compte de cette exigence, puisque nous donnons alors vraiment (pour autant que notre champ soit juste) un voisinage qui est lié le mieux possible à son centre. Cette exigence est moins bien prise en compte dans le cas des éléments de type T-S, du fait de la rigidité de leur forme sur un plan temporel donné. Ils ont aussi des difficultés à gérer les cas de rotation des objets.

8.1.3.3. Dualité par adjonction

Supposons que nous définissions une érosion à l'aide de notre élément structurant. Nous avons par définition :

$$\varepsilon(X) = \{x \in X | V(x) \subset X\} \tag{8.10}$$

$$\varepsilon(f)(x) = \min_{y \in V(x)} f(y) \tag{8.11}$$

De nombreuses implémentations de la dilatation utilisent une formule similaire[1] pour la définir, où $\check{V}(x)$ est le symétrisé de $V(x)$ par rapport à x :

$$\delta(Y) = \bigcup_{y \in Y} \check{V}(y)$$

$$\delta(f)(y) = \max_{x \in \check{V}(y)} f(x)$$

Cela correspond effectivement à la dilatation adjointe quand l'élément structurant est invariant par translation. Ce n'est malheureusement pas le cas ici. Il nous faut donc revenir à la définition de la dilatation par adjonction :

$$Y \subseteq \varepsilon(X) \iff \delta(Y) \subseteq X$$

Celle-ci nous mène directement à :

$$\delta(Y) = \{y | V(y) \cap Y \neq \emptyset\} \tag{8.12}$$

$$\delta(f)(y) = \max_{x | y \in V(x)} f(x) \tag{8.13}$$

En pratique, que cela signifie-t-il ? Habituellement, l'érosion et la dilatation sont toutes les deux implémentées comme des opérations collectrices : le processus considère un point, détermine son voisinage et établit le minimum ou le maximum (selon qu'il s'agit de l'érosion ou de la dilatation) sur celui-ci, le résultat étant stocké dans l'image résultat (Algorithme 4). Ici, si le procédé reste identique dans le cas de l'érosion, il nous faut transformer le schéma de la dilatation en un processus envoyant de l'information vers les voisins. Nous plaçons tout d'abord des valeurs minimales dans chacun des points de l'image d'arrivée. Nous parcourons ensuite les voisinages de chacun des points, et nous plaçons à chaque fois le maximum entre le point du voisinage dans l'image d'arrivée

[1] Dans le cas binaire, la similarité n'est pas évidente du point de vue de la formulation. Néanmoins, comme elle est implémentée de la même facon que la dilatation à niveau de gris, le problème entre tout de même en ligne de compte.

et le point parcouru, dans le point du voisinage de l'image d'arrivée (Algorithme 5). Ce nouvel algorithme n'est pas plus compliqué que le premier, si nous considèrons que nous ne disposons pas de méthode d'une complexité moins que linéaire pour obtenir le maximum d'un voisinage non-trié.

Algorithme 4 Algorithme d'érosion (collection des minima)

pour tout $\mathbf{x} \in I$ **faire**
  $I_{\text{out}} \leftarrow +\infty$
  pour tout $\mathbf{x}' \in V(\mathbf{x})$ **faire**
    $I_{\text{out}}(\mathbf{x}) \leftarrow \min(I_{\text{out}}(\mathbf{x}), I_{\text{in}}(\mathbf{x}'))$
  fin pour
fin pour

Algorithme 5 Algorithme de dilatation (diffusion des maxima)

pour tout $\mathbf{x} \in I$ **faire**
  $I_{\text{out}} \leftarrow -\infty$
fin pour
pour tout $\mathbf{x} \in I$ **faire**
  pour tout $\mathbf{x}' \in V(\mathbf{x})$ **faire**
    $I_{\text{out}}(\mathbf{x}') \leftarrow \max(I_{\text{out}}(\mathbf{x}'), I_{\text{in}}(\mathbf{x}))$
  fin pour
fin pour

8.2. Modèles de mouvement adaptés aux ESM

Comme nous l'avons déjà mentionné, il est important, lorsqu'on estime le mouvement, de savoir quelle utilisation sera faite des résultats afin de choisir un modèle de mouvement adéquat. Dans le cas des éléments structurants suivant le mouvement, quelles sont les caractéristiques qui sont intéressantes à obtenir pour le champ de mouvement ?

8.2.1. Mouvement par objet

Dans l'idéal, nous souhaiterions utiliser un champ de mouvement estimé sur chacune des régions ("objets") de l'image. Mais n'est-ce pas un peu "tourner en boucle", dans la mesure où une des applications principales de la morphologie mathématique est justement l'obtention d'une segmentation des objets ? Si nous disposons déjà d'un champ de mouvement pour chaque région, cela signifie que nous connaissons déjà ces régions, et donc que le traitement morphologique est plus ou moins superflu. Bien évidemment, il est possible d'utiliser une sorte de boucle de rétro-contrôle : on commence par filtrer sans mouvement, on segmente, puis on estime le mouvement à partir de cette segmentation, on recommence alors les cycles de filtrage/estimation/segmentation en utilisant dorénavant le mouvement, jusqu'à ce que les régions et le mouvement obtenus soient stables.

Le procédé paraît néanmoins coûteux par rapport à des méthodes prévues pour estimer et segmenter conjointement le champ de mouvement (par exemple [32]).

8.2.2. Étude des discontinuités

L'intérêt d'un champ de mouvement estimé par région est de faire correspondre les discontinuités réelles du mouvement (qui sont incluses dans les limites des objets) et les discontinuités de la modélisation. Si nous ne pouvons imposer cela, nous faisons face à deux risques : soit d'avoir une discontinuité de l'estimation qui n'est pas placée adéquatement par rapport à une discontinuité réelle du mouvement, soit d'avoir une évolution continue du mouvement en un point où celui-ci est en réalité discontinu.

Imaginons que notre modèle nous impose la position des discontinuités, comme cela est le cas par exemple avec le modèle constant par morceaux généralement utilisé pour le block-matching. Il y a de fortes chances que les lignes de ces discontinuités coupent les lignes des vraies discontinuités du mouvement. À ces endroits-là, un filtrage, telle qu'une érosion ou une dilatation, va donner un artéfact. En effet, la frontière réelle sera déplacée dans 2 directions différentes et la ligne de frontière deviendra une ligne brisée, alors qu'elle ne l'était pas initialement.

Étudions maintenant le cas où le modèle impose la continuité sur une discontinuité du mouvement. Que se passe-t-il ? Nous effectuons notre filtrage en projetant un objet déformé continûment. Cela signifie que nous n'introduisons pas d'irrégularités sur les frontières. En revanche, il est bien évident qu'il y aura aux endroits des discontinuités réelles du mouvement une erreur dans le voisinage. Dans la mesure où cette erreur est faible et continue, il est à noter que l'impact de l'erreur se limitera sur l'entourage des objets. Cette remarque, combinée au fait qu'il n'y a pas d'irrégularité introduite dans la frontière, pousse à estimer que cette situation est préférable au cas précédent dans le cadre du filtrage.

Il existe des champs de mouvement qui n'imposent ni continuités, ni discontinuités. Ceux-là paraissent à première vue les plus adaptés, puisque l'on peut espérer que leurs discontinuités devraient correspondre à celle du mouvement. Mais nous savons malheureusement que bien rares sont les estimations exemptes d'erreurs. Dans le cas de tels champs, cela signifie qu'un pixel (ou un petit groupe de pixels voisins) peut être associé à un mouvement faux, tandis que les voisins seront associés à un mouvement correct. On risque dans ce cas de créer de nouvelles erreurs qui se seraient pas produites dans un traitement purement 3D. Imposer sur l'essentiel du domaine du champ une certaine continuité permet de contrôler ce genre d'erreur.

Dans la suite, nous utilisons le champ de mouvement issu de la méthode décrite en 4, qui présente les caractéristiques que nous recherchons.

8.3. Implantation

Intégrer les ESM au sein d'une librairie de fonctions de morphologie mathématique nécessite de modifier un des éléments du cœur de cette librairie. Si aucune souplesse n'a été prévue pour définir des éléments structurants dynamiques (au sens où la forme du

voisinage dépend du point considéré, au-delà des simples effets de bords), leur mise en oeuvre est relativement pénible, puisque cela signifie réécrire chacune des fonctions de la librairie pour qu'elle ait une variante dépendant du mouvement. C'est le cas par exemple de la librairie Xlim3d du Centre de Morphologie Mathématique.

