

Thèse

présentée en vue de
l'obtention du titre de

Docteur

de

**L'école nationale supérieure
de l'aéronautique et de l'espace**

ÉCOLE DOCTORALE: Informatique et télécommunications

SPÉCIALITÉ: Signal et image

par

Emmanuel CHRISTOPHE

**Compression des Images Hyperspectrales et son
Impact sur la Qualité des Données**

Soutenue le 20 octobre 2006 devant le jury:

Francis CASTANIÉ	Président
Pierre DUHAMEL	Co-directeur de thèse
James FOWLER	Rapporteur
Christine GUILLEMOT	Rapporteur
Catherine LAMBERT	Rapporteur
Dimitri LEBEDEF	
Dominique LÉGER	
Corinne MAILHES	Co-directrice de thèse

*À force de croire en ses rêves,
l'Homme en fait une réalité.*

Georges Rémi

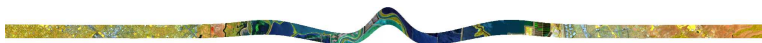
Remerciements

Cette page est peut-être la plus difficile à écrire de toute la thèse tant ceux qui ont eu une influence sur ce travail sont nombreux. . .

Je tiens d'abord à remercier ceux qui ont suivi ces travaux de près pendant trois ans. J'ai eu la chance d'avoir un grand nombre d'encadrants qui se sont tous impliqués, ont tous contribué aux résultats présentés ici. Ils ont formé une équipe formidable et j'ai pris énormément de plaisir à travailler dans ces conditions. Je commencerai par mes deux directeurs de thèse. **Corinne Mailhes** a toujours été disponible pour mes questions, enthousiaste pour mes résultats et indulgente pour mes 's' égarés. **Pierre Duhamel** m'a apporté des méthodes solides, un œil critique et des discussions productives malgré la distance. Ce tandem de direction a trouvé un équilibre difficile et réussi une alchimie stimulante. **Dominique Léger** a été avec moi au jour le jour pendant mes premiers pas dans le monde de la recherche à l'Onera, il a su me guider à la fois dans le domaine complexe des critères de qualité, mais aussi dans le monde subtil des règles d'orthographe. **Carole Thiebaut** m'a tenu informé des derniers développements dans le monde de la compression embarquée et m'a impliqué dans des études industrielles. **Christophe Latory** m'a permis de découvrir le système de compression SPOT5 et toutes les contraintes qui régissent les images satellites. **Dimitri Lebedeff** a donné la caution industrielle d'Alcatel Space à ces travaux, il a toujours été disponible et très impliqué malgré l'éloignement. Enfin **Bill Pearlman** m'a encadré pendant les deux mois passés au Rensselaer Polytechnic Institute, bénéficier de ses conseils et suggestions a été une expérience très enrichissante. Sans eux, cette thèse n'aurait pas vu le jour.

Christine Guillemot, **Catherine Lambert** et **Jim Fowler** m'ont fait l'honneur d'évaluer ces travaux. Je tiens à les remercier pour le temps passé à lire ce manuscrit et leurs judicieuses remarques ayant permis d'apporter des améliorations. Je souhaite spécialement remercier Catherine pour avoir été à l'origine de ce sujet de thèse, mais également pour avoir rendu possible la coopération aux États-Unis, ainsi que Jim pour QccPack, un outil souple et performant.

Francis Castanié a accepté de présider ce jury. Je souhaite également le remercier pour m'avoir accueilli au TéSA et pour avoir créé cette structure



de recherche qui a facilité bien des démarches administratives.

J'ai eu la chance de pouvoir travailler pendant trois ans dans des endroits différents, cette diversité a été enrichissante. Le département du DOTA à l'Onera m'a d'abord accueilli. Je remercie tous ceux que j'ai cotoyé quotidiennement pendant cette année : Sophie (qui a du me supporter pendant un an), Nicolas, Xavier, Hervé, Sandrine, Keyvan, Dimitri, Xavier, Yannick, Françoise, Veronique, Philippe, Christophe, Joël, Laurent, Bernard, Bruno, Jean-Pierre, Jean-Michel, François. Une période de transition m'a permis de partager (trop) brièvement le bureau de Nicolas dans *l'ancien* TésA. Enfin j'ai eu la chance de passer plus de la moitié de ma thèse dans le *bureau de la culture* (dixit Patrice) en compagnie de Lucile, Ferdinand et Patrice. Je remercie toute l'équipe du Tésa pour les discussions autour du café matinal : Marie-Jo, Manu, Anchalee, Bernard, Mariana, Vincent, David, Xavier, Jean-Yves, Anne, Tai, Lucile, Wilfried, Maria, Tan ainsi que Marie, Nathalie, Vincent et Florent. Je n'oublie pas non plus Sebastien, Charles, Mustafa, Lotta, Liz, Tara et tous les autres de RPI.

Mes parents m'ont donné cette curiosité et ce goût pour les sciences. Je n'écrirai probablement pas cette page si ils n'avaient pas encouragé ces argumentations interminables autours de la table du diner avec mes frères et sœurs. Une pensée aussi pour mon grand-père qui m'a fait découvrir les Sciences (avec un 'S') et ses problèmes éthiques.

Enfin, *last but not least*, un grand merci à **Sharon** mon épouse. Pendant ces années, elle m'a toujours encouragé, soutenu, même à plus de 10000 km de distance. Les voyages autours du monde que j'ai été *obligé* de faire et les longues vacances que j'ai été *forcé* de prendre étaient des pauses indispensables. Un message stéganographié lui est destiné tout au long de ce mémoire ;)

Table des matières

	Guide de lecture	1
1	Contexte	5
	1.1 Les contraintes de l'hyperspectral.	6
	1.1.1 Pourquoi faire des images hyperspectrales?.	6
	1.1.2 Acquisition des images hyperspectrales	10
	1.1.3 Espace et satellites	12
	1.2 Propriétés des images hyperspectrales	15
	1.2.1 Des dimensions aux propriétés différentes	15
	1.2.2 Problème de la normalisation	18
	1.2.3 Un espace presque vide	19
	1.2.4 Une estimation de l'entropie	19
	1.3 Compression	22
	1.3.1 Notions de base	22
	1.3.2 État de l'art	26
	1.3.3 Applications	28
2	Critères qualité pour les images hyperspectrales	29
	2.1 Taxinomie des critères qualité	30
	2.1.1 Définition	30
	2.1.2 Intérêt pour la compression.	31
	2.1.3 Méthodes d'évaluation	31
	2.1.4 Les méthodes bivariantes	32
	2.1.5 Extension isotrope de critères de traitement du signal	33
	2.1.6 Critères orientés hyperspectral.	35
	2.1.7 Critères statistiques évolués.	39
	2.1.8 Critères issus de la théorie de l'information	41
	2.2 Démarche proposée pour l'évaluation des critères	43
	2.2.1 Applications de référence.	44
	2.2.2 Images hyperspectrales testées	47
	2.2.3 Dégradations appliquées	48
	2.2.4 Démarche	49



2.3	Résultats de l'étude	50
2.3.1	Présentation des résultats	50
2.3.2	Interprétation graphique et sensibilité des applications.	51
2.3.3	Interprétation des critères de qualité	54
2.3.4	Évaluation quantitative	55
2.3.5	Critères de qualité proposés.	58
2.4	Conclusion	59
3	Compression des images hyperspectrales.	61
3.1	JPEG 2000 pour l'hyperspectral	62
3.1.1	La norme JPEG 2000	62
3.1.2	Adaptation aux images hyperspectrales	63
3.2	Choix de la décomposition optimale.	66
3.2.1	Décomposition anisotropique 3D en ondelettes	66
3.2.2	Optimisation débit-distorsion	69
3.2.3	Résultats sur les images 2D.	74
3.2.4	Décomposition en ondelettes optimale pour l'hyperspectral	76
3.2.5	Décomposition fixe	76
3.3	Structures d'arbres	78
3.3.1	Idées principales	78
3.3.2	Principes généraux de EZW et SPIHT	81
3.3.3	Étude statistique pour le choix de la structure d'arbre.	82
3.3.4	Adaptation de EZW	86
3.3.5	Adaptation de SPIHT	89
3.4	Comparaison des résultats	92
3.5	Variations avec la notation binaire signée	95
3.5.1	Un inconvénient de EZW	95
3.5.2	Utilisation de la notation binaire signée	95
3.5.3	Utiliser les dépendances spatiales	98
3.6	Conclusion	100
4	Vers une plus grande flexibilité	103
4.1	Décomposition et arbre utilisés	108
4.2	Le codage par groupes	111
4.2.1	Pourquoi?	111
4.2.2	Comment?	111
4.3	Permettre la progression en résolution	113
4.3.1	Introduction de la progression	113
4.3.2	Illustration sur un exemple	115
4.3.3	Permutation des progressions	117
4.4	Les désavantages du codage par groupes.	119
4.4.1	Conservation de la progression en qualité	119
4.4.2	Organisation en couches et débit-distorsion.	120
4.4.3	Connaître la distorsion pendant la compression	121
4.4.4	La formation du train binaire final	122

	4.5 Résultats	123
	4.5.1 Données	123
	4.5.2 Performances de compression	124
	4.5.3 Flexibilité	126
	4.5.4 Codage de régions d'intérêt	127
	4.6 Conclusions	129
5	Retour sur les critères qualité	131
	5.1 Représentation des critères qualité	132
	5.1.1 Représentation	132
	5.1.2 Robustesse par rapport à l'amplitude de la dégradation	132
	5.1.3 Robustesse par rapport à l'image utilisée	132
	5.2 Validation	135
	5.2.1 Une distance entre dégradations	135
	5.2.2 Variation de niveau de dégradation (1er cas)	136
	5.2.3 Dégradation inconnue (2ème cas)	137
	5.2.4 Image différente (3ème cas)	137
	5.2.5 Intérêt par rapport au PSNR	138
	5.3 Impact de SPIHT-RA	140
	5.4 Aller plus loin	140
	Conclusion	143
A	DWT	145
	A.1 Implémentation générale	145
	A.1.1 Approche convolutive	145
	A.1.2 Approche <i>lifting</i>	146
	A.2 La transformée entière	147
	A.3 Notation des sous-bandes	148
B	Exemples pour EZW et SPIHT	151
	B.1 Déroulement sur EZW	151
	B.2 Déroulement sur SPIHT	153
C	Utilisation de JPEG2000	155
	C.1 Correction préliminaire	155
	C.2 Décorrélacion interbandes par DWT	156
	C.3 Décorrélacion interbandes par KLT	156
	C.4 Utilisation de Kakadu	157
D	Données hyperspectrales	159
	D.1 Images utilisées	159
	D.2 Format des données	159
E	Instruments	165
	E.1 Démarche	165
	E.2 Les capteurs actuels	166
	E.2.1 AVIRIS	166
	E.2.2 CASI, CASI-2	168
	E.2.3 CASI-3	170

E.2.4	MIVIS	171
E.2.5	Hymap	172
E.2.6	Hyperion	174
E.2.7	CHRIS-PROBA	175
E.2.8	COIS-NEMO	177
E.2.9	Spectra	179
E.2.10	Autres capteurs	180
E.2.11	Résumé	181
	Bibliographie	182

Table des figures

1.1	Vers l'hyperspectral	6
1.2	Diverses bandes spectrales	7
1.3	Points chauds	8
1.4	Cube hyperspectral	9
1.5	Termes utilisés	9
1.6	Principe d'acquisition par un capteur <i>pushbroom</i>	10
1.7	Effet de smile	12
1.8	Effet de keystone	12
1.9	Corrélation	16
1.10	Corrélation spectrale	17
1.11	Profil spectral usuel	17
1.12	Entropie d'une source à deux états	21
1.13	Schéma de compression par transformée	23
1.14	Arbre de Huffman	25
2.1	Spectres pour la détection	45
2.2	Moffett2	47
2.3	Moffett3	47
2.4	Moffett4 Classification	48
2.5	Moffett3 Classification	48
2.6	Méthode de Brill	50
2.7	MAD vs. SAM	52
2.8	F_λ vs. SAM	52
2.9	MSE vs. détection	53
2.10	MSE vs. ACP	53
2.11	MAE vs. Mahalanobis	55
2.12	PMAD vs. Mahalanobis	55
2.13	RRMSE vs. SAM	56
2.14	$Q_{(x,y)}$ vs. SAM	56
3.1	Décorrélacion pour JPEG2000	65
3.2	Décomposition multirésolution classique	67
3.3	Exemple de décomposition anisotropique.	68



3.4	Décomposition anisotropique et notations.	69
3.5	Décomposition multirésolution	70
3.6	Décomposition anisotropique	71
3.7	Calcul de la courbe R-D	72
3.8	Best-basis	73
3.9	Décomposition optimale (Lena)	75
3.10	Décomposition optimale (Barbara)	75
3.11	Performances de la décomposition optimale	76
3.12	Images utilisées	77
3.13	Visualisation des décompositions	79
3.14	Performances de la décomposition anisotropique	80
3.15	Principe des arbres de zéros	81
3.16	Arbre de zéros de degré 0	83
3.17	Arbre de zéros de degré 1	83
3.18	Arbre de zéros de degré 2	83
3.19	Descendance dans le cas isotropique	87
3.20	Descendance dans le cas anisotropique	87
3.21	Ordre de parcours des coefficients pour EZW.	88
3.22	Tree-crossing	88
3.23	Terminologie SPIHT pour les descendants	90
3.24	Descendance de LL pour EZW et SPIHT	91
3.25	Comparaison JPEG 2000, EZW-3D et SPIHT-3D	93
3.26	Performances sur diverses images	94
3.27	Probabilités selon les contextes	99
3.28	Performances de 3D-EZW-NAF	100
4.1	Barbara et détail.	104
4.2	Barbara et codage progressif en qualité	105
4.3	Barbara et codage progressif en résolution	106
4.4	Barbara et résistance aux erreurs	107
4.5	Structure d'arbre utilisée	108
4.6	Groupes	109
4.7	Descendance par groupe	112
4.8	Numérotation des résolutions	113
4.9	Illustration des codages progressifs	115
4.10	Ordre de parcours	117
4.11	Train binaire pour une progression en résolution	118
4.12	Train binaire pour une progression en qualité	118
4.13	Organisation progressive en résolution avec en-têtes	119
4.14	Organisation progressive en qualité avec en-têtes	119
4.15	Choix des points de coupures	121
4.16	Train binaire final	121
4.17	Extraction de différente résolution du même train binaire	127
4.18	Décodage de régions d'intérêt	129

5.1	Bruit blanc sur moffett4	133
5.2	Flou spatial sur moffett4	133
5.3	Flou spectral sur moffett4	133
5.4	Flou spatial et spectral sur moffett4	133
5.5	Gibbs sur moffett4	134
5.6	JPEG 2000 sur moffett4	134
5.7	Bruit blanc sur diverses images	134
5.8	JPEG 2000 sur diverses images	134
5.9	Flou spatial sur diverses images	135
5.10	Flou spectral sur diverses images	135
5.11	Comparaisons pour PSNR=63 dB	139
5.12	Comparaison JPEG2000-SPIHT-RA	141
A.1	Décomposition en ondelettes : filtrage	145
A.2	Notation des sous-bandes pour multirésolution	149
A.3	Notation des sous-bandes pour multirésolution (généralisée)	149
B.1	Exemple de coefficients	151
D.1	Railroad Valley pour la détection	160
D.2	Moffett Field par Aviris	161
D.3	Moffett Field par Hyperion	161
D.4	Hawaï par Aviris	162
D.5	Harvard Forest par Aviris	162
D.6	Jasper Ridge par Aviris	163
D.7	Cuprite par Aviris	163
D.8	BIP, BSQ ou BIL	164