A l'inverse, si l'intégration d'éléments structurants dynamiques a été prévue dès le départ, la mise en place s'en trouve grandement facilitée. C'est le cas pour la librairie orientée objet Morphée, développée elle-aussi au Centre de Morphologie Mathématique dans la continuité de Xlim3d.

8.4. Expériences

Des expériences simples ont été menées combinant des outils morphologiques simples avec les ESM. Nous avons utilisé un élément structurant octaédrique classique comme élément structurant de base. Celui-ci correspond à l'utilisation d'une 4-connexité. Pour cet élément structurant, les transformations S-T et T-S donnent les mêmes résultats.

Nous avons utilisé des blocs de 10 trames, provenant des séquences *Foreman* et *Mobile*, à la résolution CIF (352×288 pixels).

L'application d'une simple érosion de taille 1 sur *Foreman* (figure 8.6) montre l'intérêt de la compensation de mouvement dans le filtrage morphologique 2D+t, d'un point de vue visuel. Si nous utilisons un élément structurant classique, l'érosion abîmera visuellement l'image de façon significative. Le foreman se retrouve avec 3 narines, là où il ne devrait en avoir qu'une seule ! En revanche, si nous utilisons un ESM, le résultat est totalement acceptable visuellement. Nous avons résolu le problème d'aliasage qui déconnectait un objet avec lui-même au cours du temps du fait du mouvement.

La même constatation est vraie pour les ouvertures et les fermetures. Une ouverture de taille 3 sur la séquence *Mobile* avec un élément structurant classique va faire disparaître les points blancs de la balle, ce qui n'arrive pas si nous utilisons un ESM (figure 8.7). A l'inverse, une fermeture de taille 3 sur la même séquence avec un élément structurant classique fait apparaître des trainées sur la balle (comme l'on peut voir les trainées des phares sur les photos de circulation ayant un temps d'exposition trop important). La même fermeture avec un ESM donne, quant à elle, des points nets sur la balle.

Enfin, nous avons effectué des expériences simples de segmentation avec marqueurs. Nous avons tenté de segmenter la narine de foreman par rapport au reste de l'image, en plaçant des marqueurs adéquats. Un marqueur a été placé sur la $4^{\text{ème}}$ trame pour la narine, et un autre sur la $3^{\text{ème}}$ trame pour le fond (figure 8.9). Nous avons ensuite effectué une ligne de partage des eaux avec ces marqueurs. Si nous utilisons un élément structurant classique, le système échoue quasiment totalement à suivre la narine : elle n'est présente que sur les trames 2 à 4 (figure 8.10). En revanche, en utilisant un ESM, nous réussissons à suivre la narine sur toutes les trames considérées, avec un peu plus d'aisance dans la direction du temps qu'en le remontant (figure 8.11). Nous avons clairement réussi à reconnecter les différentes composantes connexes de la narine de l'espace à 3 dimensions, grâce au mouvement.

Cette expérience de segmentation est intéressante, puisqu'elle montre, outre la validité

de notre approche, l'intérêt d'effectuer ce type de segmentation par rapport aux schémas de segmentation récursifs. Dans une approche interactive, il n'est en effet pas nécessaire de placer les marqueurs de chaque objet dans la même trame. Il est donc possible pour l'opérateur de placer les marqueurs dans les trames où il est le plus à même de localiser les objets. Cela rend aussi plus facile la gestion de l'apparition et de la disparition des objets.

8.5. ESM et champs de mouvement à précision sous-pixellique

Les méthodes actuelles nous permettent d'obtenir aisément des estimations du mouvement avec une précision sous-pixellique. Par exemple, les méthodes de block-matching offrent souvent une précision de l'ordre de $1/8^{\text{ème}}$ de pixel, voire plus. De même, la méthode d'estimation de mouvement proposée en (4.1) nous offre une précision *a priori* quelconque. Comment utiliser cette information dans les ESM ?

Plusieurs pistes s'offrent à nous. La première est d'utiliser des méthodes d'interpolation afin de déterminer les valeurs de voisins quelque peu virtuels puisque situés à des positions non-entières. La seconde est de sur-échantillonner l'image sur laquelle nous souhaitons travailler, effectuer notre traitement, puis la sous-échantillonner. Cette piste, bien que proche de la précédente, ne lui est cependant pas identique, dans la mesure où le traitement s'effectuera aussi sur les nouveaux points créés. Enfin, la dernière idée consiste à utiliser intelligemment l'information sous-pixellique afin de déformer de manière plus satisfaisante nos éléments structurants en restant sur les points entiers. Mais avant toute chose, il nous faut réfléchir sur les schémas d'interpolation adéquats dans notre cadre.

8.5.1. Interpolation et Morphologie

Notons tout d'abord que toutes les idées que nous allons développer sur les rapports entre interpolation et morphologie ne sont pas valables uniquement dans le cadre des ESM, mais dans celui plus général de voisinages dont les points sont sur des positions non-entières par rapport à la grille de travail. Ce cadre est différent de celui de la morphologie sur l'espace réel, et il mérite donc une étude particulière.

Les deux premières idées que nous avons mentionnées supposent qu'au lieu de travailler dans un espace de départ $\mathcal{I}$ (la grille de notre image, de notre séquence), nous allons travailler à un moment sur un espace de départ $\tilde{\mathcal{I}}$, qui correspond à la précision du champ de mouvement. $\tilde{\mathcal{I}}$ peut être un raffinement de la grille $\tilde{\mathcal{I}}$, ou bien même une limite en partie continue de ces raffinements (par exemple, les plans correspondant à chacune des images dans une séquence).

Nous passons alors d'une fonction $f : \mathcal{I} \to \mathcal{J}$ à une fonction $\tilde{f} : \tilde{\mathcal{I}} \to \mathcal{J}$. Ce passage est effectué par interpolation, ce qui signifie en particulier le choix d'une méthode d'interpolation. Les schémas d'interpolation ne manquent pas, mais ils ont des propriétés bien différentes les uns des autres. Afin de sélectionner un schéma, nous devons nous interroger sur les contraintes que pose le travail sur des opérateurs morphologiques.



(a) Original



(b) Érosion avec un élément structurant classique

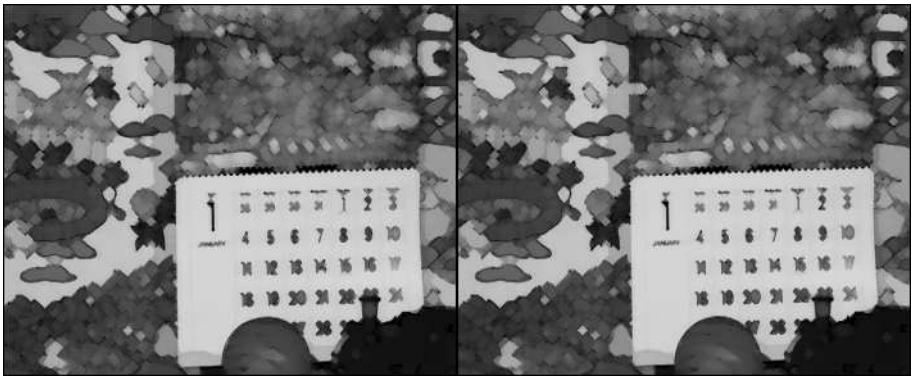


(c) Érosion avec un ESM

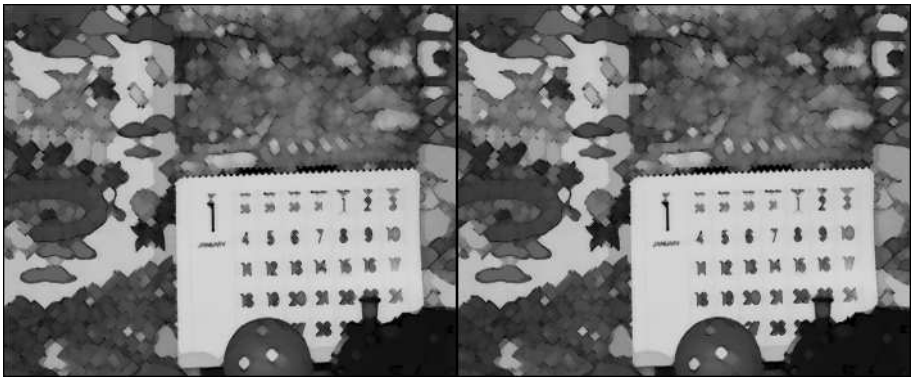
FIG. 8.6.: *Foreman* (trames 3-4) : érosion de taille 1.



(a) Original

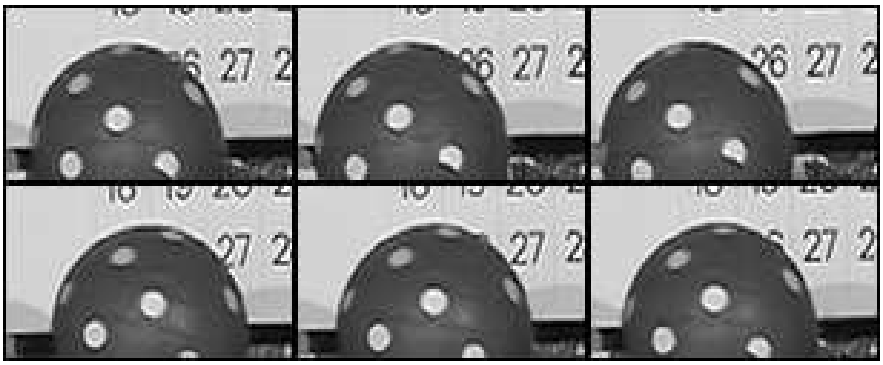


(b) Ouverture avec un élément structurant classique

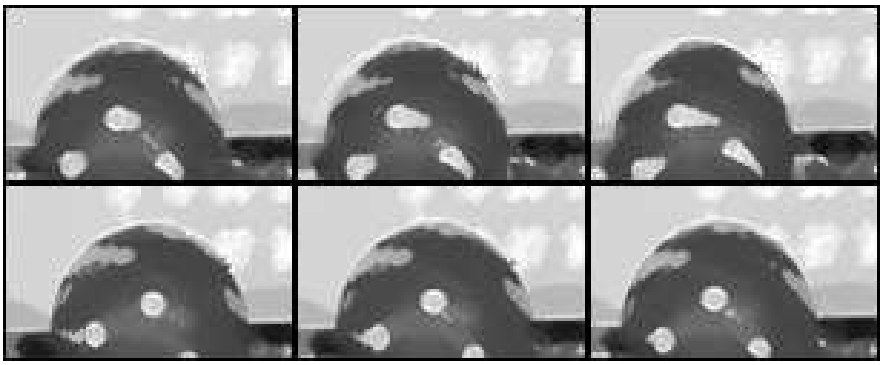


(c) Ouverture avec un ESM

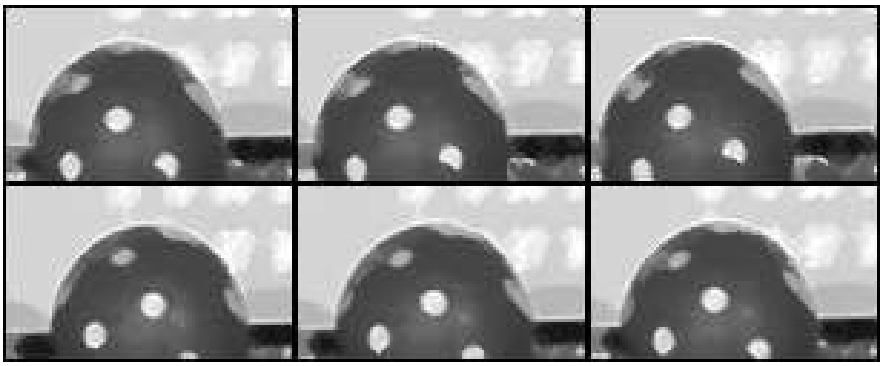
FIG. 8.7.: *Mobile* (trames 3-4) : ouverture de taille 3.



(a) Original



(b) Fermeture avec un élément structurant classique



(c) Fermeture avec un MSE

FIG. 8.8.: *Mobile* (trames 5-10, détail) : fermeture de taille 3.



FIG. 8.9.: *Foreman* (trames 1-9) : segmentation avec marqueurs — marqueurs utilisés pour la segmentation.



FIG. 8.10.: *Foreman* (trames 1-9) : segmentation avec marqueurs — ligne de partage des eaux utilisant un élément structurant classique



FIG. 8.11.: *Foreman* (trames 1-9) : segmentation avec marqueurs — ligne de partage des eaux utilisant un ESM

8.5.1.1. Invariance par changement de contraste

Une propriété de la plupart des opérateurs de morphologie mathématique plane (c'est-à-dire où l'on travaille véritablement indépendamment sur chaque ensemble de niveau) est l'invariance par changement de contraste. φ est un opérateur de contraste s'il s'agit d'une application strictement croissante d'un ensemble dans lui-même. En pratique, cela signifie que φ modifie la hauteur de chaque niveau de l'ensemble, mais ne change pas leur ordre. Nous avons donc, si φ est un changement de contraste, et ψ est un opérateur invariant par changement de constraste :

$$\varphi \circ \psi = \psi \circ \varphi$$

Les opérateurs morphologiques s'intéressent seulement à l'ordonnancement des valeurs, indépendamment des valeurs elles-même. C'est le cas de l'érosion (s'appuyant sur l'opération min), de la dilatation (s'appuyant sur l'opération max), du filtre médian, et de toutes leurs compositions. Ce n'est plus le cas des opérations comme le chapeau haut–de–forme, qui fait intervenir une différence.

L'invariance par contraste implique qu'un opérateur possédant cette propriété n'introduise pas de nouveaux niveaux dans une image. Par conséquent, si nous voulons que les opérateurs que nous définissons à l'aide d'une interpolation respectent la contrainte d'invariance par changement de contraste, il faut que le schéma d'interpolation n'introduise pas de nouvelles valeurs. C'est une forte limitation sur les types de schéma utilisables[2], puisque cela signifie qu'un tel schéma ne nous fournira jamais une fonction continue si nous raffinons jusqu'à obtenir une fonction définie sur un espace continu.

Dans le cas particulier des ESM, cette limitation est d'autant plus difficilement acceptable qu'aucune méthode d'estimation de mouvement n'a d'intérêt à utiliser un tel système d'interpolation. Un schéma comme l'interpolation au plus proche voisin équivaut à arrondir le champ à l'entier le plus proche, pour ce qui est des plans temporels voisins. Pour les suivants, il existe une diffèrence dans la mesure où nous effectuons d'abord la somme des vecteurs puis ensuite arrondissons celle-ci, au lieu de faire la somme des vecteurs arrondis. Le gain de précision engendré reste faible, d'autant que les voisinages aussi grands seront peu utilisés en pratique. Plus généralement, toujours dans le cas des ESM, il est dangereux de vouloir utiliser pour les opérations morphologiques un schéma d'interpolation différent de celui utilisé pour estimer le champ de mouvement. En effet, l'estimation du déplacement perd alors en qualité, puisque nous n'utilisons pas les valeurs qui nous ont permis de minimiser notre erreur de prédiction (ou tout autre fonctionnelle servant de critère à l'estimation du mouvement).

Dans la suite, nous abandonnons cette contrainte de l'invariance par changement de contraste qui restreint trop fortement le problème.

[2]Cette remarque est entendue suivant le type d'application recherchée. Dans le cas d'une interpolation dans laquelle la fonction se doit d'être continue, la problèmatique est différente. Nous supposons ici que les images que nous avons sont des discrétisations d'une fonction d'intensité lumineuse continue.