Liste des tableaux

1.1	Spécification d'un capteur hyperspectral	14
1.2	Exemple des spécifications d'une mémoire pour satellite	15
1.3	Estimation de l'entropie	21
1.4	Débit en bits par pixel par bande selon l'ordre des données.	22
1.5	Codage d'Huffman : code original	24
1.6	Codage d'Huffman : code réduit	25
2.1	Sensibilité des applications	54
2.2	Sensibilité des critères	57
2.3	Sensibilité des critères	57
2.4	Sensibilité des critères	58
3.1	Surface de silicium nécessaire pour une KLT	65
3.2	Exemple de codage en plan de bits.	82
3.3	Statistiques pour une structure d'arbre 3D.	85
3.4	Statistiques pour une structure d'arbre spatial.	85
3.5	Statistiques pour une structure d'arbre spectral.	86
3.6	Performance de EZW	89
3.7	Performances de SPIHT	92
3.8	Effet de la simplification de EZW	95
3.9	Représentations binaire signée pour 349	97
3.10	Augmentation de la proportion de 0	97
3.11	Perf. de EZW sans <i>refinement pass</i>	98
3.12	Perf. EZW avec NAF	99
3.13	Perf. 3D-EZW et 3D-EZW-NAF	99
4.1	Données utilisées	124
4.2	Impact des modifications sur les performances	124
4.3	Temps de compression	125
4.4	Débit (bpppb) en codage sans pertes.	125
4.5	PSNR pour différents débits pour Moffett sc3.	126
4.6	RMSE pour différents débits pour Moffett sc3.	126
4.7	MAD (ou Emax) pour différents débits pour Moffett sc3.	126



4.8	Flexibilité au décodage	128
5.1	Distances pour un bruit blanc 150	137
5.2	Distances pour un flou spectral 4	137
5.3	Distances pour JPEG 2000 1 bpppb	138
5.4	Distances pour bruit blanc sur moffett4	139
5.5	Distances pour JPEG 2000 sur moffett4	139
5.6	Distances pour SPIHT-RA à 0.5 bpppb	141
5.7	Distances pour SPIHT-RA à 0.2 bpppb	141
A.1	Filtres utilisés pour la 9/7.	146
A.2	Filtres utilisés pour la 5/3.	146
B.1	Déroulement de la première passe de EZW	152
B.2	Déroulement de la première passe de SPIHT	154

Guide de lecture

SUIVANT l'évolution générale des technologies, l'observation de la Terre par les satellites fournit des données de plus en plus précises, détaillées, mais aussi complexes. Au cœur de cette thèse sont les images hyperspectrales. Leur apparition, liée à l'amélioration des capteurs et à la volonté de voir toujours plus depuis l'espace, crée de nouvelles problématiques. La quantité importante d'information générée par ces capteurs, la forte redondance entre ces informations et l'importance des détails requiert un système de compression adapté.

Le **premier chapitre** présente les prérequis pour comprendre le contexte de la compression des images hyperspectrales. Le mode d'acquisition de ces images est expliqué, ce qui permet d'en comprendre leur origine. Les contraintes spécifiques du traitement embarqué sur satellite sont également expliquées, ces limitations vont peser sur le choix de l'algorithme de compression. On étudie ensuite les propriétés statistiques des images hyperspectrales. En effet, il est important de bien connaître ces propriétés pour en tirer parti lors de la compression. Enfin, les bases des techniques de compression sont exposées et le choix d'une compression par transformée justifié par rapport à une compression par quantification vectorielle.

Le modèle de compression choisi va introduire des dégradations sur les images hyperspectrales. L'intérêt de ces images résidant dans leurs détails, il faut s'assurer que ces dégradations ne mettent pas en péril les applications qui utilisent ces données. Le **second chapitre** détermine des critères de qualité adaptés aux images hyperspectrales. Un grand nombre de critères plus ou moins classiques sont adaptés aux images hyperspectrales. Une démarche est définie pour déterminer les propriétés de ces critères, en terme de sensibilité spécifique à un type de dégradation. Enfin, cinq critères sont finalement conservés pour leur bonne complémentarité.

Cette étude sur les critères de qualité ayant permis de se familiariser un peu plus avec les propriétés des images hyperspectrales, on peut maintenant étudier la compression dans le **troisième chapitre**. D'abord le standard récent JPEG 2000, toujours en cours de définition pour certaines parties, est appliqué aux images hyperspectrales. Différentes méthodes de décorrélation interbandes sont appliquées pour finalement retenir une transfor-



mée en ondelettes en raison d'un bon compromis performances/complexité. JPEG 2000, utilisé dans une configuration difficilement compatible avec les contraintes du spatial, mais présentant d'excellentes performances, sert de référence pour les performances que nous proposons dans cette thèse. Dans un premier temps, nous nous intéressons à la recherche d'une décomposition en ondelettes 3D optimale en termes de débit-distorsion pour les images hyperspectrales. Nous montrons l'apport d'une telle décomposition anisotrope pour la compression des images hyperspectrales. Comme une telle décomposition est dépendante de l'image et du débit visé, ce qui pose des problèmes d'implémentation, nous proposons une décomposition anisotrope fixe, assurant des performances proches de la décomposition optimale pour une large variété d'images hyperspectrales. Dans un deuxième temps, nous étudions le codage des coefficients d'ondelettes obtenus après la décomposition proposée. Les méthodes de compression par arbres de zéros donnent des bons résultats sur les images 2D tout en gardant une complexité raisonnable. On adapte donc ces méthodes, EZW et SPIHT, aux images hyperspectrales. Ainsi, on compare différentes structures d'arbres possibles en fonction de leur capacité à regrouper les zéros. Enfin, les résultats sont présentés pour diverses images, démontrant l'intérêt des méthodes de compression proposées qui approchent celles de JPEG2000, avec un coût calculatoire plus raisonnable. Pour conclure, une utilisation originale de la notation binaire signée est proposée pour tirer parti au mieux des propriétés des arbres de zéros. Cette innovation conduit à proposer une version simplifiée et parallélisable de EZW sans pertes majeures de qualité.

L'évolution des algorithmes de compression ne se fait pas nécessairement seulement vers une augmentation brute des performances. La flexibilité de ces algorithmes est aussi un point important. Une adaptation est donc faite dans le **quatrième chapitre** pour permettre l'accès aléatoire à une partie déterminée de l'image et pour permettre également de décoder des versions à basse résolution en spatial et en spectral tout en lisant un nombre minimum de bits dans le train binaire. On détaille donc le rassemblement des coefficients d'ondelettes correspondant à la même zone de l'image par groupes pour permettre l'accès aléatoire et par la même occasion pour introduire une certaine résistance aux erreurs. Une allocation de débit doit être réalisée entre les différents groupes pour garder une progression en qualité. La séparation des différentes résolutions pour permettre le décodage progressif en résolution ainsi que les problèmes posés par cette organisation sont expliqués ensuite. Enfin, les résultats et la flexibilité de cet algorithme sont illustrés sur des images hyperspectrales.

Finalement, pour vérifier l'impact de l'algorithme de compression sur la qualité des images, une méthode originale est définie dans le **cinquième chapitre** pour utiliser les cinq critères de qualité sélectionnés. Cette méthode est ensuite validée numériquement sur différents exemples. Elle montre des performances satisfaisantes pour identifier le type de dégradation et l'im-

peut être probable sur une application. La compression définie montre ainsi un diagramme d'erreur très semblable à celui de JPEG 2000 tout en gardant une erreur maximale plus faible et un coût calculatoire moindre.

Chapitre 1

Contexte

1.1 Les contraintes de l'hyperspectral.	6
1.1.1 Pourquoi faire des images hyperspectrales ?	6
1.1.2 Acquisition des images hyperspectrales	10
1.1.3 Espace et satellites	12
1.2 Propriétés des images hyperspectrales	15
1.2.1 Des dimensions aux propriétés différentes	15
1.2.2 Problème de la normalisation	18
1.2.3 Un espace presque vide	19
1.2.4 Une estimation de l'entropie	19
1.3 Compression	22
1.3.1 Notions de base	22
1.3.2 État de l'art	26
1.3.3 Applications	28



HYPERSPECTRAL, compression et contraintes de l'espace, ce chapitre présente le contexte de cette étude qui porte sur la compression des images hyperspectrales. Dans un premier temps, nous verrons ce que sont les images hyperspectrales, la manière dont elles sont acquises et les contraintes spécifiques liées au traitement des données embarquées à bord des satellites. Nous étudierons ensuite leurs particularités ainsi que les problèmes posés par leur nature. Les notions de bases nécessaires pour comprendre les principes de la compression d'images seront rappelées, et enfin nous dresserons un état de l'art de la compression des images hyperspectrales.

1.1 Les contraintes de l'hyperspectral

1.1.1 Pourquoi faire des images hyperspectrales ?

L'observation de la Terre depuis l'espace s'est d'abord faite en monochrome, principalement dans le domaine militaire avec une récupération physique des films. Les capteurs électroniques présentant des avantages évidents pour la récupération des images se sont très vite développés. Il est ensuite apparu que l'observation de la même scène à plusieurs longueurs d'onde permettait une meilleure exploitation des données. Un très grand nombre de capteurs multispectraux se sont ainsi développés, le premier étant Landsat au début des années 70. Pour permettre la formation d'images aux couleurs naturelles, l'observation est alors souvent faite dans les trois couleurs usuelles (rouge, vert et bleu) dans le proche infrarouge. Cette dernière bande est particulièrement intéressante pour l'observation des végétaux qui ont une réponse spectrale forte dans ce domaine à cause de la présence de chlorophylle.

L'évolution naturelle des capteurs d'images a conduit à l'acquisition non pas d'une, trois ou quatre bandes spectrales mais plutôt de plusieurs cen-

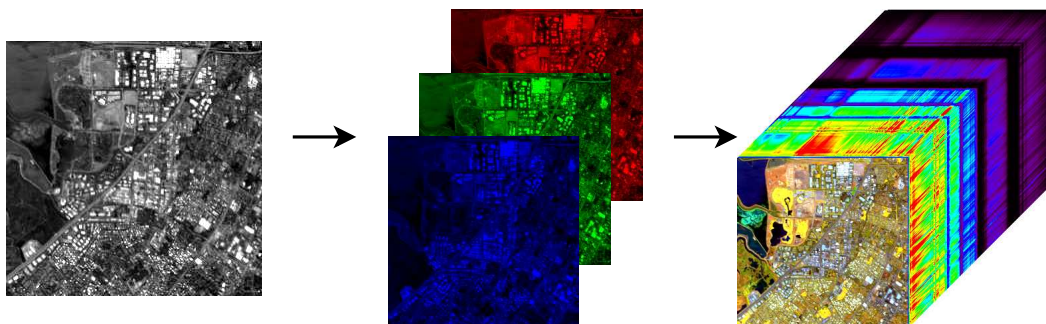


Fig. 1.1 Du monochrome à l'hyperspectral.

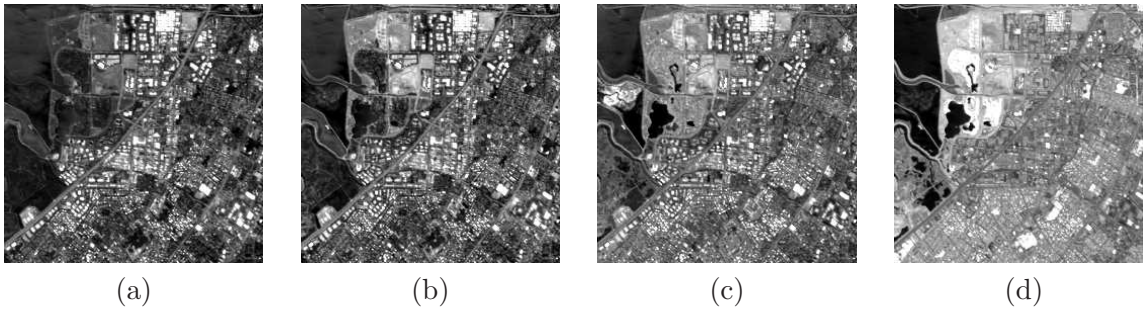


Fig. 1.2 Même scène dans diverses bandes spectrales (Moffett Field) à 458 nm (a), 664 nm (b), 712 nm (c) et 1211 nm (d). Il existe une grande ressemblance entre les images, mais les différences contiennent beaucoup d'information.

taines (Fig. 1.1).

L'ajout de bandes spectrales permet d'augmenter le pouvoir discriminant des données acquises (Fig. 1.2). On peut ainsi arriver à différencier deux matériaux possédant une couleur identique à l'œil. Par exemple, une peinture verte et une feuille qui ont la même couleur, i.e. la même réponse spectrale dans le rouge, le vert et le bleu, ne pourront pas être différenciées à l'œil, l'ajout d'autres bandes spectrales permettront de faire la différence. Les images hyperspectrales tirent parti de ces propriétés. Sur la figure 1.3, rien ne retient l'attention dans l'observation des données en couleurs naturelles. En revanche, en regardant la bande infrarouge, on remarque immédiatement un point chaud qui est confirmé sur le terrain par la présence de cheminées en activité.

Une image hyperspectrale est obtenue grâce à un spectro-imageur. L'acquisition d'une même scène est réalisée dans plusieurs bandes spectrales. L'évolution entre les capteurs multispectraux et hyperspectraux suit la même logique qu'entre les capteurs monochromes et multispectraux. La différence par rapport aux images multispectrales tient au nombre important de bandes (100 à 200), à leur largeur fine (10 à 20 nm) et au fait qu'elles soient contiguës. Cette dernière propriété permet une reconstruction du spectre de chaque pixel : on réalise en fait un échantillonnage du spectre. Cet échantillonnage doit être assez fin pour permettre une bonne reconstruction. Les techniques dites de *Full Spectral Imaging* (FSI) devraient conduire de plus en plus à cette vision spectrale des données [Bol03]. Les sondeurs atmosphériques actuels arrivent à plusieurs milliers de bandes (2378 pour l'instrument AIRS de la NASA) on parle alors d'ultra-spectral, mais ces capteurs ne sont pas encore imageurs. Certains instruments, comme Hymap, ne conservent pas les valeurs dans les bandes d'absorption de l'eau qui sont situées dans l'infrarouge (autour de 1.4 et 1.9 μm), mais comme ces bandes contiennent plus de bruit que de signal, on peut en faire abstraction.

Les données hyperspectrales sont donc acquises selon trois dimensions :

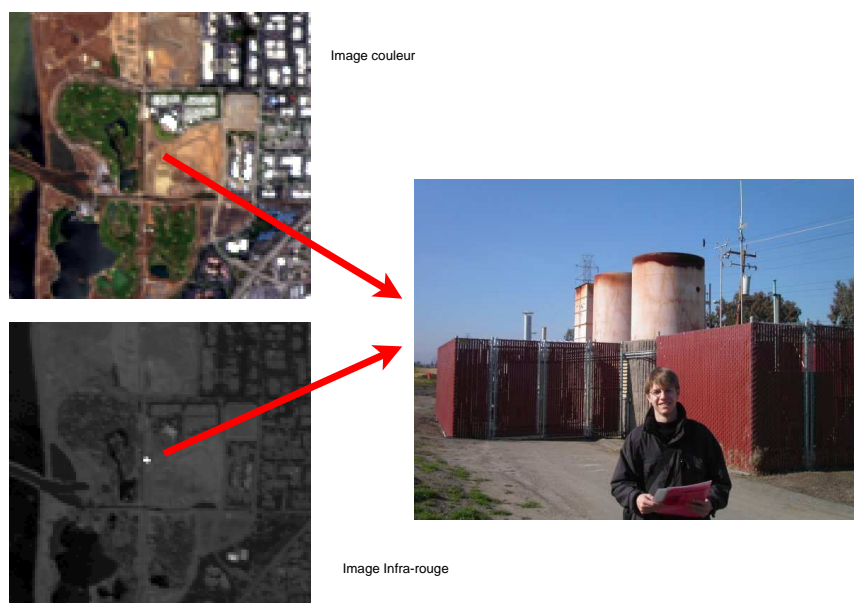


Fig. 1.3 L'observation des données dans une bande infrarouge permet de détecter la présence de *points chauds* qui étaient invisibles dans le domaine visible.

deux spatiales et une spectrale. Une représentation possible de ces données peut être faite sous la forme d'un cube hyperspectral (Fig. 1.4). La face supérieure du cube correspond à la scène spatiale, souvent une composition colorée de trois bandes spectrales. Toutes les scènes pour les différentes longueurs d'onde sont ensuite empilées pour donner le cube. Les autres faces du cube représentent alors respectivement les luminances, selon les longueurs d'onde (λ), des lignes (x) et des colonnes (y) en bordure du cube. Sur la droite de la figure, sont représentés des exemples de spectres des pixels de cubes extraits de l'image totale. L'abscisse correspond aux longueurs d'onde et l'ordonnée aux luminances.

Sur la figure 1.5 (le cube de données ayant tourné pour avoir la scène spatiale sur l'avant du cube), sont détaillés les termes utilisés pour préciser la position des pixels dans une image hyperspectrale. On notera $I(x, y, \lambda)$ la valeur sur la colonne x (on parle également d'échantillon ou *samples*), la ligne y et dans la bande spectrale λ .

Les conditions d'acquisition de ces images ainsi que les contraintes spécifiques du spatial vont être précisées ci-après, mais on peut déjà entrevoir que ces images hyperspectrales constituent un flot de données considérable à transmettre. Le but de cette thèse est de proposer une méthode de compression.

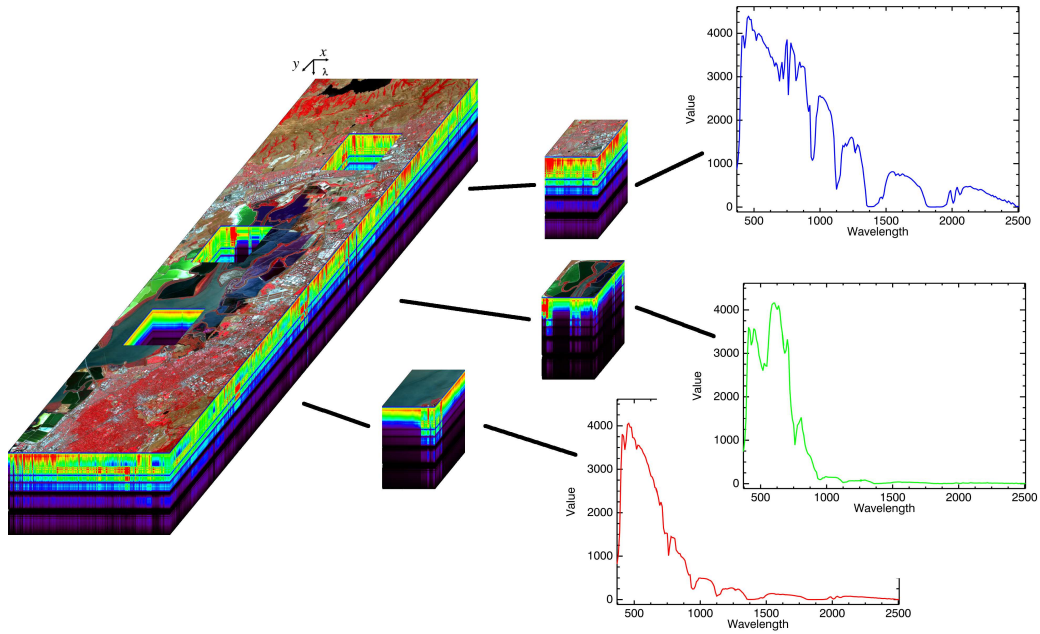


Fig. 1.4 Données hyperspectrales.

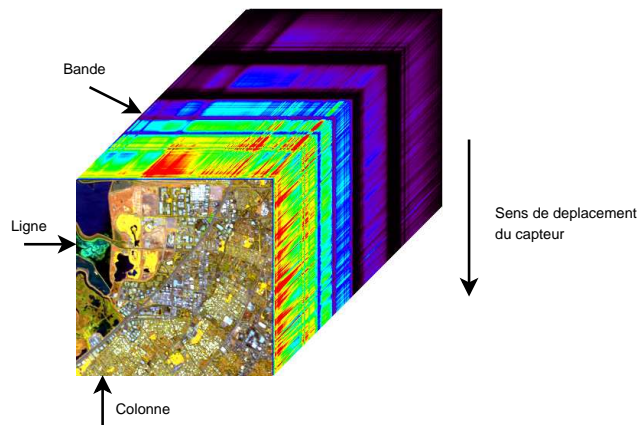


Fig. 1.5 Termes utilisés pour la position des pixels dans le cube hyperspectral.

1.1.2 Acquisition des images hyperspectrales

1.1.2.1 Types de capteurs

Il existe principalement deux types de capteurs hyperspectraux : *whiskbroom* et *pushbroom*. Pour échantillonner le signal reçu selon la longueur d'onde, les deux utilisent un élément dispersif : parfois un prisme, mais le plus souvent un réseau. La différence se situe au niveau de l'acquisition d'une ligne (au sens de la figure 1.5). Le *whiskbroom* possède une barrette CCD, où chaque élément acquiert une longueur d'onde différente. L'acquisition d'une ligne complète se fait par un système de miroir mobile pour imager une portion différente de la fauchée (largeur de l'image) à différents instants. Le capteur *pushbroom* possède une matrice CCD où une des dimensions correspond aux différentes longueurs d'ondes et l'autre à la fauchée de l'instrument (Fig. 1.6). L'acquisition des différentes lignes se fait dans les deux cas par déplacement du capteur.

L'inconvénient du *whiskbroom* est qu'il y a une partie mobile (le miroir), ce système est moins fiable dans un contexte de système spatial. Le *pushbroom* est plus robuste, mais produit en général des images rayées car les différentes colonnes de la matrice CCD n'ont pas toutes la même sensibilité. Ce défaut peut être corrigé en partie par étalonnage. La tendance actuelle pour les capteurs spatiaux est clairement orientée vers les capteurs *pushbroom*.

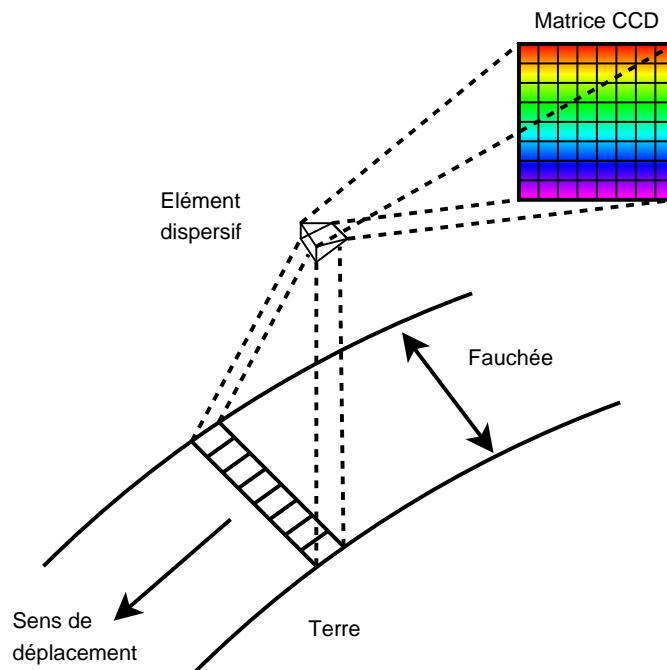


Fig. 1.6 Principe d'acquisition par un capteur *pushbroom*.

1.1.2.2 Dégradations typiques sur les données

Bruit radiométrique : Comme tous les capteurs, les instruments hyperspectraux sont affectés par le bruit présent dans le signal. La valeur du rapport signal sur bruit (RSB ou Signal to Noise Ratio, SNR) permet déjà de donner une première caractérisation du bruit. Les rapports, assez faibles pour les premiers instruments hyperspectraux (de l'ordre de 50 : 1 en linéaire, soit 17 dB, pour les premières version d'AVIRIS), atteignent maintenant des valeurs plus importantes pour les capteurs aéroportés (Annexe E.2.1 : environ 600 : 1 soit 28 dB pour AVIRIS). Les premiers instruments spatiaux présentent des valeurs plus faibles (Annexe E.2.6 : de l'ordre de 150 : 1 soit 22 dB pour Hyperion).

Cependant, cette valeur seule n'est pas suffisante pour caractériser le bruit. Simmons et Brower [Sim97] ont notamment étudié la corrélation du bruit entre les différentes bandes pour les capteurs HYDICE et AVIRIS.

Dérégistration spatiale entre bandes spectrales : Les images hyperspectrales sont a priori fortement comprimables. En effet, les différentes bandes spectrales sont très liées, la corrélation spectrale est donc importante. Cette corrélation requiert toutefois que les bandes soit *registrées*, c'est-à-dire que les différentes bandes soit superposables spatialement. Un même pixel sur toutes les bandes correspondra alors à la même zone physique observée.

Le problème se pose surtout pour les images multispectrales dont la dérégistration peut être de l'ordre d'un pixel. Dans une étude réalisée par Alcatel Space [Fra01], il a été montré que les performances de la compression multispectrale sont sensibles à la dérégistration de l'image. Le gain de compression par rapport à une compression mono-spectrale devient quasiment nul lorsque la dérégistration atteint la valeur de 0.5 pixels.

Dans le cas des images hyperspectrales, les technologies utilisées pour l'acquisition des images sont différentes : acquisition simultanée et par le même capteur des différentes bandes spectrales. Le problème de la dérégistration est donc moins important. Dans les spécifications du capteur SPECTRA, on peut voir par exemple que la dérégistration pour les différentes bandes d'un même spectromètre est inférieure à 0.2 pixels.

Autres dégradations dues au capteur : D'autres effets, principalement dus à la conception, peuvent affecter les instruments hyperspectraux. Le premier de ces effets est un effet appelé *smile*. Cet effet est provoqué par une déformation de la projection de la fente image sur la matrice CCD. L'effet au niveau de l'image se traduit par une différence d'étalonnage selon les points de l'image. Les pixels situés au centre sont normalement étalonnés, tandis qu'un *décalage spectral* apparaît sur les bords de l'image. Cet effet est illustré sur la figure 1.7.

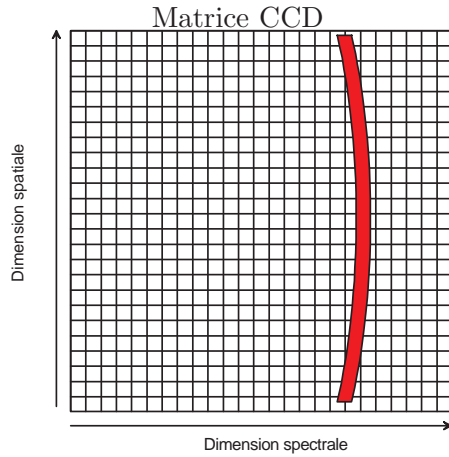


Fig. 1.7 Effet de *smile* : une ligne de l'image a une projection déformée sur le CCD

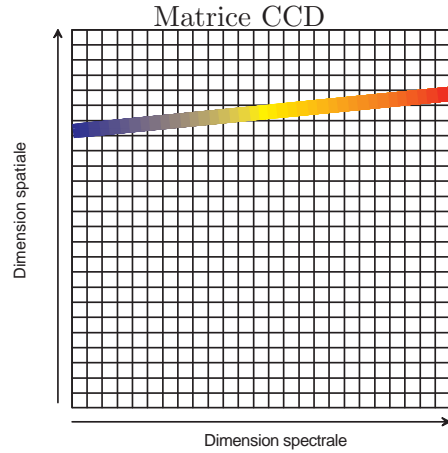


Fig. 1.8 Effet de *keystone* : la projection sur le CCD des bandes spectrales d'un même pixel est oblique

Le deuxième effet, appelé *keystone*, est dû à une inclinaison de l'image d'un point sur la matrice de capteur comme indiqué sur la figure 1.8. Cet effet provoque une contamination des valeurs d'un pixel par les pixels voisins en fonction des longueurs d'onde.

La dernière version du capteur aéroporté CASI est spécifiquement optimisée pour réduire ces distorsions. D'autre part, les techniques de conception actuelles pour les instruments hyperspectraux éliminent quasiment les problèmes de *smile* et de *keystone* [Bol03]. On ne prendra donc pas ces effets en compte.

Dégradations après acquisition : Les post-traitements appliqués aux données causent également des dégradations irréversibles. C'est notamment le cas de la quantification, dont l'utilisation est inhérente aux données numériques.

La compression, lorsqu'elle n'est pas *sans-pertes*, cause aussi des dégradations sur les données.

1.1.3 Espace et satellites

Les contraintes sur un système de compression embarqué à bord d'un satellite sont très fortes. Il y a tout d'abord une limitation sur la complexité des circuits électroniques utilisables en environnement spatial, des contraintes spécifiques sur l'acquisition des données et leur stockage.

1.1.3.1 Circuits électroniques pour le spatial

Compte-tenu du coût de la mise en orbite d'un satellite en orbite basse (de l'ordre de 15000 euros/kg), il y a des contraintes fortes sur le poids de l'électronique embarquée. Cette contrainte de poids est encore plus forte dans le cas des sondes lointaines. Les traitements coûteux sont effectués au sol dans la mesure du possible. La compression est nécessaire pour permettre de transmettre le plus d'information possible au sol. Dans ce cas, il n'y a pas de choix possible et le module de compression doit être embarqué.

L'énergie disponible à bord d'un satellite est également un facteur limitant. La consommation des circuits électroniques est donc un paramètre qui entre fortement en ligne de compte au moment de la conception. Les circuits dédiés de type ASIC permettent de réduire cette consommation, mais augmentent les coûts de développement.

L'espace est un environnement hostile pour les composants électroniques. Les particules ionisantes et les ions lourds qui, sur Terre, sont en grande partie arrêtés par l'atmosphère, atteignent directement les composants électroniques. Le piégeage de ces particules dans les semi-conducteurs altère leurs caractéristiques [CNE05] : apparition d'événements singuliers, SEU (Single Event Upset), ou destruction du composant, SEL (Single Event Latch-up).

Un SEU correspond à un changement de parité dans un composant mémoire ou dans un registre interne sans dommage significatif de l'élément. Dans les circuits actuels, le passage d'un ion peut affecter plusieurs bits. Ces erreurs multiples (MBU : Multiple Bit Upsets) sont plus difficiles à détecter et à corriger.

Un SEL est un phénomène qui apparaît avec certaines technologies (CMOS). Le passage de l'ion provoque un court-circuit (*latch-up*) ayant un effet destructif sur le composant.

Des stratégies spécifiques de développement sont mises en place pour réduire ces risques : sélection des composants les moins sensibles, durcissement aux radiations, méthodes de détection des *latch-up*, redondance des composants, ...

Les composants électroniques homologués pour une utilisation spatiale sont relativement peu nombreux et présentent des performances bien plus faibles que d'autres composants sur le marché.

1.1.3.2 Acquisition au fil de l'eau et régulation de débit

Un satellite permet d'observer la Terre en continu. C'est un des principaux avantages de l'observation spatiale par rapport à l'observation aéroportée. Lors d'une telle observation, le flot de données en sortie de l'instrument est ininterrompu. Il faut donc commencer à comprimer les données sans connaître la fin de l'acquisition : une compression *au fil de l'eau* est donc nécessaire [Par03]. Le flux de données binaires issu de la compression est

ensuite transmis au sol ou stocké dans des mémoires lorsque les stations de réception ne sont pas en visibilité.

Les débits d'entrée et de sortie de ces mémoires de masse est fixe. Le débit disponible pour transmettre les informations est également constant. Si le compresseur ne fournit pas un débit constant, il faut mettre en place des mécanismes complexes de régulation de débit.