8.5.1.2. Préservation des extrema

Certains schémas d'interpolation introduisent des nouveaux extrema dans les fonctions. Dans le cadre de la morphologie mathématique, ceci est dangereux, pour plusieurs raisons :

- Nous avons tout d'abord un changement de dynamique de l'image, ce qui est un problème en soi.
- Nous risquons d'introduire des nouvelles composantes connexes dans certains ensembles de niveau (figure 8.12). L'arbre d'inclusion de ceux-ci risque d'être modifié. Nous aurons alors des résultats très différents pour les expériences de segmentation, mais aussi pour les fortes érosions ou dilatations.

Nous souhaitons donc conserver autant que faire se peut les extrema existants, en position et en valeur. Cela n'est possible que si une valeur interpolée est comprise dans l'intervalle maximal des valeurs de ses plus proches voisins sur la grille initiale. Les ondelettes d'interpolation sans rebonds développées au chapitre 7, si nous utilisons un encadrement d'ordre 0, font partie des systèmes satisfaisant une telle contrainte. Plus simplement, l'interpolation bilinéaire est un outil plus simple, mais tout aussi efficace dans le respect des extrema. D'autre part, ce type d'interpolation peut être utilisé en estimation de mouvement. C'est donc celui que nous utiliserons par la suite.

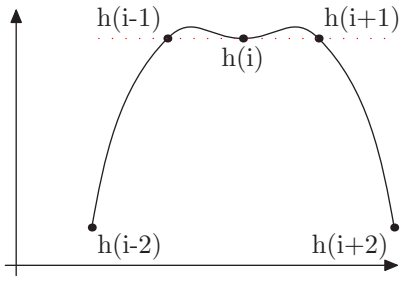


FIG. 8.12.: Création de nouvelles composantes connexes dans les ensembles de niveau lors d'une interpolation à un ordre trop élevé.

8.5.2. Interpolation directe des voisins

Cette méthode peut être vue comme l'approche naïve du problème posé : puisque la précision sous-pixellique nous donne la position exacte du correspondant du point, nous pouvons aller prendre la valeur interpolée à cette position comme valeur du voisin. L'extension de la définition de l'érosion (8.11) ne semble pas poser de problème de définition : nous allons effectivement récupérer les valeurs de la fonction en chacun des points (entiers ou non-entiers) du voisinage, puis nous effectuons un calcul qui nous donne la valeur de la fonction au point que nous traitons.

Que faire en revanche pour la "dilatation" adjointe ? La première idée qui vient est de réutiliser notre définition par l'équation (8.13). Il nous faudrait trouver pour chaque

point y de $\mathcal{I}$, les points x de $\tilde{\mathcal{I}}$ pour lesquels $y \in V(x)$. Nous avons déjà ici deux soucis :

- il ne s'agit plus seulement de définir nos voisinages sur chacun des points de la grille, mais aussi sur chacun des points de $\tilde{\mathcal{I}}$. Ce qui signifie, dans le cas des ESM, qu'il nous faut interpoler, non seulement l'image, mais aussi le champ de mouvement. Ensuite, il faut s'assurer que les points des voisinages de $\tilde{\mathcal{I}}$ sont eux-mêmes dans $\tilde{\mathcal{I}}$. Cela ne pose pas de problème dans le cas d'une interpolation bilinéaire, mais force en pratique à recalculer les valeurs interpolées à chaque fois, au lieu de pouvoir utiliser des valeurs précalculées sur une grille sur-échantillonnée, comme nous pourrions le faire si nous utilisions un champ de mouvement dont la précision est demi-entière.
- il nous faut ensuite faire une recherche pour trouver les points x. Dans le cadre des ESM, cette recherche peut être accélérée en notant avant toute chose quels carrés définis par la grille se déplacent sur quels carrés.

Mais passées ces deux difficultés, nous nous rendons compte que ceci ne définit pas une dilatation. Il est en effet possible de trouver des contre-exemples pour lesquels la propriété d'adjonction (8.1.3.3) traduite pour les fonctions en niveaux de gris :

$$g \leq \varepsilon(f) \Longleftrightarrow \delta(g) \leq f \tag{8.14}$$

n'est pas respectée. Prenons le cas où notre élément structurant est défini par un ensemble fixe de décalages $S = \{(0,0),(-1,0),(1,0),(\frac{1}{2},1)\}$ (figure 8.13(a)). Pour chaque point $\mathbf{y}$, l'ensemble des points $\mathbf{x}$ tels que $\mathbf{y} \in V(\mathbf{x})$ est $\check{S} = \{(0,0),(1,0),(-1,0),(-\frac{1}{2},-1)\}$ (figure 8.13(b). Si nous prenons une bout d'image quelconque (figure 8.13(c)), que nous lui appliquons une "érosion" s'appuyant sur l'élément structurant S (figure 8.13(d)), puis ce que nous espérions être la dilatation adjointe (figure 8.13(e)), nous obtenons une fonction pour laquelle certains points (en rouge) ne valident pas l'équation (8.14), où nous prenons $g = \varepsilon(f)$:

$$\delta(\varepsilon(f)) \leq f.$$

Une des raisons pour lesquelles nous n'avons pas obtenu la dilatation adjointe provient du passage de $\mathcal{I}$ à $\tilde{\mathcal{I}}$ et inversement. Notre "érosion" est en réalité définie comme :

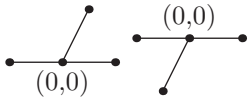
$$\varepsilon(f)(x) = \min_{y \in V(x)} \tilde{f}(y)$$

et la pseudo-dilatation comme :

$$\delta(g)(x) = \max_{x \in V(y)} \tilde{g}(y).$$

L'interpolation fausse nos définitions. La pseudo-dilatation n'utilise pas les mêmes points de $\tilde{\mathcal{I}}$ que "l'érosion". Nous avons :

$$g \leq \varepsilon(\tilde{f}) \not\Leftrightarrow \delta(\tilde{g}) \leq f. \tag{8.15}$$



(a) S (b) $\check{S}$

1	3	5	8	11	3	5	8
3	3	6	7	9	3	4	7
8	7	9	11	13	5	2	7
9	6	3	4	9	8	10	5

(c) g

·	·	·	·	·	·	·	·
·	3	3	6	3	3	3	·
·	4.5	6.5	8	5	2	2	·
·	3	3	4	4	3.5	4.5	·

(d) $\varepsilon(g)$

·	·	·	·	·	·	·	·
·	·	6	7.25	6.5	3.5	·	·
·	·	8	8	8	8	·	·
·	·	·	·	·	·	·	·

(e) $\delta_1(\varepsilon(g))$

FIG. 8.13.: δ définie par $\check{S}$ n'est pas la dilatation adjointe de ε définie avec S. Ici, $\delta_1(\varepsilon(g)) \not\leq g$ (sur les points rouges) et la propriété d'adjonction (8.14) n'est pas respectée. Les pixels marqués par un · ont une valeur indifférente pour nous, dépendant des valeurs au-delà de notre extrait de l'image.

Pour tenter d'obtenir une définition correcte de la dilatation adjointe, il nous faut revenir à la propriété d'adjonction. Définissons $\mathcal{I}_V$, l'ensemble des points touchés par les voisinages de notre grille :

$$\mathcal{I}_V = \bigcup_{x \in \mathcal{I}} V(x).$$

Nous avons :

$$\begin{aligned} g \leq \varepsilon(f) &\Leftrightarrow \forall x \in \mathcal{I}, g(x) \leq \min_{y \in V(x)} \tilde{f}(y) \\ &\Leftrightarrow \delta(g) \leq f \end{aligned}$$

Reformulons : si nous prenons une fonction g, et si nous pouvons trouver une fonction h telle que :

$$\forall x \in \mathcal{I}, \forall y \in V(x), g(x) \leq \tilde{h}(y), \tag{8.16}$$

alors cette fonction h sera telle que :

$$h \geq \delta(g).$$

L'équation (8.16) nous donne un jeu de contraintes à respecter pour les valeurs de $\tilde{h}$ et donc pour les valeurs de h. Dans le cas de l'interpolation bilinéaire, nous avons donc un jeu de contraintes linéaires sur h. Ce jeu de contraintes nous fournit un ensemble de fonctions qui seront toutes plus grandes que notre hypothétique fonction $\delta(g)$.