1.1.3.3 Taille des images

Les données hyperspectrales sont volumineuses. Observer la même scène dans environ 200 longueurs d'onde multiplie logiquement la taille des données par 200. Le capteur spatial Hyperion acquiert en 3 secondes une scène de 7.5×19.8 km, ce qui représente 256×660 pixels dans chacune des 242 bandes spectrales. Comme ces données sont quantifiées sur 12 bits, cela représente environ 490 Mbits pour une résolution spatiale de 30 mètres seulement. Le capteur aéroporté AVIRIS présente des caractéristiques similaires. Le tableau 1.1 présente les spécifications typiques pour un capteur spatial. Les données précises concernant un plus grand nombre de capteurs peuvent être trouvées dans l'annexe E.

Spectre	400-2500 nm
Résolution spatiale	20 m
Nombre de bandes	200
Résolution spectrale	10 nm
Quantification	12 bits
Fauchée	20 km

Tab. 1.1 Spécification typique pour un capteur hyperspectral spatial (2004)

Le débit disponible pour transmettre les informations au sol est d'environ 105 Mbits/s en orbite basse, beaucoup moins pour les sondes lointaines. Comme la tendance est à une augmentation de la résolution, tant spectrale que spatiale, une compression efficace des données est indispensable.

1.1.3.4 Visibilité des stations et stockage bord

Les satellites d'observation hyperspectraux sont généralement des satellites à défilement en orbite basse. La transmission des données vers une station de réception n'est pas possible en permanence : il faut attendre d'être en visibilité d'une station. En attendant d'avoir cette visibilité, les données sont stockées à bord (Tab. 1.2).

La compression à bord est plus problématique que la compression au sol. En cas d'erreur, il n'est pas possible de revenir en arrière et les pertes sont irréversibles. Les contraintes qui pèsent sur les algorithmes de compression bord sont donc plus strictes.

Composant	IBM 16 Mbits DRAM
Capacité	90 Gbits (fin de vie)
Taux d'erreur binaire	$< 10^{-12}$
Masse	73 kg
Volume	61 l
Débit (écriture)	3×50 Mbps
Débit (lecture)	2×50 Mbps
Consommation (rétention)	61 W (max)
Consommation (lecture/écriture)	187 W (max)

Tab. 1.2 Exemple des spécifications d'une mémoire pour satellite (ici pour SPOT 5).

1.2 Propriétés des images hyperspectrales

Avant de s'intéresser à la compression des images hyperspectrales, il faut bien comprendre leurs propriétés statistiques. Le fait de construire un cube avec des dimensions aux propriétés différentes (axe spectral ou spatial) va introduire des spécificités statistiques sur les données.

1.2.1 Des dimensions aux propriétés différentes

Une première manière de voir les données est de considérer chaque valeur indépendamment des autres, chaque valeur correspondant à la luminance d'un pixel pour une longueur d'onde donnée. On considère alors les trois directions comme équivalentes.

En réalité, les propriétés de ces valeurs sont différentes selon les directions spectrale et spatiales. Dans les directions spatiales, la corrélation est forte à faible distance et décroît rapidement quand le décalage augmente (Fig. 1.9 (a)). Au contraire, la corrélation spectrale est présente pour tout le spectre (Fig. 1.9 (b)). Les propriétés statistiques sont donc différentes selon la direction considérée.

La matrice de corrélation spectrale, dont l'utilité dans l'analyse des données hyperspectrales a été montrée par Simmons et Brower [Sim97], représente le facteur de corrélation entre les différentes bandes spectrales. La figure 1.10 présente deux exemples de cette matrice sur deux images différentes (le détail de ces images est donné dans l'annexe D). Le coefficient (i, j) de cette matrice de corrélation représente la corrélation entre les bandes i et j et est défini par

$$r(i, j) = \frac{\sum_{x=1}^{n_x} \sum_{y=1}^{n_y} I_c(x, y, \lambda_i) I_c(x, y, \lambda_j)}{\sqrt{\sum_{x,y} I_c(x, y, \lambda_i)^2} \sqrt{\sum_{x,y} I_c(x, y, \lambda_j)^2}}, \quad (1.1)$$

$I_c(x, y, \lambda_i)$ correspondant à une variable centrée (donc de moyenne nulle) et n_x et n_y étant le nombre de colonnes et de lignes de l'image.

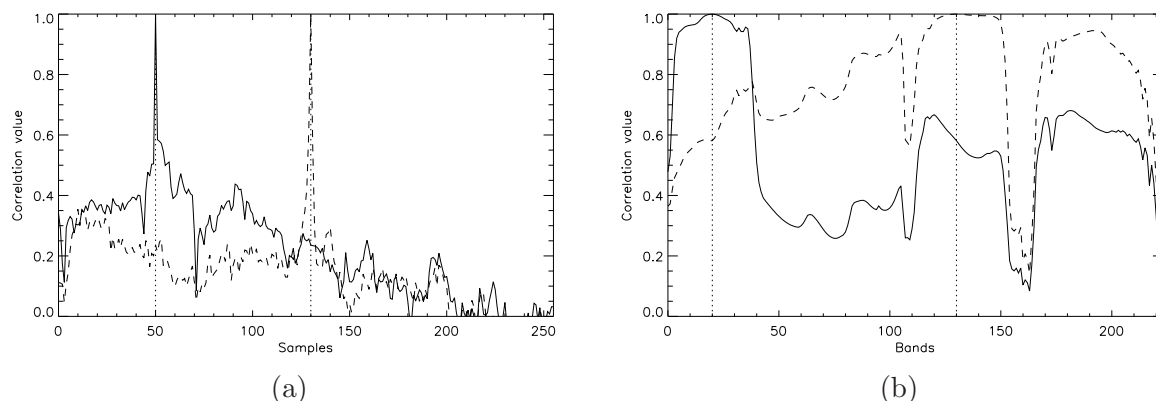


Fig. 1.9 Comparaison des coefficients de corrélation spatiale (a) et spectrale (b) en fonction du décalage entre les lignes ou entre les bandes. (a) *ligne continue* : corrélation entre la colonne 50 et les autres ; *ligne pointillée* : corrélation entre la colonne 130 et les autres. (b) *ligne continue* : corrélation entre la bande spectrale 20 et les autres ; *ligne pointillée* : corrélation entre la bande spectrale 130 et les autres. On remarque que la corrélation spectrale est beaucoup plus forte.

Le coefficient de corrélation se situe donc dans l'intervalle $[-1; 1]$, une valeur de 1 indiquant une égalité entre les deux bandes. Les valeurs étant en fait fortement corrélées, il y a rarement apparition de coefficients négatifs en pratique.

Pour expliquer la corrélation spectrale, on peut remarquer que le profil spectral présente des caractéristiques indépendantes du type du terrain observé. La forme générale du spectre présente des traits communs à toutes les zones observées. Cela est dû à l'influence de l'atmosphère (bandes d'absorption) et à l'éclairement solaire, dont l'énergie est maximale dans le domaine du visible et s'atténue dans le domaine de l'infrarouge. Ces caractéristiques modulent fortement le spectre reçu au niveau du capteur (spectre en luminance et non en réflectance).

La plus grande partie de l'énergie recueillie par le capteur se trouve dans le domaine 400-900 nm. Pour les longueurs d'ondes comprises dans le domaine d'absorption de la vapeur d'eau, l'énergie est très faible (domaines autour de 950 nm, 1130 nm et surtout 1400 nm et 1900 nm) (Fig. 1.11). A l'exception des zones de végétation, pour lesquelles on observe un pic dans l'infrarouge (à partir de 750 nm) également appelé *red edge*, la forme générale du profil spectral est assez semblable pour tous les pixels. L'information importante réside donc plutôt dans les détails.

L'anisotropie des images hyperspectrales a pour conséquence qu'il peut être préférable de privilégier l'une ou l'autre des dimensions. Une première manière de traiter ces données consiste à considérer le cube hyperspectral comme un empilement d'images pour les différentes longueurs d'onde. On y applique successivement des opérations issues du traitement d'image (com-

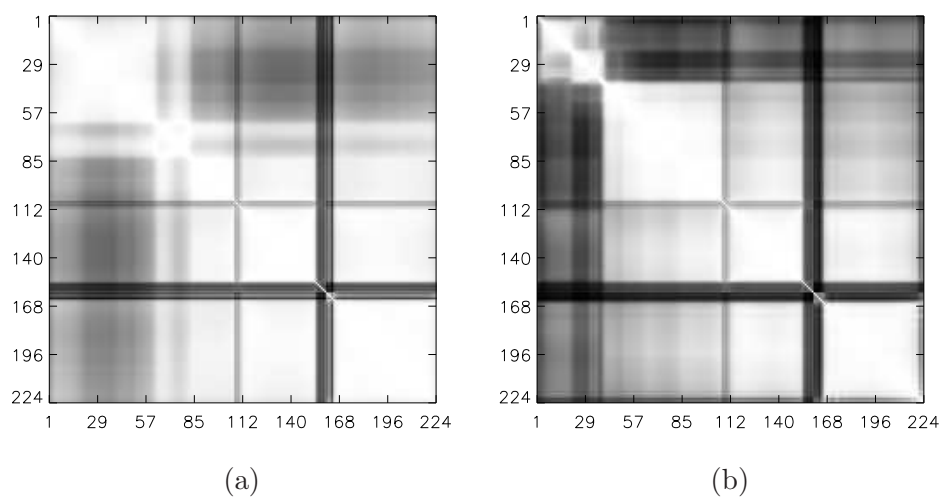


Fig. 1.10 Corrélation *spectrale* (interbandes), sur les images AVIRIS de *Railroad Valley* (a) et sur *Moffett* (b). Le pixel (i, j) de l'image de corrélation représente la valeur du coefficient de corrélation $r(i, j)$ entre les bandes i et j , blanc correspondant à la valeur 1 et noir à 0. Les bandes noires très marquées correspondent au domaine d'absorption de la vapeur d'eau où le signal est très faible et très bruité. Il apparaît que des corrélations existent entre des bandes très éloignées.

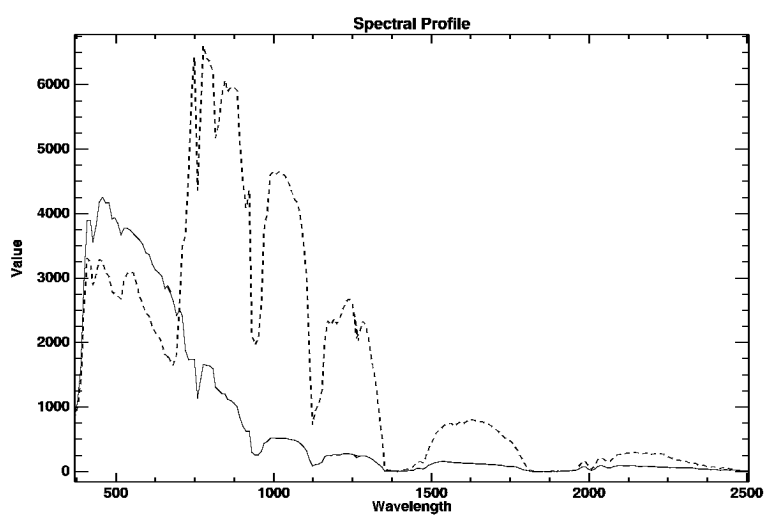


Fig. 1.11 Exemple de profil spectral pour une zone minérale (ligne continue) et une zone de végétation (ligne pointillée).

pression, estimation du bruit) pour toutes les images à toutes les longueurs d'onde et on combine ensuite ces résultats pour obtenir une opération globale. Cette approche est en continuité directe avec le traitement des images multispectrales.

Une deuxième manière d'aborder le problème consiste à traiter chaque pixel séparément. Les pixels sont alors représentés par une succession de valeurs représentant leur spectre. On peut alors traiter les pixels comme des vecteurs à n_λ dimensions, ou bien comme un signal à une dimension (n_λ étant le nombre de bandes spectrales de l'image). Des techniques de réduction de dimension peuvent être utilisées. Dans ce dernier cas, les traitements appliqués sont plutôt du domaine du traitement du signal et de la spectroscopie.

1.2.2 Problème de la normalisation

La question de la normalisation des données est aussi une question importante pour la généralisation des résultats. Classiquement, pour se ramener à une variable centrée réduite, on normalise la variable aléatoire x de la manière suivante :

$$x_n = \frac{x - \mathbb{E}[x]}{\sigma_x}, \quad (1.2)$$

$\mathbb{E}[x]$ étant l'espérance de x et σ_x^2 étant sa variance.

Le problème pour l'hyperspectral est de savoir de quelle manière on considère les données. Est-ce une image (i.e. bande spectrale) pour différentes longueurs d'onde, ou un spectre pour différents points de l'image ?

Pour certains, l'image hyperspectrale est vue comme une extension du multispectral. Dans ce cas, la normalisation est effectuée pour chaque image : on calcule alors la moyenne et la variance pour les valeurs d'une bande spectrale et on normalise grâce à ces facteurs (1.2). La normalisation est *dépendante* du contenu spatial de l'image : le même spectre sur deux images différentes donnera après normalisation deux spectres différents. On parlera de normalisation bande par bande.

Une autre possibilité est de faire la normalisation pour chaque pixel. On calcule alors la moyenne et la variance de chaque spectre, puis on normalise par rapport à ces valeurs. Ceci correspond à une normalisation par spectre. Le spectre normalisé obtenu est alors *indépendant* du contenu spatial de l'image : un même spectre donnera le même spectre normalisé quelque soit l'image. Par contre, le fait de considérer ou non les bandes bruitées (de valeur faible) changera les résultats.

Une autre manière de normaliser les données, de façon à accorder la même importance à toutes les bandes par la suite, est de diviser chaque bande par sa valeur moyenne (1.3). L'intérêt de cette opération est d'équilibrer un peu la situation entre les bandes qui reçoivent peu d'énergie et

celles qui en reçoivent plus selon l'éclairement solaire et les phénomènes atmosphériques.

$$x_n = \frac{x}{\mathbb{E}[x]} \quad (1.3)$$

1.2.3 Un espace presque vide

Pour les applications, une image hyperspectrale est souvent vue comme une représentation des différentes réalisations d'une variable aléatoire à n_λ dimensions, n_λ étant ici le nombre de bandes. Les différentes réalisations de cette variable aléatoire sont les différents pixels.

Cette vision des données est intéressante pour la classification ou la discrimination. Les pixels *proches* (notion de distance) dans cet espace vectoriel sont des pixels semblables, la classification regroupera donc ces pixels *proches* dans une même classe.

Les capteurs hyperspectraux acquérant généralement de l'ordre de 200 bandes, les données se retrouvent alors représentées dans un espace à 200 dimensions. Les espaces de grande dimensionnalité posent des problèmes particuliers qui ont été exposés par Hughes [Hug68] : de tels espaces sont presque vides. En effet, en considérant des données échantillonnées sur 12 bits et acquises pour 200 bandes, il y a alors $2^{12} = 4\,096$ possibilités pour chaque bande et un total de $4\,096^{200} \approx 10^{720}$ localisations possibles dans l'espace vectoriel de discrimination. Même avec des images de taille importante (de l'ordre de 10^6 pixels), la probabilité d'avoir deux pixels égaux ou voisins dans cet espace vectoriel est très faible.

D'autres problèmes apparaissent qui rendent difficile l'extension des idées intuitives en 3 dimensions. Landgrebe [Lan99] détaille une grande partie des effets provenant de ces espaces à grande dimension.

Ces considérations ont conduit à des méthodes pour réduire le nombre de dimensions. C'est le cas de l'Analyse en Composantes Principales (ACP, ou Principal Component Analysis, PCA) ou de l'Analyse en Composantes Indépendantes (ACI, ou Independent Component Analysis, ICA). Le nombre de dimensions est alors réduit à environ 10 ou 20. Ces décompositions projettent les données sur les vecteurs propres et sélectionnent les meilleurs vecteurs pour conserver l'énergie maximale. Il y a alors une perte d'information, mais ces pertes ne sont pas prohibitives pour certaines applications.

1.2.4 Une estimation de l'entropie

Dans un contexte de compression sans pertes, la notion d'entropie permet d'avoir une évaluation des performances que l'on peut espérer. Dans un message, on comprend intuitivement que tous les symboles n'apportent pas

la même information. Un symbole inattendu (faible probabilité) apportera plus d'information qu'un symbole attendu (probabilité forte).

Cette notion intuitive d'information a été formulée par Shannon en 1948 [Sha49]. L'information $\mathcal{I}(s_i)$ associée au symbole s_i est fonction de la probabilité d'apparition de s_i : $p(s_i)$. Cette fonction $F(p(s_i))$ est définie par trois conditions :

1. si la source ne délivre qu'un seul message (donc une probabilité égale à 1), l'information associée à ce message est nulle ;
2. en considérant s_i l'union de deux événements indépendants (s_j et s_k), $s_i = s_j \cup s_k$, avec $p(s_i) = p(s_j)p(s_k)$, l'information $\mathcal{I}(s_i)$ est égale à la somme des informations associées à s_j et s_k : $F(p(s_i)) = F(p(s_j)) + F(p(s_k))$;
3. F est continue, monotone et positive.

Une fonction F vérifiant ces trois conditions est $-\lambda \log(\cdot)$. L'unité binaire (binary unit ou bit) choisie par Shannon est définie comme la quantité d'information contenue dans une expérience binaire équiprobable : pile ou face avec une pièce équilibrée par exemple (Fig. 1.12). Ainsi, en choisissant d'utiliser les logarithmes en base 2, le facteur λ est égal à l'unité et la quantité d'information associée à un événement s_i s'écrit :

$$\mathcal{I}(s_i) = -\log_2(p(s_i)), \quad (1.4)$$

exprimée en bits.

L'entropie $H(S)$ d'une source sans mémoire est définie par :

$$H(S) = -\sum_{i=0}^{N-1} p(s_i) \log_2(p(s_i)), \quad (1.5)$$

où $\{s_0, \dots, s_{N-1}\}$ représente l'alphabet de la source.

L'entropie est une mesure de l'information moyenne de chaque symbole de la source. Elle est maximale si tous les symboles de la source sont équiprobables. De plus, l'entropie représente la longueur moyenne minimale d'un codage binaire sans pertes des données de la source. C'est pourquoi il est intéressant d'évaluer l'entropie des images hyperspectrales avant d'étudier des algorithmes de compression.

Dans un premier temps, on peut supposer avoir un modèle de source sans mémoire et estimer l'entropie des images en supposant que tous les pixels sont indépendants. Toutefois, il serait plus réaliste de prendre en compte une dépendance entre les pixels et d'introduire un modèle de Markov d'ordre 1, 2, ...

Cependant, on se heurte rapidement à un problème d'estimation de probabilités. Dans le cas d'un modèle d'ordre 0, chacun des symboles est une valeur codée sur 16 bits et peut donc prendre $2^{16} = 65536$ valeurs. En effet, après calibration, les données acquises sur 12 bits sont généralement codées

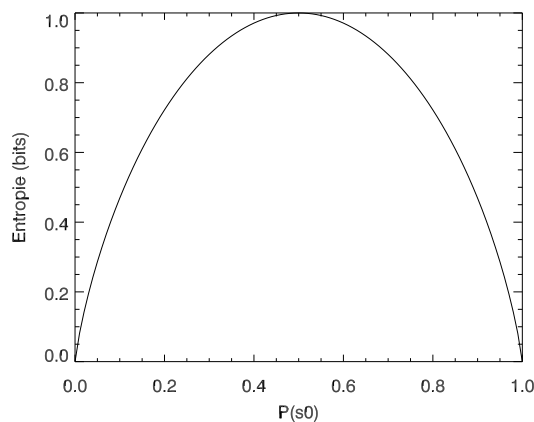


Fig. 1.12 Entropie en fonction de la probabilité $p(s_0)$ pour une source à deux états (s_0, s_1) d'où $p(s_1) = 1 - p(s_0)$. L'entropie est maximale pour $p(s_0) = p(s_1) = 0.5$.

sur 16. Le nombre de points dans une image hyperspectrale étant typiquement de l'ordre de $15 \cdot 10^6$ ($256 \times 256 \times 224$), on peut considérer que la loi de probabilité est suffisamment bien évaluée.

Dans le cas du modèle d'ordre 1, un symbole est alors un groupe de deux valeurs : on cherche la probabilité qu'un pixel prenne une valeur sachant la valeur du pixel voisin. Ce symbole prend une valeur parmi $(2^{16})^2 \sim 4.3 \cdot 10^9$. L'estimation de la loi de probabilité devient alors délicate. L'entropie a cependant été estimée jusqu'à l'ordre 3 pour donner une idée des ordres de grandeur (Tab. 1.3). Pour des ordres supérieurs, l'estimation de la densité de probabilité n'est plus possible de manière fiable.

Ordre	Entropie par pixel (en bits)
0	11.39
1	9.13
2	6.88
3	5.28

Tab. 1.3 Estimation de l'entropie à différents ordres d'une image hyperspectrale (*moffett3*). Cette estimation devient rapidement peu fiable à cause du manque de données pour évaluer la loi de probabilité.

Une autre manière de donner un ordre de grandeur de l'entropie (ou au moins une borne supérieure) est de tester les méthodes de codage sans pertes sans tenir compte de la nature de l'image. On utilise les utilitaires

- *gzip* : algorithme LZ77 [Ziv77] ;
- *bzip2* : algorithme de Burrows-Wheeler [Bur94], suivi de Huffman [Huf52] ;
- *ppmd* : basé sur un modèle de Markov d'ordre D [Cle84], dont l'implé-

mentation est décrite dans [Shk02].

Les performances dépendent de la façon dont sont arrangées les données (BIP, BSQ ou BIL, précisée dans l'annexe D.2) et sont présentées dans le tableau 1.4.

	original	gzip	bzip2	PPMD
BIP	16	11.87	8.19	8.43
BSQ	16	10.22	8.61	8.26
BIL	16	11.20	9.15	8.88

Tab. 1.4 Débit en bits par pixel par bande selon l'ordre des données.

Ceci nous permet seulement de comprendre que l'entropie des images hyperspectrales est inférieure à 8.19 bits (débit minimum en bits/pixel obtenu à l'aide d'un des algorithmes de compression sans perte standard). Comme l'entropie représente aussi la longueur minimale d'un code binaire sans perte, on peut ainsi montrer que les modèles d'ordre 0 et 1 envisagés dans l'estimation de l'entropie ne conviennent pas puisqu'ils conduisent à des valeurs d'entropies supérieures à 8.19 bits. Cela conforte l'idée que pour estimer l'entropie des images hyperspectrales, un modèle de source de Markov d'ordre supérieur à 1 doit être envisagé, ce qui pose les problèmes d'estimation des probabilités mis en évidence ci-dessus.

1.3 Compression

1.3.1 Notions de base

Deux grandes familles d'algorithmes de compression existent : ceux qui peuvent reconstituer l'information exacte (algorithmes sans pertes) et ceux qui tolèrent une perte d'information (algorithmes avec pertes). Les algorithmes sans pertes paraissent préférables à première vue. En général, les images satellitaires sont complexes. Les taux de compression obtenus avec des algorithmes sans pertes sur de telles images dépassent rarement 2 ou 3. Cette limitation est due en partie au bruit introduit par les capteurs haute résolution [Aia01]. Les algorithmes sans pertes présentent donc des limitations assez importantes en terme de compression. Tolérer quelques pertes permet de dépasser ces limites. Un autre avantage des algorithmes avec pertes est qu'ils permettent en général d'ajuster le débit. Si on tolère des distorsions assez importantes, on pourra compresser plus (débit plus faible) et inversement.

Un schéma typique de système de compression par transformée est présenté sur la figure 1.13. La quantification est une étape classique dans ce type de système, cependant, dans le cas de codage par plan de bits cette étape n'apparaît plus aussi clairement. Elle s'apparente plus à un tri des coefficients pour faciliter le travail du codeur entropique ou pour tirer parti des

dépendances entre les coefficients. Ces deux étapes sont souvent fortement imbriquées et figurent donc conjointement sur le schéma. Si la transformée est parfaitement réversible, on peut obtenir un système de compression sans pertes en supprimant l'étape de quantification ou en poussant le codage jusqu'au dernier plan de bits.

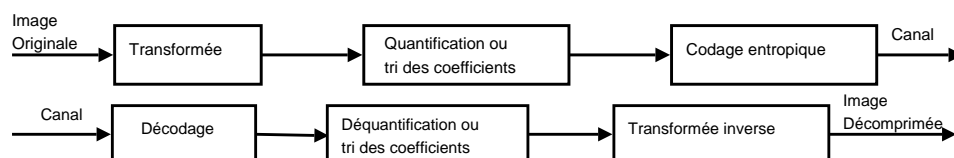


Fig. 1.13 Schéma de compression-décompression par transformée. Les pertes sont dues à l'étape de quantification.

1.3.1.1 Transformée

La transformée DCT (Discrete Cosine Transform) a montré de bonnes performances pour la compression des images. Elle est notamment utilisée dans le standard JPEG ainsi que dans l'algorithme de compression embarqué de SPOT 5. Les dernières avancées montrent que la transformée en ondelettes est plus intéressante car elle supprime notamment les effets de blocs. La transformée en ondelettes est utilisée dans le nouveau standard JPEG 2000 ainsi que par le satellite Pléiades-HR, le successeur de SPOT 5.

Il existe également des transformées adaptées aux données. Par exemple, la KLT (Karhunen-Loeve Transform) consiste à projeter l'image sur ses vecteurs propres. Cette transformation est souvent utilisée pour la réduction de données hyperspectrales sous le nom d'analyse en composantes principales. L'analyse en composantes indépendantes est également utilisée. Ces transformations sont efficaces mais dépendent de l'image. Le fait de devoir recalculer la transformée à chaque image augmente considérablement le coût de calcul. Dans un contexte de compression embarquée, il n'est pas envisageable d'effectuer une KLT sur 200 bandes spectrales à bord : cela impliquerait de diagonaliser une matrice de 200×200 coefficients. Ce calcul n'est d'ailleurs pas envisagé à bord dans le cas d'une image multispectrale de 4 bandes spectrales [Thi06].

1.3.1.2 Codage des coefficients

De nombreux travaux ont été développés sur le codage des coefficients issus de la transformée. On peut avoir une simple quantification, de la quantification vectorielle ou des systèmes par DPCM (Differential Pulse Code Modulation). Dans le cas de la DCT, il existe par exemple un algorithme d'allocation de bits pseudo optimal [Ger92].

Pour la transformée en ondelettes, des algorithmes de codage par plans de bits basés sur des structures d'arbres de zéros donnent de bons résultats. Il s'agit de EZW [Sha93] et SPIHT [Sai96] principalement. La quantité et l'ordre des données dépend des valeurs codées. Ces algorithmes seront détaillés plus précisément par la suite.

1.3.1.3 Codeur entropique

Le principe du codage entropique est de coder les symboles les plus fréquents avec un code plus court. L'exemple le plus simple d'un codeur entropique est le codage de Huffman [Huf52].

On considère par exemple le message constitué des symboles $(s_0, s_1, s_2, s_3, s_4, s_5, s_6, s_7)$ codés en binaire sur 3 bits (cf. Tab. 1.5).

Symbole	Code initial	Probabilité
s_0	000	0.25
s_1	001	0.21
s_2	010	0.15
s_3	011	0.14
s_4	100	0.0625
s_5	101	0.0625
s_6	110	0.0625
s_7	111	0.0625

Tab. 1.5 Probabilité des différents symboles et codes originaux : $l_{moyenne} = 3$

L'algorithme de Huffman fonctionne de la manière suivante :

1. Arranger les probabilités des symboles $P(s_i)$ par ordre décroissant et les considérer comme les feuilles d'un arbre.
2. Tant qu'il reste plus d'un nœud :
 - Regrouper les deux nœuds avec la plus faible probabilité pour former un nouveau nœud ayant pour probabilité la somme des deux nœuds regroupés.
 - Assigner arbitrairement 0 et 1 aux deux branches conduisant au nouveau nœud formé
3. Parcourir l'arbre depuis la racine pour trouver le code affecté à chaque symbole.

Avec l'exemple du tableau 1.5, on obtient l'arbre présenté sur la figure 1.14. On arrange les symboles par ordre décroissant. À la première étape, on regroupe les nœuds avec la plus faible probabilité, ce qui correspond aux valeurs s_6 et s_7 . On affecte à ce nœud la probabilité $p(s_6) + p(s_7) = 0.125$ et à chacune des branches la valeur 0 ou 1. On continue ensuite le processus, qui va regrouper s_4 et s_5 . Ensuite, les probabilités restantes sont 0.25, 0.21, 0.15, 0.14 et celles des deux nouveaux nœuds : 0.125 et 0.125.

Symbole	Probabilité	Code de Huffman
s_0	0.25	00
s_1	0.21	10
s_2	0.15	010
s_3	0.14	011
s_4	0.0625	1100
s_5	0.0625	1101
s_6	0.0625	1110
s_7	0.0625	1111

Tab. 1.6 Probabilité des différents symboles et code de Huffman : $l_{moyenne} = 2.79$

Ces deux nouveaux nœuds ont encore la probabilité la plus faible, on les regroupe donc pour former un nœud de probabilité 0.25. On continue ensuite l'algorithme jusqu'à l'obtention de l'arbre de la figure 1.14.

Pour savoir comment sera codé un symbole, on parcourt l'arbre en sens inverse (droite à gauche) pour arriver au symbole que l'on désire coder. On arrive au code final présenté dans le tableau 1.6.

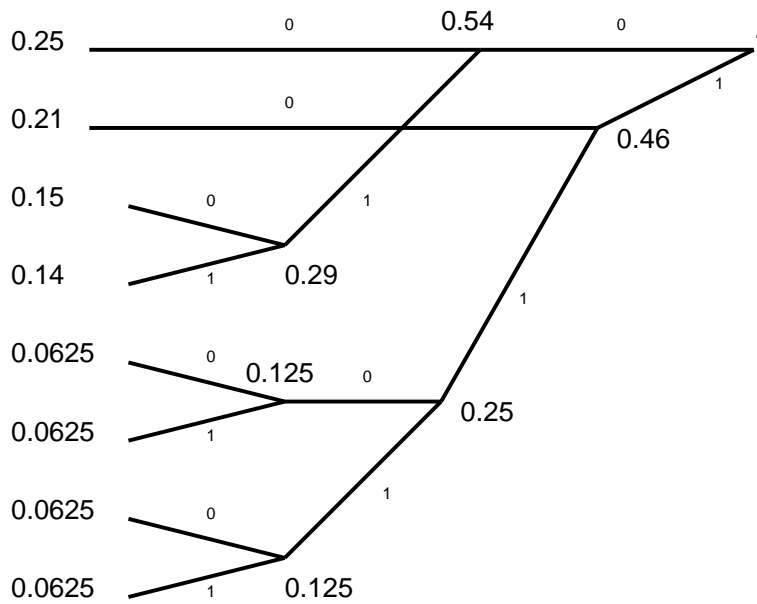


Fig. 1.14 Arbre obtenu avec les symboles et les probabilités présentées dans Tab. 1.5

L'entropie de ce message est égale à :

$$H(S) = - \sum_{i=0}^7 p(s_i) \log_2(p(s_i)) = 2.781 \text{bits.} \quad (1.6)$$

Au départ (Tab. 1.5), on avait une longueur moyenne du code qui était

égale à 3. Avec le code de Huffman la longueur moyenne est passée à 2.79. Le codage de Huffman garantit une longueur moyenne comprise entre $H(S)$ et $H(S) + 1$, c'est donc un code efficace. Si les probabilités sont connues, cet algorithme est optimal.