Nous arrivons ici à un écueil sérieux. En effet, nous pouvons avoir deux fonctions h_1 et h_2 vérifiant (8.16), mais telle que $\min(h_1, h_2)$ ne la vérifie pas (figure 8.14(a). Il semble que notre édifice s'écroule, puisqu'il existe alors des $h \geq \delta(g)$ et pour lesquelles (8.16). Nous n'avons donc pas d'adjonction possible (figure 8.14(c)). La recherche du minimum fonctionne dans le cas entier dans la mesure où toutes les contraintes sont alors des contraintes ponctuelles compatibles avec la comparaison point à point du treillis des fonctions (figure 8.14(b)).

En fait, en allant jusqu'au bout du raisonnement, nous pouvons constater que notre "érosion" n'en est pas une. Le fait déjà mentionné que le minimum de deux fonctions vérifiant une contrainte en un point non-entier ne vérifie pas nécessairement cette contrainte, contrevient à la propriété fondamentale de l'érosion de fonctions :

$$\wedge\varepsilon(h_i) = \varepsilon(\wedge h_i),$$

ce que nous n'avons pas ici (voir la figure 8.14(a)).

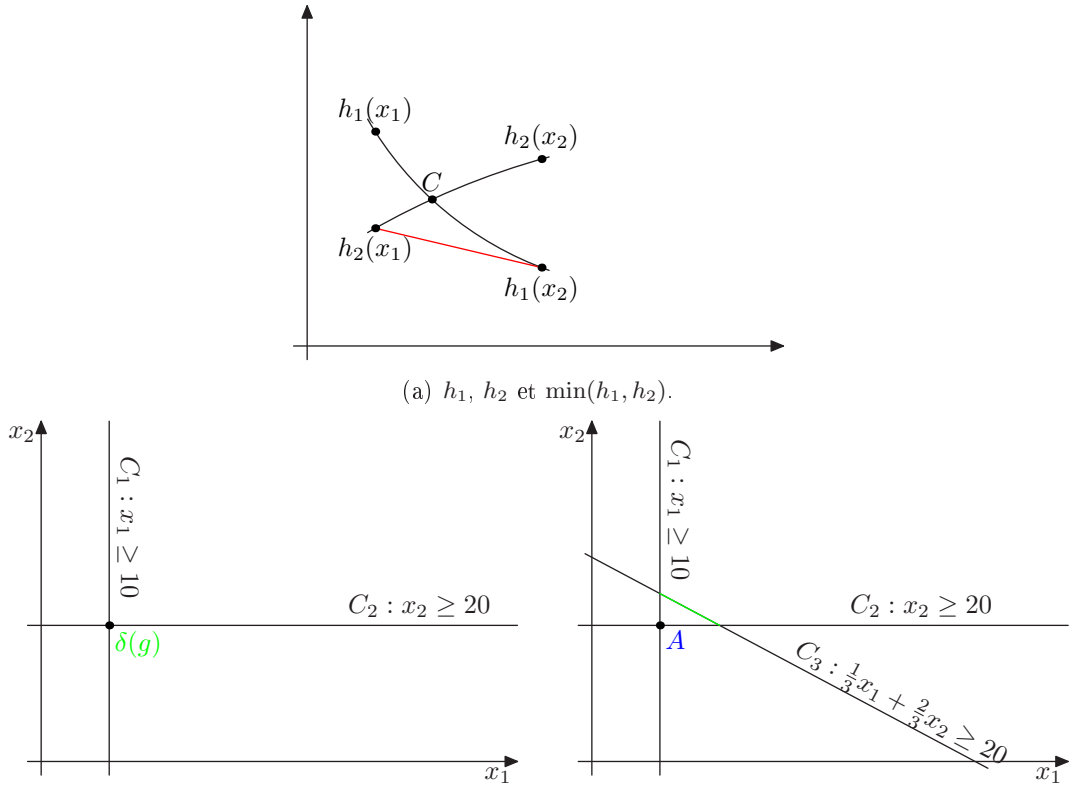
8.5.3. Sur-échantillonnage de l'image

Les problèmes auxquels nous avons dû faire face dans la section précédente tenaient essentiellement au fait que nous allions chercher des valeurs, ou tentions d'en envoyer, sur des points hors de notre grille de travail. Nous pouvons tenter d'éviter ce problème en travaillant sur tous les points d'un raffinement de notre grille initiale. Au lieu de calculer le résultat de nos opérations seulement sur les points de $\mathcal{I}$, nous pouvons nous intéresser à tous les points de $\tilde{\mathcal{I}}$, où $\tilde{\mathcal{I}}$ est une grille sur-échantillonnée dans les directions de l'espace, d'un facteur n dans chacune de ces directions. Dans le cas des ESM, ce facteur dépend de la précision que nous attribuons au champ de mouvement. En l'occurrence, si notre champ possède une précision de $\frac{1}{n}$ pixel, alors nous pouvons sur-échantillonner de ce facteur n. Si nous disposons d'un champ de précision quelconque, il nous est possible d'arrondir les vecteurs au multiple de $\frac{1}{n}$ le plus proche.

Dans la nouvelle grille, les vecteurs de mouvement deviendront à précision entière (puisqu'il faut les multiplier par n afin de préserver leur échelle). De même que précédemment, il faut interpoler puis arrondir le champ de mouvement aux nouveaux points de la grille, afin que le voisinage soit défini sur tous les points de $\tilde{\mathcal{I}}$.

Une fois que nous sommes munis de l'image sur-échantillonnée et du champ sur-échantillonné, nous nous retrouvons dans une situation classique d'utilisation des ESM où le champ est à précision entière. Il faut cependant prendre garde à faire aussi subir une homothétie spatiale d'un facteur n à l'élément structurant de base, afin de respecter le changement d'échelle. Nous avons tout intérêt à ce que l'élément structurant agrandi soit le plus proche possible de la forme sous-jacente. Il existe diverses méthodes pour arriver à cette fin, par exemple en utilisant les distances de Borgefors. Il est aussi possible plus simplement de redigitaliser la forme sous-jacente de l'élément structurant comme le montre la figure 8.15.

Nous pouvons donc effectuer la suite de traitements à partir de cette image. Néanmoins, nous ne devons pas oublier que sur-échantillonner la séquence vidéo d'un facteur n dans ses dimensions spatiales multiplie par n^2 le temps de travail pour un algorithme

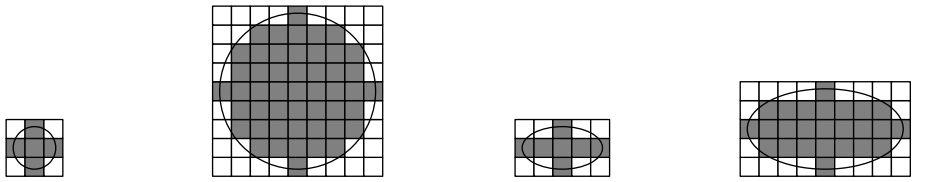


(a) h_1, h_2 et $\min(h_1, h_2)$.

(b) Contraintes sans interpolation.

(c) Contraintes avec interpolation.

FIG. 8.14.: Illustration de l'impossibilité de trouver un dilaté adjoint dans le treillis des fonctions. (a) Deux fonctions dont la valeur interpolée est au-dessus d'une valeur C. La valeur interpolée du minimum ponctuel de ces deux fonctions n'est pas au-dessus de C. (b) Recherche du dilaté en l'absence d'interpolation. Toutes les contraintes sont de la forme $h(x_i) \geq$ constante. L'infimum des fonctions vérifiant les contraintes (ici C_1 et C_2) se trouve dans l'ensemble formé par celles-ci. (c) Recherche du dilaté en présence d'interpolation. Les contraintes sont de la forme $\sum \lambda_i x_i \geq$ constante. L'infimum A des fonctions vérifiant les contraintes (ici C_1, C_2 et C_3) ne les vérifie pas. Il existe des fonctions h supérieures à A qui ne vérifient pas les contraintes (l'implication $h \geq A \Rightarrow \varepsilon(h) \geq g$ est fausse). N'importe quelles deux fonctions sur le segment vert ne peuvent être comparées entre elles, mais vérifient chacune les contraintes.