Le codage de Huffman a deux limitations principales :

- il n'est pas toujours facile d'évaluer les probabilités $p(s_i)$, notamment lorsque la taille des symboles augmente (pour des symboles codés sur 16 bits, il y a $2^{16} = 65536$ symboles possibles, il faut donc disposer d'une grande quantité de données pour évaluer les probabilités) ou lorsque la loi de probabilité change au cours de la transmission ;
- la construction du code de Huffman oblige à prendre un nombre entier de bits pour coder un symbole alors que la valeur optimale pourrait être fractionnaire. Pour atteindre une longueur fractionnaire, il faudrait utiliser le théorème de Shannon du codage sans bruit et coder les extensions d'ordre supérieur de la source (grouper les symboles initiaux par 2, 3, ...).

Le codage arithmétique [Ris79, Wit87, Mof98] remédie à cette dernière limitation du codage de Huffman. C'est actuellement une des méthodes de codage entropique les plus efficaces pour les données binaires. Cependant, l'implémentation matérielle du codeur arithmétique est délicate. Il n'est actuellement pas envisageable d'avoir un codeur arithmétique à bord d'un satellite. Cette situation va bien sûr probablement changer dans les prochaines années avec l'évolution des circuits électroniques.

1.3.2 État de l'art

Pour la compression des images hyperspectrales, deux grandes tendances existent : des systèmes de compression plutôt orientés vers les applications de type classification et des systèmes plus génériques [Mot06]. Les systèmes de compression orientés vers la classification sont basés sur la quantification vectorielle avec création d'un *codebook*. Les systèmes plus génériques sont basés sur un schéma classique de compression par transformée.

Par leur nature de données tridimensionnelles, les images hyperspectrales s'apparentent aux données vidéo et à certaines données médicales. La compression de ces données posent des problèmes semblables mais avec des contraintes parfois différentes. Par exemple, pour l'imagerie médicale, l'absence d'artefacts est une condition indispensable ; la compression des images ne doit pas pouvoir conduire à un mauvais diagnostic. En vidéo, pouvoir accéder de manière aléatoire à une partie de la vidéo sans avoir à décoder tout le flux depuis le début correspond à une demande forte. Des différences existent également concernant les propriétés à exploiter, dans le domaine vidéo la compensation de mouvements est très utilisée, ce qui n'existe pas dans le cas de l'hyperspectral.

1.3.2.1 Compression par quantification vectorielle

La compression par quantification vectorielle des images hyperspectrales est tentante. On peut voir une image hyperspectrale comme une collection de spectres. Si deux pixels de l'image correspondent au même objet au sol, les spectres obtenus pour ces deux pixels seront les mêmes (au bruit près). La notion de *codebook* apparaît naturellement : on identifie les éléments qui sont présents dans l'image, on leur affecte un code. Il suffit ensuite de garder en mémoire le *codebook* et la répartition de ces codes dans l'image.

La quantification vectorielle suit ce principe. Divers travaux sont en cours sur la quantification vectorielle pour la compression des images hyperspectrales au sein de la Canadian Space Agency [Qia00, Qia04], mais également dans d'autres équipes [Mot03, Rya97, Rya00]. Les taux de compression visés sont très importants, de l'ordre de 100, et introduisent une distorsion importante sur les images [Qia04]. La distorsion en terme d'influence sur une application de type classification est généralement faible.

Un système basé sur une classification à bord est utilisé pour le satellite Nemo sur Cois (Annexe E.2.8). Ce système, nommé Orasis (Optical Real-time Adaptive Signature Identification System), permet d'obtenir des taux de compression de l'ordre de 30 tout en préservant de bonnes performances pour une application de classification [Bow00].

Dans le cas de compression bord où l'application n'est pas forcément connue au moment de l'acquisition des données, il est délicat d'utiliser un système de compression introduisant de telles distorsions. Pour ces raisons, l'attention se portera plutôt sur des compresseurs à base d'ondelettes introduisant une distorsion faible des données.

1.3.2.2 Codage par transformée

Divers travaux ont été réalisés avec différentes transformées notamment au niveau spectral : utilisation d'une transformée de Karhunen-Loeve (aussi appelée Analyse en Composantes Principales), une DCT, une analyse en composantes indépendantes (ACI), ondelettes . . .

Les principaux développements ont lieu autour de la transformée en ondelettes. En effet, la transformée en ondelettes a montré de bonnes capacités de décorrélation dans un but de compression sur les images réelles. Des implémentations avec une complexité limitée existent et il est également possible de faire un traitement au fil de l'eau.

Il existe principalement deux tendances pour tirer partie de la décorrélation après transformation en ondelettes. La première méthode qui est appliqué dans le standard JPEG 2000 [Tau02] est d'utiliser un codeur arithmétique pour coder les coefficients d'ondelettes. La seconde méthode tire partie du lien existant entre les coefficients de la transformée situés dans différentes sous-bandes (même si la corrélation est presque nulle). Ces der-

nières méthodes sont appelées codage par arbres de zéros. Les deux méthodes de codage par arbres de zéros les plus populaires sont EZW [Sha93] et SPIHT [Sai96].

Le seul compresseur ondelettes embarqué pour les images hyperspectrales est basé sur une adaptation de SPIHT [Lan00]. Cette adaptation ne tire pas entièrement partie de la structure 3D des images hyperspectrales, mais c'est une implémentation qui a fonctionné pour plusieurs missions lointaines, notamment pour la mission ROSETTA à destination de la comète 67P Churyumov-Gerasimenko.

1.3.3 Applications

Les applications de Détection, Reconnaissance, Identification (DRI) sur les images hyperspectrales ne fonctionnent pas de la même manière que sur les images classiques. Sur ces dernières, les traitements évolués sont effectués sur les pixels en tenant compte des pixels voisins. Les opérations de détection de contours et les méthodes évoluées de segmentation fonctionnent toutes en considérant le pixel dans son environnement. Au contraire, pour les applications hyperspectrales, l'accent est mis sur l'identification du spectre. Les applications actuelles fonctionnent donc essentiellement à partir de l'information spectrale. À plus long terme, il serait néanmoins intéressant de considérer des applications qui tiennent compte du spectre du pixel, mais également des pixels avoisinants.

Les images classiques sont souvent analysées par des photo-interprètes d'où l'importance de critères de qualité image en adéquation avec les spécificités de l'œil humain. Les récents développements sur les critères de qualité pour le domaine des images classiques mettent surtout l'accent sur les aspects visuels des images. Les distorsions structurées (effets de blocs...) auxquelles l'œil est particulièrement sensible devront donc être bien détectées. Les applications de segmentation, fonctionnant souvent par détection des frontières, seront également très sensibles à ces effets.

À cause de leur complexité, les images hyperspectrales sont généralement analysées par des systèmes automatiques, il est donc important de prendre en compte les dégradations qui affecteront spécifiquement ce type de systèmes. Les critères de qualité évolués basés sur les critères visuels ne sont pas adaptés, des critères de qualité valides pour ces systèmes doivent donc être développés.

L'accent devra être mis sur la détection des distorsions du spectre qui pourraient produire une chute dans les performances des applications. Le prochain chapitre propose des critères qualité permettant de relier les dégradations (causées par exemple par les algorithmes de compression) aux performances des applications des images hyperspectrales.

Chapitre 2

Critères qualité pour les images hyperspectrales

2.1	Taxinomie des critères qualité	30
2.1.1	Définition	30
2.1.2	Intérêt pour la compression	31
2.1.3	Méthodes d'évaluation	31
2.1.4	Les méthodes bivariantes	32
2.1.5	Extension isotrope de critères de traitement du signal	33
2.1.6	Critères orientés hyperspectral.	35
2.1.7	Critères statistiques évolués.	39
2.1.8	Critères issus de la théorie de l'information	41
2.2	Démarche proposée pour l'évaluation des critères	43
2.2.1	Applications de référence.	44
2.2.2	Images hyperspectrales testées	47
2.2.3	Dégradations appliquées	48
2.2.4	Démarche	49
2.3	Résultats de l'étude	50
2.3.1	Présentation des résultats	50
2.3.2	Interprétation graphique et sensibilité des applications.	51
2.3.3	Interprétation des critères de qualité	54
2.3.4	Évaluation quantitative	55
2.3.5	Critères de qualité proposés.	58
2.4	Conclusion	59



A PRÈS avoir défini les particularités du domaine de l'hyperspectral et avant de commencer l'étude de la compression des images hyperspectrales, il est nécessaire de définir un ou plusieurs critères de qualité image. Ces critères permettront de comparer les différentes méthodes de compression et, en particulier, de mesurer les différents types de dégradations causées par les diverses méthodes de compression. Le terme de *critère qualité* est tout d'abord précisé, ainsi que son intérêt pour évaluer les performances de compression. Les critères qualité classiques sont ensuite énumérés et leur adaptation aux images hyperspectrales est faite (2.1). Une démarche est ensuite proposée pour évaluer l'adéquation de ces critères aux applications typiques des images hyperspectrales (2.2). Enfin, des critères sont sélectionnés pour leur bonne complémentarité (2.3). Cette partie de l'étude a fait l'objet de publications [Chr05a, Chr05b, Chr05c].

2.1 Taxinomie des critères qualité

2.1.1 Définition

Le terme de *qualité image* ou de *critère qualité* n'est pas un terme intuitif, il est donc nécessaire de préciser sa signification. La définition suivante est commune au CNES et à l'ONERA et est tiré d'un rapport commun [Kat99]. Cette définition couvre le domaine de l'acquisition d'image en général et n'est pas orienté spécialement pour l'imagerie hyperspectrale.

La notion de qualité image est indispensable pour caractériser le besoin selon les applications. Cette notion est donc relative et dépend du besoin en aval. Les critères de qualité image qualifient toute la chaîne image, depuis la scène observée jusqu'au produit final. Tous les aspects de la chaîne doivent donc être pris en compte, ce qui inclut la partie instrumentale (optique, détection des photons, chaîne électronique), les traitements bord (quantification, égalisation, compression), la plate-forme (contrôle d'attitude et orbite, architecture mécanique), mais également les traitements sol (débruitage, corrections radiométriques et géométriques) effectués avant la distribution de l'image.

Les critères adoptés sont fortement liés aux caractéristiques des instruments : bruit radiométrique, FTM (Fonction de Transfert de Modulation), localisation, altération des longueurs, . . . Certains critères sont plus difficiles à définir. C'est le cas par exemple d'un critère rendant compte des artefacts structurés créés par une compression de type JPEG.

Pour les images classiques, cette notion de qualité image peut être classée en trois domaines : la géométrie, la radiométrie et la FTM. La FTM, fortement liée à la radiométrie est aussi dépendante de la géométrie, c'est pour cela qu'elle est classée dans une catégorie particulière. Dans le cas des images hyperspectrales, la dimension spectrale devra faire l'objet d'une at-

tention particulière puisque c'est principalement selon cette dimension que fonctionnent les applications. Cette anisotropie au niveau des dimensions, tant au niveau des propriétés statistiques que des utilisations qui en sont faites rend délicate l'adaptation des critères qualité existants.

2.1.2 Intérêt pour la compression

Lorsque l'on utilise des techniques de compression avec pertes, l'utilisation de mesures de qualité est indispensable pour l'évaluation des performances. D'après Eskicioglu [Esk95], le problème majeur dans l'évaluation des techniques de compression avec pertes réside dans la difficulté à décrire la *nature* et l'*importance* des dégradations sur l'image reconstruite.

Cette notion de critère qualité, en adéquation avec les applications en aval, est donc particulièrement intéressante pour évaluer l'impact d'un procédé de compression sur l'exploitation de l'image finale.

2.1.3 Méthodes d'évaluation

Dans la suite, on appelle *bruit* toutes les modifications sur les données causées par le système. Ce *bruit* peut donc provenir des parties optique, électronique et numérique selon le système considéré. Le flou causé par la FTM, une déregression entre les bandes, aussi bien que les changements de valeurs causés par la quantification seront regroupés sous le terme de *bruit*.

Dans le cas des images classiques, plusieurs méthodes d'évaluation sont utilisées.

- Méthodes *subjectives* : basées sur l'évaluation de la qualité par des observateurs humains. Ces méthodes consistent à faire attribuer une note de qualité (Mean Opinion Score ou MOS) par un panel d'observateurs. Cette notation, lourde à mettre en œuvre, est adaptée lorsque les images sont exploitées par des observateurs humains (photo-interprètes).
- Méthodes *objectives* : basées sur des critères mathématiques pour évaluer la qualité des images. Les critères qualité utilisés pour mesurer les performances des instruments optiques sont, par exemple, le rapport signal/bruit (RSB ou SNR), l'erreur quadratique moyenne (EQM ou MSE). Ces méthodes sont surtout efficaces pour mesurer l'influence d'un bruit blanc, mais sont mal adaptées lorsque les dégradations sont structurées (blocs, rebonds aux abords des contours, ...). La Fonction de Transfert de Modulation (FTM ou MTF) permet de prendre en compte des dégradations spécifiques.

Dans le cas hyperspectral, les images sont surtout exploitées à l'aide de programmes informatiques, les critères d'évaluation utilisant des observateurs humains seront donc moins pertinents.

Les méthodes de calcul de ces critères mathématiques peuvent être de deux sortes, selon qu'on a accès ou non à l'image originale :

- Les méthodes *bivariantes* ou *full-reference* fonctionnent avec une comparaison par rapport à l'image originale. Ces méthodes sont généralement les plus efficaces, mais nécessitent d'avoir accès à l'image non dégradée.
- Les méthodes *univariantes* sont basées sur une estimation des critères de qualité lorsqu'on ne possède que l'image finale.
- Il existe également une méthode intermédiaire par références réduites à l'image originale. En plus de l'image codée, on transmet certains paramètres de l'image originale (moyenne, variance...). Une évaluation de la dégradation est possible à partir de ces données. C'est une technique qui est notamment utilisée dans les applications de transmission liées à la qualité de service (QoS) [Car03].

Bien sûr, les méthodes bivariantes sont plus robustes. Dans le cas de l'évaluation de la compression d'image, on dispose des images originales. Il est donc plus efficace d'utiliser des critères bivariants.

2.1.4 Les méthodes bivariantes

Les méthodes bivariantes sont basées sur des critères de distances entre l'image originale et l'image après passage dans la chaîne image. Dans notre cas, il s'agit de l'image originale, sans compression, notée I , et de l'image obtenue après compression-décompression dont on veut évaluer les distorsions, notée \tilde{I} . Les images hyperspectrales sont représentées sous la forme d'une matrice tridimensionnelle : $I(x, y, \lambda)$, x est la position du pixel dans la ligne, y est le numéro de la ligne et λ la bande spectrale considérée. n_x , n_y et n_λ sont respectivement le nombre de pixels par ligne, le nombre de lignes et le nombre de bandes spectrales. On notera également $\sum_{x=1}^{n_x} \sum_{y=1}^{n_y} \sum_{\lambda=1}^{n_\lambda} I(x, y, \lambda)$ par $\sum_{x,y,\lambda} I(x, y, \lambda)$. Ces notations seront conservées par la suite.

Les mesures bivariantes pour les images hyperspectrales peuvent être classées en 3 types :

- Une extension isotrope des critères classiques issus du traitement du signal. À l'origine, ces traitements sont adaptés à des données à une seule dimension. Ils ont été ensuite étendus avec succès aux images classiques à deux dimensions. Pour les images hyperspectrales, des extensions à une troisième dimension ont été réalisées. Dans ce premier groupe, on considère les critères traitant les 3 dimensions (2 spatiales et une spectrale) de la même manière. La spécificité des images hyperspectrales n'est donc pas prise en compte.
- Le second groupe contient les critères spécifiques aux images hyperspectrales. Ces critères fonctionnent de manière non symétrique

(moyenne d'un critère classique sur l'ensemble des pixels considérés comme des vecteurs par exemple). Ce type de critère permet d'insister sur certaines propriétés du spectral par rapport au spatial.

- Le troisième groupe contient des critères statistiques évolués adaptés de plusieurs manières aux images hyperspectrales.

2.1.5 Extension isotrope de critères de traitement du signal

Dans cette partie, les critères sont directement tirés des mesures statistiques classiques. Chaque valeur est considérée individuellement selon les dimensions spatiales et spectrale. Il n'est pas tenu compte de la différence de nature entre les deux types de dimension. L'aspect structurel des erreurs n'apparaît pas. Par la suite, d'autres mesures seront considérées qui prendront la dimension spectrale sous forme d'un vecteur par exemple.

Les mesures les plus utilisées sont issues des normes \mathcal{L}_p et basées sur les distances associées.

La norme \mathcal{L}_p est définie par :

$$\mathcal{L}_p(I) = \|I\|_p = \left(\sum_{x,y,\lambda} |I(x,y,\lambda)|^p \right)^{1/p}. \quad (2.1)$$

La distance correspondante, également appelée distance de Minkowski, est définie par

$$d_p(I, \tilde{I}) = \mathcal{L}_p(I - \tilde{I}) = \left(\sum_{x,y,\lambda} |I(x,y,\lambda) - \tilde{I}(x,y,\lambda)|^p \right)^{1/p}. \quad (2.2)$$

Ces mesures sont ensuite normalisées de différentes manières pour les rendre plus ou moins indépendantes de la taille et de la nature des données originales. Ceci nous permet de proposer un certain nombre de critères listés ci-après.

- Mean Absolute Error (MAE) ou moyenne de \mathcal{L}_1 . On moyenne ici par le nombre de pixels pour rendre la mesure indépendante de la taille de l'image.

$$\text{MAE}(I, \tilde{I}) = \frac{1}{n_x n_y n_\lambda} \sum_{x,y,\lambda} |I(x,y,\lambda) - \tilde{I}(x,y,\lambda)|. \quad (2.3)$$

- Erreur Quadratique Moyenne (EQM) ou Mean Square Error (MSE) ou moyenne de \mathcal{L}_2^2 . Ici, on moyenne également sur le nombre de pixels.

$$\text{EQM} = \text{MSE} = \frac{1}{n_x n_y n_\lambda} \sum_{x,y,\lambda} \left(I(x,y,\lambda) - \tilde{I}(x,y,\lambda) \right)^2. \quad (2.4)$$

Pour une meilleure interprétation, cette mesure peut être reliée à des notions de moyenne et variance. En notant μ_U la moyenne de l'ensemble U et σ_U^2 sa variance,

$$\text{MSE} = \sigma_{I-\tilde{I}}^2 + \mu_{I-\tilde{I}}^2 \quad (2.5)$$

- RMSE (Root Mean Square Error) à ne pas confondre avec \mathcal{L}_2 .

$$\text{RMSE} = \sqrt{\text{MSE}}. \quad (2.6)$$

$$\text{RMSE} = \sqrt{\frac{1}{n_x n_y n_\lambda} \sum_{x,y,\lambda} \left(I(x,y,\lambda) - \tilde{I}(x,y,\lambda) \right)^2}. \quad (2.7)$$

- Relative RMSE (RRMSE) : on normalise en plus ici avec le niveau du signal, ce qui sous entend qu'une erreur de même amplitude sera moins gênante sur un signal important que sur un signal faible.

$$\text{RRMSE} = \sqrt{\frac{1}{n_x n_y n_\lambda} \sum_{x,y,\lambda} \left(\frac{I(x,y,\lambda) - \tilde{I}(x,y,\lambda)}{I(x,y,\lambda)} \right)^2}. \quad (2.8)$$

- Les méthodes issues du traitement du signal fonctionnent surtout avec le Rapport Signal sur Bruit (RSB) ou Signal to Noise Ratio (SNR). On met ici en rapport la puissance de l'erreur avec la puissance du signal original (P_S).

$$\text{SNR}_{(dB)} = 10. \log_{10} \frac{P_S}{\text{MSE}}. \quad (2.9)$$

Lorsque le signal est de moyenne nulle, on a $P_S = \sigma_I^2$. En revanche, quand ce n'est pas le cas, une ambiguïté existe sur la définition de la puissance du signal, certains gardent la variance pour exprimer la puissance du signal (puissance du signal centré en fait) et d'autres choisissent la puissance réelle du signal.

On trouve également dans [Aia01] la formulation suivante :

$$\text{SNR}_{(dB)} = 10. \log_{10} \frac{\sigma_I^2}{\text{MSE} + \frac{1}{12}}. \quad (2.10)$$

Le facteur $\frac{1}{12}$ correspond à l'erreur de quantification. Introduire ce facteur permet de borner les valeurs du SNR lorsqu'il n'y a pas de pertes. La puissance de l'erreur de quantification, lorsque le nombre de bits est important, s'approche de $\frac{D^2}{12}$ avec D le pas de quantification. Dans le cas des images, le pas de quantification est de 1, le facteur devient donc $\frac{1}{12}$. Cette définition est notamment utilisée par Aiazzi [Aia01]. Elle n'est pas tout à

fait en accord avec la définition usuelle du SNR. Néanmoins, pour les valeurs usuelles de SNR ($\text{SNR} < 60 \text{ dB}$), la variation entre les deux définitions est au maximum de 0.4 dB . De plus, cette variation décroît logarithmiquement : à $\text{SNR} < 50 \text{ dB}$, la variation est inférieure à 0.04 dB .

- Le PSNR (Peak SNR) est également utilisé. Il tient compte de la dynamique du signal. Dans le cas des images numériques, on a généralement $\text{PeakSignal} = 2^q$, q étant le nombre de bits utilisés pour coder les valeurs, ce qui sous entend que la dynamique est bien dimensionnée.

$$\text{PSNR}_{(dB)} = 10. \log_{10} \frac{\text{PeakSignal}^2}{\text{MSE}}. \quad (2.11)$$

De manière équivalente au SNR, on trouve dans [Aia01] la définition suivante :

$$\text{PSNR}_{(dB)} = 10. \log_{10} \frac{\text{PeakSignal}^2}{\text{MSE} + \frac{1}{12}}. \quad (2.12)$$

- Maximum Absolute Distortion (MAD) ou \mathcal{L}_∞ ou Maximum Absolute Error (ambiguïté avec l'abréviation MAE, utilisée par Motta [Mot03])

$$\text{MAD} = \mathcal{L}_\infty(I - \tilde{I}) = \max_{(x,y,\lambda)} \left\{ \left| I(x, y, \lambda) - \tilde{I}(x, y, \lambda) \right| \right\}. \quad (2.13)$$

- Percentage MAD (PMAD) (également appelé Percentage Maximum Absolute Error, PMAE) correspond à la même notion que le Relative RMSE, on tient compte du niveau de l'erreur par rapport au niveau du signal initial.

C'est une mesure très utilisée, notamment par Pickering et Ryan [Bai01, Pic00, Rya98, Rya00]. La définition se trouve dans [Aia03a] :

$$\text{PMAD} = \max_{(x,y,\lambda)} \left\{ \frac{\left| I(x, y, \lambda) - \tilde{I}(x, y, \lambda) \right|}{I(x, y, \lambda)} \right\} \times 100. \quad (2.14)$$

Parmi ces critères, seuls \mathcal{L}_∞ (2.13) et le PMAD (2.14) bornent l'erreur sur l'ensemble de l'image. Les autres critères ne garantissent pas que des distorsions locales importantes ne seront pas présentes même si ces distorsions locales ont un impact très important sur les applications.

Une utilisation du MAD pourrait être faite, par exemple pour borner l'erreur de compression sous le bruit introduit par la quantification. Dans ce cas, la compression serait virtuellement sans pertes.

2.1.6 Critères orientés hyperspectral

Les critères présentés ici sont spécialement adaptés au domaine de l'hyperspectral. Certains, comme le SAM sont très largement utilisés. On définit par \mathbf{v} et $\tilde{\mathbf{v}}$ respectivement un vecteur de l'image originale et un vecteur de l'image reconstruite contenant les n_λ composantes spectrales du pixel (x, y) .

2.1.6.1 Critères basés sur les vecteurs spectraux

Un ensemble de critères classiques peut être proposé :

- Distance euclidienne minimale (Euclidean Minimum Distance (EMD) [Kes02]).

$$d_2(\mathbf{v}, \tilde{\mathbf{v}}) = \mathcal{L}_2(\mathbf{v} - \tilde{\mathbf{v}}) = \sqrt{\sum_{\lambda=1}^{n_\lambda} [\mathbf{v}(\lambda) - \tilde{\mathbf{v}}(\lambda)]^2} \quad (2.15)$$

$$= \sqrt{\sum_{\lambda=1}^{n_\lambda} [I(x, y, \lambda) - \tilde{I}(x, y, \lambda)]^2}. \quad (2.16)$$

$$EMD = \min_{\tilde{\mathbf{v}}} d_2(\mathbf{v}, \tilde{\mathbf{v}}) \quad (2.17)$$

- Average RMSE (A-RMSE) : Ce critère réalise une moyenne du RMSE calculé pour chaque vecteur [Aia03a, Aia03b].

$$ARMSE = \frac{1}{n_x n_y} \sum_{x,y} \sqrt{\frac{1}{n_\lambda} \sum_{\lambda} [I(x, y, \lambda) - \tilde{I}(x, y, \lambda)]^2} \quad (2.18)$$

$$= \frac{1}{n_x n_y \sqrt{n_\lambda}} \sum_{x,y} \mathcal{L}_2(\mathbf{v} - \tilde{\mathbf{v}}). \quad (2.19)$$

- Peak RMSE (P-RMSE) : Ce critère représente le RMSE maximum calculé selon les vecteurs.

$$PRMSE = \max_{x,y} \sqrt{\frac{1}{n_\lambda} \sum_{\lambda} [I(x, y, \lambda) - \tilde{I}(x, y, \lambda)]^2} \quad (2.20)$$

$$= \frac{1}{\sqrt{n_\lambda}} \max_{x,y} \{\mathcal{L}_2(\mathbf{v} - \tilde{\mathbf{v}})\}. \quad (2.21)$$

2.1.6.2 Maximum Spectral Angle (MSA)

Le Maximum Spectral Angle (MSA) est un critère utilisé dans de nombreux articles dont [Aia03a, Qia03, Kes02]. Le *spectral angle* est une mesure d'écart entre deux vecteurs. Pour chaque pixel, on mesure l'angle entre le vecteur original et le vecteur reconstruit. Il est plus rigoureux de parler de *Spectral Angle* pour définir la mesure entre deux vecteurs, le *Spectral Angle Mapper* désignant l'application de classification correspondant à cette mesure.

$$SA(\mathbf{v}, \tilde{\mathbf{v}}) = \cos^{-1} \left(\frac{\langle \mathbf{v}, \tilde{\mathbf{v}} \rangle}{\|\mathbf{v}\|_2 \cdot \|\tilde{\mathbf{v}}\|_2} \right), \quad (2.22)$$

ou encore directement en utilisant les valeurs du cube hyperspectral :

$$SA(x, y) = \cos^{-1} \left(\frac{\sum_{\lambda=1}^{n_\lambda} I(x, y, \lambda) \cdot \tilde{I}(x, y, \lambda)}{\sqrt{\sum_{\lambda=1}^{n_\lambda} I(x, y, \lambda)^2 \sum_{\lambda=1}^{n_\lambda} \tilde{I}(x, y, \lambda)^2}} \right). \quad (2.23)$$

Le SA est invariant par multiplication des vecteurs pouvant provenir d'une différence d'illumination ou d'orientation angulaire. Il est donc intéressant pour les images contenant des zones d'ombres et des zones plus fortement éclairées.

Ce critère peut être considéré en moyenne, ou bien en prenant le maximum de manière à borner l'erreur. Il faut toutefois noter que ce critère mesure les distorsions spectrales et que les distorsions radiométriques n'impliquent pas nécessairement de distorsions spectrales.

On prendra ici le maximum, avec

$$MSA = \max_{x,y} \{SA_{x,y}\} \quad (2.24)$$

2.1.6.3 Distance de Mahalanobis

Cette distance est une distance statistique entre deux vecteurs \mathbf{v} et $\tilde{\mathbf{v}}$

$$d_S(\mathbf{v}, \tilde{\mathbf{v}}) = \sqrt{(\mathbf{v} - \tilde{\mathbf{v}})^T S^{-1} (\mathbf{v} - \tilde{\mathbf{v}})}, \quad (2.25)$$

S étant la matrice de covariance. Cette distance est couramment utilisée en classification de manière générale. Toutefois, dans le domaine particulier de la classification en hyperspectral, le *Spectral Angle* (SA) est aussi beaucoup employé.

2.1.6.4 Similarité spectrale (SS)

Ce critère apparaît dans [Rup01, Swe01]. Il cherche à mesurer la ressemblance entre deux spectres vus comme des vecteurs à n_λ dimensions, \mathbf{v} et $\tilde{\mathbf{v}}$.

$$SS(\mathbf{v}, \tilde{\mathbf{v}}) = \sqrt{\text{RMSE}(\mathbf{v}, \tilde{\mathbf{v}})^2 + (1 - \text{corr}(\mathbf{v}, \tilde{\mathbf{v}}))^2} \quad (2.26)$$

avec

$$\text{RMSE}(\mathbf{v}, \tilde{\mathbf{v}}) = \sqrt{\frac{1}{n_\lambda} \sum_{\lambda=1}^{n_\lambda} (\mathbf{v}(\lambda) - \tilde{\mathbf{v}}(\lambda))^2} \quad (2.27)$$

et

$$\text{corr}(\mathbf{v}, \tilde{\mathbf{v}}) = \frac{\frac{1}{n_\lambda - 1} \sum_{\lambda=1}^{n_\lambda} (\mathbf{v}(\lambda) - \mu_{\mathbf{v}}) (\tilde{\mathbf{v}}(\lambda) - \mu_{\tilde{\mathbf{v}}})}{\sigma_{\mathbf{v}} \sigma_{\tilde{\mathbf{v}}}}. \quad (2.28)$$

Une valeur de SS faible indique une erreur faible.

2.1.6.5 RMSCCE (Root Mean Squared Correlation Coefficient Error)

Ce critère est utilisé par Briles dans [Bri96]. C'est une mesure de RMSE sur la matrice de corrélation spectrale (Fig. 1.9) entre la matrice avant compression et la matrice après compression.

Il n'est pas évident de voir le lien entre ce critère et la nature des distorsions détectées.

2.1.6.6 Valeurs propres

La plupart des applications de détection, reconnaissance, identification fonctionnant après une réduction de dimensions par analyse en composantes principales (PCA), il est logique de vérifier l'influence du bruit introduit par la compression sur ces valeurs propres. Dans [Hal99], il est montré que l'introduction d'un bruit de type bruit blanc a tendance à aplatir la courbe des valeurs propres, et donc augmente le nombre de valeurs propres significatives, diminuant le pouvoir de discrimination de la décomposition en composantes principales. Ce critère est à la limite entre application et critère, mais vu sa complexité (temps de calcul) et sa sensibilité à la nature de l'image, il sera plutôt considéré dans les applications.

2.1.6.7 Mesures de détectabilité

Cette mesure est mentionnée dans [Sub00], elle est définie selon l'équation

$$\text{scr} = \sqrt{\mathbf{v}_t^T S^{-1} \mathbf{v}_t}, \quad (2.29)$$

\mathbf{v}_t étant la cible recherchée dans l'image et \mathbf{v} le spectre d'un pixel de la scène, S est la matrice de covariance de l'image. Le critère de qualité qui est alors utilisé est le rapport entre la détectabilité dans l'image originale et la détectabilité dans l'image dégradée :

$$\rho = \frac{\tilde{\text{scr}}}{\text{scr}}. \quad (2.30)$$

Cette mesure est trop proche d'une application spécifique pour l'utilisation souhaitée. Elle ne sera donc pas utilisée dans la suite.

2.1.7 Critères statistiques évolués

Dans ce paragraphe, nous proposons l'extension de critères particuliers aux images hyperspectrales.