(a) Cercle, échelle 1 (b) Cercle, échelle 4 (c) Ellipse, échelle 1 (d) Ellipse, échelle 2

FIG. 8.15.: Digitalisation d'une forme sous-jacente à différentes échelles.

linéaire par rapport à la taille de l'image. Ceci peut constituer dans certains cas un sévère inconvénient par rapport à un raffinement qui n'apportera pas nécessairement une différence de qualité significative.

D'autre part, si nous souhaitons intégrer les résultats du processus à un autre module, nous devrons revenir à la taille d'origine de l'image, et donc effectuer un sous-échantillonnage. Malheureusement, celui-ci coupera selon toute vraisemblance des composantes connexes en plusieurs morceaux, ou en fera disparaître certaines. Indépendamment de la qualité de la méthode de sous-échantillonnage (qui peut elle-même s'appuyer sur des outils morphologiques, voir par exemple [13]), elle introduira nécessairement une perte d'information, information qui n'existait certes pas à l'origine de nos traitements, mais que ceux-ci ont introduite.

8.5.4. Utilisation de la partie sous-pixellique du mouvement pour déformer avec soin

Dans (8.1.3), le problème en cas de convergence du mouvement a été signalé. Ne pouvons-nous pas essayer d'utiliser la partie sous-pixellique du mouvement afin de pallier cette difficulté ?

Imaginons que n points $(\mathbf{x_i})_{1\leq i\leq n}$ $(n>1)$ soient transportés en un nombre de points m $(m<n)$ par l'opérateur de compensation utilisant le champ entier $M^+_{\mathbf{Z}}$:

$$\forall i \in [1,n],\ \mathbf{y}_{i,\mathbf{Z}} = M^+_{\mathbf{Z}}(\mathbf{x_i}) \tag{8.17}$$

Si l'on considère maintenant l'opérateur de compensation utilisant le champ à précision sous-pixellique $M^+_{\mathbf{R}}$, il n'est pas certain que les points $\mathbf{y}_{i,\mathbf{R}}$, images des $\mathbf{x_i}$ par cet opérateur soient confondus.

Définissons tout d'abord, pour chaque $\mathbf{y}_{i,\mathbf{R}}$, la suite de ses plus proches voisins dans le plan spatial $(\mathbf{y}^j_{i,\mathbf{Z}})_{j\in\mathbf{N}}$.

Au lieu de prendre dans notre élément structurant les $(\mathbf{y}^0_{i,\mathbf{Z}})$ qui sont par hypothèse confondus, nous pouvons essayer de choisir ceux associés à un des n-uplets $(j_{i,opt})_{1\leq i\leq n}$ tel que

$$i_1 \neq i_2 \Rightarrow \mathbf{y}^{j_{i_1,opt}}_{i_1,\mathbf{Z}} \neq \mathbf{y}^{j_{i_2,opt}}_{i_2,\mathbf{Z}} \qquad \forall (i_1,i_2) \tag{8.18}$$

et tel que

$$\mathbf{y}_{i,\mathbf{Z}}^{j_{i,opt}} = \text{argmin} \sum_{1 \leq i \leq n} |\mathbf{y}_{i,\mathbf{Z}}^{j_i} - \mathbf{y}_{i,\mathbf{R}}|^2. \tag{8.19}$$

Il s'agit ici d'un problème d'attribution bi-partite. Dans la mesure où notre ensemble est à priori de taille faible (9 points dans le cas d'un élément structurant 3D à 26 voisins), la complexité reste raisonnable à chaque déformation.

Le fait est que le n-uplet $(j_{i,opt})_{1 \leq i \leq n}$ n'est pas nécessairement unique, particulièrement dans le cas où la précision du champ est fractionnaire. Il est possible soit de prendre un des n-uplets au hasard, ou bien choisir de minimiser une autre caractéristique du groupe de voisins. Cela peut être la similarité par rapport à la forme de l'élément structurant 3D initial, la faiblesse de l'allongement, ou tout simplement l'utilisation d'une autre norme pour la minimisation des déplacements.

Nous devons néanmoins noter que ce schéma peut nous mener à déplacer des points de plus d'un pixel par rapport à leur véritable position. Nous perdons alors une partie de l'intérêt des ESM en terme de capacité à exploiter correctement la corrélation temporelle, puisque nous allons placer dans le voisinage des points qui ont de fortes chances de n'être plus liés au centre.

8.6. Conclusion

Nous avons mis en place des éléments structurants qui s'adaptent au mouvement présent dans les séquences vidéo afin de résoudre les problèmes d'aliasage temporel, et au-delà de ceci, afin d'exploiter au mieux la structure des "blocs vidéo". Les expériences montrent que l'objectif que nous nous étions fixés est correctement satisfait : nous avons rendu à nouveau voisines les différentes composantes des objets, et nous sommes capables, à partir des mêmes outils morphologiques, de segmenter et de filtrer notre séquence d'une façon nettement plus satisfaisante.

En revanche, il semble beaucoup plus complexe de tirer parti de la partie sous-pixellique du champ de mouvement estimé. Les différentes méthodes proposées présentent des défauts que ce soit au niveau :

- du développement théorique des outils morphologiques (8.5.2)
- du temps de calcul par rapport au bénéfice que nous pouvons espérer tirer (8.5.3) des opérations effectuées
- de l'exploitation des résultats (8.5.3)
- de la combinaison avec l'objectif de recorrélation des voisins temporels (8.5.4)

La question appelle donc de plus complètes investigations.

Les champs d'application de ces nouveaux éléments sont vastes. Ils nous semblent particulièrement adéquats pour le suivi de petites structures comme la *narine de Foreman* ou plus concrètement celui des lèvres, ce qui constituerait un premier module dans l'aide aux malentendants pour la compréhension des interlocuteurs. Le suivi des lèvres nous permettrait d'analyser l'ouverture et la forme de celles-ci, indices importants pour

connaître les sons, et donc les mots, les paroles, formés par l'interlocuteur. Il est également possible d'envisager une extension au domaine de la vidéo volumique : les médecins disposent aujourd'hui de volumes en mouvement qu'ils souhaiteraient pouvoir segmenter. En supposant que nous disposons de la méthode d'estimation de mouvement adéquate, l'utilisation des ESM pourrait alors s'étendre naturellement à ce cas, de même que la morphologie en 2 dimensions s'étend naturellement vers la morphologie en 3 dimensions.

Conclusion

Conclusion

Bilan

Dans ce mémoire de thèse, nous avons parcouru les différentes problématiques de la compression vidéo :

- estimation du mouvement
- décomposition afin d'obtenir une représentation la plus creuse possible
- quantification et codage des coefficients en utilisant leur statistique afin d'optimiser le gain de débit par rapport à la distorsion introduite

Pour chacune, nous avons développé de nouveaux algorithmes qui s'appuient sur des hypothèses sur les données à traiter :

- Dans le chapitre 3, nous avons supposé, d'une part, que le champ de mouvement était localement constant et globalement régulier, d'autre part, que la résolution de l'équation du flot optique était formellement équivalente à la minimisation de l'erreur de prédiction.
- Dans le chapitre 4, nous avons supposé que le champ de mouvement était bilinéaire par morceau et continu.
- Dans le chapitre 5, le raisonnement de Mallat et Falzon suppose un certain type de décroissance des coefficients de la décomposition en ondelettes ou par cosinus discret.
- Dans le chapitre 6, nous postulons que les discontinuités de chacune des composantes du champ de mouvement se situent aux mêmes endroits.
- Dans le chapitre 7, l'analyse de régularité suppose que les fonctions que nous traitons sont k-simples.

Le fait est que, si ces hypothèses sont vérifiées sur des données synthétiques que nous créons en les ayant en tête, elles peuvent ne pas l'être sur les données réelles. Dans ce cas, les algorithmes qui en découlent sont naturellement moins performants et peuvent devenir moins efficaces que des algorithmes plus simples déjà existants. Il faut donc trouver un juste équilibre entre la généralité des hypothèses de travail et leur capacité à justifier les propriétés des algorithmes. L'expérience du chapitre 3 nous a appris à nous assurer que l'optimisation de l'obtention d'un jeu de données doit se réaliser avec en tête l'utilisation qui en sera faite.

Par ailleurs, l'expérience obtenue au travers des algorithmes développés appelle une autre réflexion : la sophistication d'un modèle ou d'un algorithme se doit de lui apporter une différence significative par rapport à l'équivalent plus simple. Dans le cas contraire,

il est possible que la difficulté supplémentaire pour trouver les paramètres optimaux finit par annuler, si ce n'est rendre négatif, l'intérêt de l'enrichissement du modèle. C'est le cas par exemple pour l'estimation du champ bilinéaire avec une taille de bloc fine qui souffre de contraintes supplémentaires par rapport au block-matching. Celles-ci l'empêchent de bénéficier de la résolution quasi-exhaustive utilisée pour le block-matching. À l'opposé, une des raisons du succès de l'adaptation de l'algorithme de Mallat et Falzon au cas de la vidéo par rapport, semble-t-il, à l'algorithme de He et Kim, est sa simplicité : elle permet d'analyser plus aisément le rôle du paramètre et de pouvoir le corriger si besoin. En comparaison, les travaux de He et Kim fournissent un jeu de constantes dont l'influence est difficilement intelligible.

Les résultats positifs des éléments structurants suivant le mouvement confortent l'idée qu'un algorithme reposant sur peu de paramètres et sur un modèle simple (deux points reliés par un vecteur de mouvement sont mieux corrélés que deux points voisins temporellement) a plus de chances d'atteindre ses objectifs de façon générale d'un algorithme plus complexe bâti autour d'un modèle seulement partiellement vérifié comme dans le cas de la méthode d'estimation du chapitre 4.

Contributions

Dans ce mémoire de thèse, nous avons établi une boîte à outils pour la compression :

- Nous avons tout d'abord montré l'intérêt d'introduire un minimum de régularité dans l'estimation du mouvement lorsque le champ obtenu doit être utilisé en compression vidéo. En effet, la transmission de vecteurs de mouvement précis a un coût, dont il faut s'assurer qu'il n'est pas supérieur au gain que nous pouvons espérer faire en utilisant des vecteurs précis plutôt que des vecteurs approximatifs.
- Nous avons ensuite développé une méthode d'obtention des valeurs des points de contrôle dans un champ de mouvement de type CGI. Cette méthode de résolution multi-échelles donne des résultats tantôt meilleurs, tantôt moins bons que le block-matching classique dans le cas de champs de mouvement dont les paramètres sont épars.
- Nous avons validé l'utilisation du modèle liant pas de quantification et débit établi par Mallat et Falzon dans le cas de la compression des images fixes au cas de la compression vidéo.
- Par l'étude des possibilités offertes par le schéma de *lifting*, nous avons introduit deux nouvelles formes de décompositions non-linéaires visant à répondre à des problèmatiques particulières. Les ondelettes sans rebonds, en particulier, atteignent l'objectif de limiter les effets d'oscillation autour des discontinuités, au moins dans le cas des données synthétiques.

Enfin, nous avons développé une nouvelle variété d'éléments structurants pour la morphologie mathématique. Ces éléments s'appuient sur une vision intuitive de l'idée de voisinage dans un espace dont une des dimensions est le temps et, d'une part, apportent une amélioration significative aux résultats visuels de filtrage, d'autre part, permettent d'effectuer des segmentations "volumiques" au travers de la séquence vidéo.

Perspectives

Les pistes qui s'ouvrent suite à ce travail sont variées :

- Comme nous l'avons mentionné, l'adaptation du modèle de Mallat et Falzon au codeur H.264 n'est pas immédiate si nous partons du travail effectué sur le codeur de H.263. Néanmoins, il doit être possible de réfléchir en séparant en deux passes le codage : une première passe afin d'estimer le mouvement pour la séquence qui s'appuie sur un paramètre de quantification a priori, puis une seconde passe où nous utilisons un système similaire à celui de H.263.
- Les résultats d'approximation à N termes pour les ondelettes sans rebonds montrent que l'étape de mise-à-jour non-linéaire n'est pas aussi productive pour ces ondelettes que peut l'être l'étape de mise-à-jour linéaire. Il y a donc un travail à effectuer sur les opérateurs de mise-à-jour non-linéaires afin qu'ils se combinent harmonieusement et productivement avec l'étape de prédiction qui a fait ses preuves.
- L'intérêt de la formulation des ondelettes sans rebonds 2D à l'aide des médians d'intervalle reste à évaluer, à la fois vis à vis des ondelettes linéaires, mais aussi par rapport aux ondelettes sans rebond 2D définies en symétrisant le schéma du *lifting*.
- La nature des champs utilisés pour tester le schéma de *lifting* joint ne validait pas l'hypothèse qui était faite. Il est possible que des effectués avec des champs de mouvement issus d'autres méthodes d'estimation soient plus probants. Selon toute vraisemblance, la validation ou non de l'hypothèse dépend en bonne partie de la forme de la modélisation du champ, selon un lien à établir.
- Les travaux théoriques réalisés pour utiliser la partie sous-pixellique du mouvement au sein des ESM n'ont pour l'instant rien donné de probant. Le tour de la question n'est cependant pas terminé
- Si nous pouvons bien comprendre le sens des opérations de filtrage 2D+t, ou bien la segmentation temporelle, la signifcation exacte d'opérations moins courantes munies des ESM n'est pas évidente. C'est le cas par exemple de la tranformée en tout ou rien. Ce travail d'analyse permettrait de définir de nouveaux algorithmes de traitement pour des situations particulières.

Bibliographie

[1] G. Charith K. Abhayaratne et Henk Heijmans : A novel morphological subband decomposition scheme for 2d+t wavelet video coding. *In International Symposium on Image and Signal Processing and Analysis*, pages 239–244, Rome, Italie, Septembre 2003.

[2] Vincent Agnus : *Segmentation spatio-temporelle de séquences d´images par des opérateurs de morphologie mathématique*. Thèse de doctorat, Université Louis Pasteur, Strasbourg, Octobre 2001.

[3] Yiannis Andreopoulis, Adrian Munteanu, Joeri Barbarien et M. van der Schaar : In-band motion compensated temporal filtering. *Signal Processing : Image Communication, special issue on Subband/Wavelet Interframe Video Coding*, 19(7):653–673, Août 2004.

[4] J. L. Barron, D. J. Fleet et S. S. Beauchemin : Performance of optical flow techniques. *International Journal of Computer Vision*, 12(1):43–77, Février 1994.

[5] Christophe Bernard : *Ondelettes et problèmes mal posés : la mesure du flot optique et l'interpolation irrégulière*. Thèse de doctorat, Ecole Polytechnique, Novembre 1999.

[6] H. Brusewitz : Motion compensation with triangles. *In Proc. 3rd Int. Conf. on 64 kbit Coding of Moving Video*, Rotterdam, Pays-Bas, Septembre 1990.

[7] Robert Calderbank, Ingrid Daubechies, Wim Sweldens et Boon-Lock Yeo : Wavelet transforms that map integers to integers. Rapport technique, Department of Mathematics, Princeton University, 1996.

[8] Andrea Cavallaro, Olivier Steiger et Touradj Ebrahimi : Tracking video objects in cluttered background. *IEEE Transactions on Circuits and Systems for Video Technology*, 15(4):575–584, Avril 2005.

[9] P. Chen et John Woods : Bi-directional mc-ezbc with lifting implementation. *IEEE Transaction on Circuits and Systems for Video Technology*, 14(10):1183–1194, Octobre 2004.

[10] S. Choi et John Woods : Motion-compensated 3-d subband coding of video. *IEEE Transactions on Image Processing*, 3(2):155–167, Février 1999.

[11] Roger L. Claypoole, Goeffrey M. Davis, Wim Sweldens et Richard G. Baraniuk : Nonlinear wavelet transforms for image coding. *In Proceedings of the 31st Asilomar Conference on Signals, Systems and Computers*, volume 1, pages 662–667, Pacific Grove, USA, Novembre 1997.

[12] Ingrid Daubechies et Wim Sweldens : Factoring wavelet transforms into lifting steps. Rapport technique, Department of Mathematics, Princeton University, 1996.

[13] Étienne Decencière et Michel Bilodeau : Downsampling of binary images using adaptive crossing numbers. *In* Christian Ronse, Laurent Najman et Étienne Decencière, éditeurs : *Mathematical Morphology : 40 Years On, Proc. ISMM'05*, pages 279–288, Paris, France, Avril 2005. Springer.

[14] Cristina Gomila et Fernand Meyer : Tracking objects by graph matching of image partition sequences. *In 3rd IAPR-TC15 Workshop on Graph-Based Representations in Pattern Recognition*, pages 1–11, Ischia, Italie, Mai 2001.

[15] Christophe Gratin : *De la représentation des images au traitement morphologique d´images tridimensionnelles.* Thèse de doctorat, École Nationale Supérieure des Mines de Paris, 1993.

[16] Frédéric Guichard et Jean-Philippe Morel : Image analysis and p.d.e.

[17] Zhihai He : *ρ-domain rate-distortion analysis and rate control for visual coding and communication.* Thèse de doctorat, University of California, Santa-Barbara, Juin 2001.

[18] Zhihai He, Yong Kwan Kim et Sanjit K. Mitra : Low-delay rate control for dct video-coding via ρ-domain source modeling. *IEEE Transactions on Circuits and Systems for Video Technology*, 11(8):928–940, 2001.

[19] Zhihai He et Sanjit K. Mitra : A unified rate-distortion analysis framework for transform coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 11(12):1221–1236, 2001.

[20] Henk Heijmans et John Goutsias : Multiresolution signal decomposition schemes. part 2 : Morphological wavelets. *In 4672*, page 60. Centrum voor Wiskunde en Informatica (CWI), ISSN 1386-3711, 30 1999.

[21] Berthold K. P. Horn et Brian G. Schunk : Determining optical flow. *Artificial Intelligence*, 17(1-3):185–203, Août 1981.

[22] S.-T. Hsiang et John Woods : Embedded video coding using invertible motion compensated 3-d subband/wavelet filter bank. *Signal Processing : Image Communication*, 16:705–724, May 2001.

[23] Nicolas Laveau et Richard Beare : Object tracking and segmentation using region growing. *In APRS/IEEE Workshop on Stereo Image and Video Processing*, pages 11–14, Sydney, Australie, 2000. AIPR.

[24] Nicolas Laveau et Christophe Bernard : Structuring elements following the optical flow. *In* Christian Ronse, Laurent Najman et Étienne Decencière, éditeurs : *Mathematical Morphology : 40 Years On, Proc. ISMM'05*, pages 13–22, Paris, France, Avril 2005. Springer.

[25] Nicolas Laveau et Christophe Bernard : Structuring elements following the optical flow. *Computer Vision and Image Understanding*, soumis.

[26] Romain Lerallut, Étienne Decencière et Fernand Meyer : Image filtering using morphological amoebas. *In* Christian Ronse, Laurent Najman et Étienne Decencière, éditeurs : *Mathematical Morphology : 40 Years On, Proc. ISMM'05*, pages 13–22, Paris, France, Avril 2005. Springer.

[27] Bruce D. Lucas et Takeo Kanade : An iterative image registration technique with an application to stereo vision. *In Seventh International Joint Conference on Artificial Intelligence*, pages 674–679, 1981.

[28] Stéphane Mallat : *Une exploration des signaux en ondelettes*. Les Éditions de l'École Polytechnique, 2000.

[29] Stéphane Mallat et Frédéric Falzon : Analysis of low bitrate image transform coding. *IEEE Transactions on Signal Processing*, pages 1027–1042, Avril 1998.

[30] Michael W. Marcellin, Michael J. Gormish, Ali Bilgin et Martin P. Boliek : An overview of JPEG-2000. *In Proceedings of IEEE Data Compression Conference*, pages 523–544, Mars 2000.

[31] Ferran Marqués, Beatriz Marcotegui et Fernand Meyer : Tracking areas of interest for content-based functionalities in segmentation-based video coding. *In International Conference on Acoustics, Speech and Signal Processing, ICASSP'96*, Atlanta, USA, Mai 1996.

[32] Étienne Mémin et Patrick Pérez : Dense estimation and object-based segmentation of the optical flow with robust techniques. *IEEE Transactions on Image Processing*, 7(5):703–719, 1998.

[33] Y. Nakaya et H. Harashima : An iterative motion estimation method using triangular patches for motion compensation. *In Proc. of SPIE Visual Comm. Image Processing*, volume 1605, pages 546–557, Boston, USA, Novembre 1991.

[34] Y. Nakaya et H. Harashima : Motion compensation based on spatial transformations. *IEEE Transactions on Circuits and Systems for Video Technology*, 4(6):339–356, Juin 1994.

[35] Jens-Rainer Ohm : Motion grid interpolation with simple contour adaptation. *In Proc. Picture Coding Symposium*, Melbourne, Mars 1996.

[36] Montse Pardàs et Philippe Salembier : Time-recursive segmentation of image sequences. *In EUSIPCO-94*, Edinburgh, Septembre 1994.

[37] Gemma Piella et Henk Heijmans : Adaptive lifting schemes with perfect reconstruction. *IEEE Transactions on Signal Processing*, 50(7):1620–1630, Juillet 2002.

[38] William H. Press, Saul A. Teukolsky, William T. Vetterling et Brian P. Flannery : *Numerical Recipes in C*. Cambridge University Press, 1992.

[39] Jordi Ribas-Corbera et Shawmin Lei : Rate control in dct video coding for low-delay communications. *IEEE Transactions on Circuits and Systems for Video Technology*, 9(1):172–185, 1999.

[40] Jordi Ribas-Corbera et D. L. Neuhoff : Optimizing block size in motion-compensated video coding. *J. Electron. Imaging*, 7:155–165, Janvier 1998.

[41] Amir Said et William A. Pearlman : A new fast and efficient image codec based on st partitioning in hierarchical trees. *IEEE Transactions on Circuits and Systems for Video Technology*, 6:243–250, Juin 1996.

[42] Philippe SALEMBIER, Ferran MARQUÉS, Montse PARDÀS, Josep Ramon MORROS, Isabelle CORSET, Sylvie JEANNIN, Lionel BOUCHARD, Fernand MEYER et Beatriz MARCOTEGUI : Segmentation-based video coding system allowing the manipulation of objects. *IEEE Transactions on Circuits and Systems for Video Technology*, 7(1): 60–74, Janvier 1997.

[43] A SECKER et D. S. TAUBMAN : Highly scalable video compression using a lifting-based 3d wavelet transform with deformable mesh motion compensation. volume 3, pages 749–752, Juin 2002.

[44] Jean SERRA : *Image Analysis and Mathematical Morphology - Volume I.* Academic Press, 1982.

[45] Eero P. SIMONCELLI, Edward H. ADELSON et David J. HEEGER : Probability distributions of optical flow. *In Proceedings 1991 of IEEE Conference on Computer Vision and Pattern Recognition*, pages 310–315, Maui, USA, Juin 1991.

[46] Christoph STILLER et Janusz KONRAD : Estimating motion in image sequences. *IEEE Signal Processing Magazine*, pages 70–91, Juillet 1999.

[47] Gary SULLIVAN et R. L. BAKER : Motion compensation for video compression using control grid interpolation. *In Proceedings of IEEE ICASSP'91*, pages 2713–2716, Los-Angeles, USA, 1991.

[48] Gary J. SULLIVAN : Efficient scalar quantization of exponential and laplacian random variables. *IEEE Transactions on Information Theory*, 42(5):1365–1374, Septembre 1996.

[49] Wim SWELDENS : The lifting scheme : A construction of second generation wavelets. *SIAM J. Math. Anal.*, 29(2):511–546, 1997.

[50] D. WANG : Unsupervised video segmentation based on watersheds and temporal tracking. *IEEE Transactions on Circuits and System for Video Technology*, 8(9):539–546, Septembre 1998.

[51] H. WATANABE et S. SINGHAL : Windowed motion compensation. *In Proceedings of the SPIE Conference on Visual Communication and Image Processing*, volume 1605, pages 582–589, Boston, USA, Novembre 2001.

[52] Francisca ZANOGUERA, Beatriz MARCOTEGUI et Fernand MEYER : A segmentation pyramid for the interactive segmentation of 3-d images and video sequences. *In Mathematical Morphology and its Applications to Image Processing, Proc. ISMM'00*, pages 263–272, Palo Alto, USA, Juin 2000. Kluwer Ac. Publ., Nld.