2.1.7.1 Critère Q

Un critère tel que le MSE, bien que largement utilisé ne fournit pas une mesure représentative de la réalité. Un critère évolué statistique est développé par Wang [Wan02a, Wan02b] pour le domaine de l'imagerie classique. Les résultats sont mieux corrélés à la perception visuelle que le MSE bien que ne faisant intervenir que des variables statistiques. Ce critère repose sur l'utilisation de la distorsion structurelle plutôt que sur la sensibilité à l'erreur. Une adaptation de ce critère au domaine vidéo a également été développée [Wan04b]. Une comparaison de cette métrique dans le cadre du VQEG (*Video Quality Expert Group*) semble donner de bons résultats [VQE03].

Soit U et V deux ensembles de valeurs (initialement les pixels de deux images) dont on veut évaluer la différence. On verra par la suite comment prendre U et V dans le cas d'une image hyperspectrale.

$$Q(U, V) = \frac{4 \sigma_{UV} \mu_U \mu_V}{(\sigma_U^2 + \sigma_V^2)(\mu_U^2 + \mu_V^2)}, \quad (2.31)$$

σ_{UV} notant l'intercorrélacion ($\mathbb{E}[(U - \mu_U)(V - \mu_V)]$), tandis que μ et σ^2 sont les notations de la moyenne et de la variance respectivement.

Le facteur Q est alors un nombre dans l'intervalle $[-1, 1]$.

On peut décomposer Q de la manière suivante :

$$Q(U, V) = \frac{\sigma_{UV}}{\sigma_U \sigma_V} \cdot \frac{2 \mu_U \mu_V}{\mu_U^2 + \mu_V^2} \cdot \frac{2 \sigma_U \sigma_V}{\sigma_U^2 + \sigma_V^2}. \quad (2.32)$$

Il est possible d'interpréter ces différents termes en utilisant des notions propres au traitement d'image : le premier terme est le coefficient de corrélation entre U et V , le second terme mesure l'écart de luminance moyenne et le troisième terme mesure la différence de contraste entre les deux images (en considérant que la variance σ^2 donne une estimation du contraste).

Il est possible d'appliquer ce critère à des sous-parties de l'image dans le cas de dégradations locales. Dans [Wan02a], Q est calculé pour des parties d'images par fenêtre glissante. Ensuite, le coefficient final est obtenu en moyennant les valeurs obtenues pour chaque fenêtre.

Les résultats ont été obtenus pour différents types de bruit : bruit gaussien additif, bruit impulsionnel, bruit multiplicatif (speckle), décalage de moyenne (mean shift), rehaussement de contraste, flou et compression JPEG (dégradations structurées). Ces résultats montrent dans tous les cas une meilleure correspondance avec les applications que le MSE.

Le critère Q peut être adapté de plusieurs manières aux images hyperspectrales. Nous proposons ici trois adaptations différentes.

- Application du critère à chaque pixel (spectre 1D) :

$$Q_\lambda = \min_{(x,y)} \left\{ Q \left(I(x, y, \cdot), \tilde{I}(x, y, \cdot) \right) \right\} \quad (2.33)$$

où $I(x, y, \cdot) = \{I(x, y, \lambda) \mid 0 \leq \lambda \leq n_\lambda - 1\}$.

- Application du critère à chaque bande (image 2D) :

$$Q_{(x,y)} = \min_\lambda \left\{ Q \left(I(\cdot, \cdot, \lambda), \tilde{I}(\cdot, \cdot, \lambda) \right) \right\} \quad (2.34)$$

où $I(\cdot, \cdot, \lambda) = \{I(x, y, \lambda) \mid 0 \leq x \leq n_x - 1, 0 \leq y \leq n_y - 1\}$.

- Une combinaison de ces deux critères peut être également envisagée avec :

$$Q_m = Q_\lambda \cdot Q_{(x,y)}. \quad (2.35)$$

Une adaptation du critère Q est réalisée par Alparone [Alp03] pour une image multispectrale de 4 bandes avec utilisation des quaternions. Néanmoins, cette adaptation n'est valable que pour les images de 4 bandes, les quaternions étant à 4 composantes.

2.1.7.2 Fidélité (F)

Ce critère a été développé par Eskicioglu [Esk93] et utilisé dans [Lam95, Oli98]. On définit la fidélité par :

$$F(U, V) = 1 - \frac{\mathcal{L}_2^2(U - V)}{\sigma_U^2 + \mu_U^2}, \quad (2.36)$$

soit pour les images hyperspectrales, une fidélité spatiale définie par :

$$F(I, \tilde{I}) = 1 - \frac{\sum_{x,y,\lambda} [I(x, y, \lambda) - \tilde{I}(x, y, \lambda)]^2}{\sum_{x,y,\lambda} [I(x, y, \lambda)]^2}. \quad (2.37)$$

Le terme *spatial* est ici à comprendre par opposition au domaine de Fourier, le critère F mesure une fidélité de l'image dans son domaine d'origine.

L'adaptation aux images hyperspectrales est réalisée de trois manières différentes. En plus de la formulation ci-dessus considérant les trois dimensions de manière isotrope, on propose les adaptations F_λ et $F_{(x,y)}$ privilégiant respectivement les directions spectrale et spatiales,

$$F_\lambda = \min_{(x,y)} \left\{ F \left(I(x, y, \cdot), \tilde{I}(x, y, \cdot) \right) \right\} \quad (2.38)$$

et

$$F_{(x,y)} = \min_\lambda \left\{ F \left(I(\cdot, \cdot, \lambda), \tilde{I}(\cdot, \cdot, \lambda) \right) \right\}. \quad (2.39)$$

2.1.7.3 Structural Content (SC)

Le contenu structurel ou Structural Content (SC) est mentionné dans les mêmes articles que ceux évoquant la fidélité et est défini comme

$$\text{SC} = \frac{\sigma_I^2 + \mu_I^2}{\sigma_I^2 + \mu_I^2} \quad (2.40)$$

soit

$$\text{SC} = \frac{\sum_{x,y,\lambda} [\tilde{I}(x,y,\lambda)]^2}{\sum_{x,y,\lambda} [I(x,y,\lambda)]^2}. \quad (2.41)$$

2.1.7.4 Normalized Cross-Correlation (NCC)

L'intercorrélacion normalisée ou Normalized Cross-Correlation (NCC) est mentionnée dans les mêmes articles que ceux proposant d'utiliser la fidélité

$$\text{NCC} = \frac{\sum_{x,y,\lambda} I(x,y,\lambda)\tilde{I}(x,y,\lambda)}{\sum_{x,y,\lambda} [I(x,y,\lambda)]^2}. \quad (2.42)$$

L'interprétation physique de ces trois derniers critères (fidélité, SC, NCC) reste plus difficile. Dans [Esk93], Eskicioglu mesure les performances de ces trois critères. L'intercorrélacion normalisée, ainsi que le contenu structurel ne donnent pas de très bons résultats. La fidélité semble plus performante.

2.1.8 Critères issus de la théorie de l'information

La théorie de l'information permet de mesurer l'information contenue dans des données. Des critères basés sur cette théorie pourraient être intéressants pour mesurer la perte d'information entre une image originale et une image dégradée.

2.1.8.1 Spectral Information Divergence (SID)

Ce critère est dérivé de la distance de Kullback-Leibler [Aia03a]. En notant $\mathbf{v} = (v_0, v_1, \dots, v_i, \dots)$ on a

$$\text{SID}(\mathbf{v}, \tilde{\mathbf{v}}) = D(\mathbf{v} \parallel \tilde{\mathbf{v}}) + D(\tilde{\mathbf{v}} \parallel \mathbf{v}) \quad (2.43)$$

avec $D(\mathbf{v} \parallel \tilde{\mathbf{v}})$ la distance de Kullback-Leibler, ou divergence entropique, ou discrimination définie entre deux vecteurs par

$$D(\mathbf{v} \parallel \tilde{\mathbf{v}}) = \sum_i p_i \log \left(\frac{p_i}{\tilde{p}_i} \right) \quad (2.44)$$

où $p_i = \frac{v_i}{\|\mathbf{v}\|_1}$ et $\tilde{p}_i = \frac{\tilde{v}_i}{\|\tilde{\mathbf{v}}\|_1}$,
d'où

$$\text{SID}(\mathbf{v}, \tilde{\mathbf{v}}) = \sum_i (p_i - \tilde{p}_i) \log \left(\frac{p_i}{\tilde{p}_i} \right). \quad (2.45)$$

Les mêmes remarques peuvent s'appliquer pour le SID et pour le SAM : ce critère peut être considéré en moyenne, ou bien en prenant le maximum de manière à borner l'erreur. De même, ce critère mesure les distorsions spectrales tandis que les distorsions radiométriques n'impliquent pas nécessairement de distorsions spectrales.

On prendra ici le maximum, avec

$$\text{MSID} = \max_{x,y} \{\text{SID}_{x,y}\}. \quad (2.46)$$

2.1.8.2 Information mutuelle

L'entropie d'un ensemble de valeurs discrètes X à valeurs dans $[1, 2^q]$ est définie par Shannon [Sha49] comme

$$H(X) = - \sum_{i=1}^{2^q} P(i) \ln P(i), \quad (2.47)$$

où $P(i)$ désigne la proportion de X prenant la valeur i .

L'entropie conjointe entre X et Y , qui représente l'*information* apportée par X et par Y , est alors définie par :

$$H(X, Y) = - \sum_{i=1}^{2^q} \sum_{j=1}^{2^q} P(i, j) \ln P(i, j). \quad (2.48)$$

L'information mutuelle entre X et Y correspond à la *quantité d'information* commune à X et à Y

$$I(X, Y) = H(X) + H(Y) - H(X, Y), \quad (2.49)$$

et l'entropie conditionnelle s'écrit :

$$H(X/Y) = H(X) - I(X, Y). \quad (2.50)$$

Le critère considéré est alors le rapport

$$\frac{I(X, Y)}{H(X/Y)}. \quad (2.51)$$

Ce critère n'est pas largement employé et n'est utilisé que dans peu d'articles [Ma01]. Une étude plus approfondie serait nécessaire pour mesurer son apport sur les images hyperspectrales.

Ces critères s'appliquent bien aux images classiques, car le nombre de pixels est généralement grand devant les valeurs possibles de ces pixels. Par exemple, les images SPOT5 (pour le mode HM) font 12000×12000 pixels sur 8 bits, donc environ $144 \cdot 10^6$ pixels pour 256 valeurs possibles. Vu le large nombre d'échantillons devant le nombre de valeurs prises, il est possible d'effectuer des statistiques. En revanche, comme expliqué dans le chapitre précédent (paragraphe 1.2.4), une image hyperspectrale sur 210 bandes échantillonnées à 12 bits a $4096^{210} \approx 10^{720}$ valeurs de vecteurs possibles. Les méthodes statistiques ne sont donc plus fiables en considérant les pixels comme vecteurs. Si on veut utiliser ces méthodes, il est indispensable de travailler sur des sous-ensembles de bandes spectrales. Comme la corrélation spectrale est importante également pour des bandes éloignées, il peut être délicat de déterminer ces sous-ensembles.

2.2 Démarche proposée pour l'évaluation des critères

L'objectif consiste à valider les critères qualité retenus par rapport à un panel d'applications courantes en hyperspectral. On peut comparer ce problème à la validation de critères qualité pour les images classiques ou pour les séquences vidéo, l'objectif étant dans ce cas d'avoir des résultats significatifs par rapport à l'œil humain. Ce problème n'est pas simple et a fait l'objet de nombreuses publications au cours des 20 dernières années. La démarche utilisée ici est principalement adaptée des travaux du *Video Quality Expert Group* [VQE03] dont le but est de déterminer un critère objectif de mesure de la qualité des séquences vidéo.

La démarche du VQEG va être précisée pour mieux comprendre l'adaptation qui en est faite. Un panel de critères qualité mathématiques est d'abord sélectionné, ces critères servent à évaluer diverses dégradations produites sur les séquences vidéo. Tous ces critères sont de type *bivariant (full-reference)*, c'est-à-dire que l'on dispose de la séquence originale pour effectuer le calcul des critères. D'un autre côté, un panel d'observateurs évalue les séquences dégradées en leur donnant une note (*Mean Opinion Score* ou MOS). Cette évaluation se déroule également par rapport à la séquence vidéo de référence visualisée avant ou après la séquence dégradée. Le but est alors de trouver le critère de qualité qui se rapproche le plus possible de la perception humaine, l'important étant surtout d'avoir une relation stable entre le critère et la notation humaine.

Dans le cas des images hyperspectrales, ce sont rarement des observa-

teurs humains qui vont directement utiliser les images. La quantité d'information est trop importante pour être directement exploitée, on utilise donc des traitements automatisés. **Ce sont les traitements automatisés qui vont jouer le rôle des observateurs humains dans l'évaluation des critères.**

Des applications de référence sont choisies pour comparer les résultats obtenus par les critères qualité. Ces applications sont détaillées dans la partie 2.2.1. Les tests seront effectués sur plusieurs images dont les particularités sont précisées dans 2.2.2. Dans 2.2.3, les types de dégradations appliquées ainsi que les phénomènes qu'elles modélisent sont expliqués. Le panel d'images de départ présentant des situations variées rencontrées en hyperspectral sera dégradé de différentes manières et à différents niveaux. La qualité de ces images dégradées sera ensuite évaluée par les différents critères qualité et comparée aux résultats obtenus par les applications de référence dans 2.2.4.

Certains articles utilisent directement une application pour qualifier les dégradations des images hyperspectrales : nombre de pixels mal classifiés après application d'un classificateur bayésien K-moyennes [Lur94], détection d'anomalies et identification [She98] mais ceci induit un coût calculatoire important, une difficulté de mise en place et un manque de généralité des résultats. Si on peut proposer un ou plusieurs critères mathématiques de coût calculatoire plus faible mais qui nous permettent d'avoir une idée de l'impact des dégradations sur les applications, cela serait plus intéressant.

2.2.1 Applications de référence

Les applications pour le domaine de l'hyperspectral sont encore au stade du développement. La plupart sont du type DRI (cf. 1.3.3) et procèdent par identification entre les spectres des pixels.

2.2.1.1 Détection d'anomalies

Cette application détecte les anomalies présentes dans une image. Elle fonctionne par poursuite de projection. Une analyse en composantes principales permet de ne conserver que les 12 meilleures composantes. La composante qui maximise le kurtosis ¹ des valeurs de l'image est sélectionnée. Les points en queue de distribution sont alors considérés comme étant les anomalies. Un programme a été développé à l'ONERA à partir de l'article de Chiang [Chi01].

Pour vérifier la bonne détection ou non des cibles recherchées, on introduit artificiellement des cibles subpixels en différentes proportions. On réalise

¹moment d'ordre 4, qui correspond au coefficient d'aplatissement relativement à une distribution Gaussienne

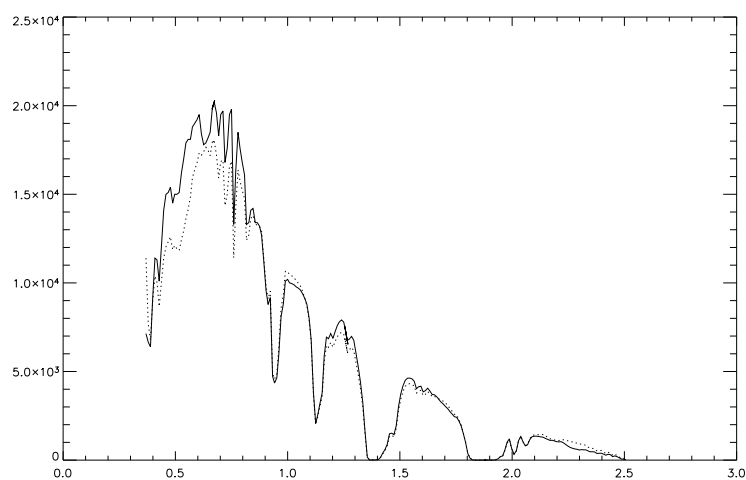


Fig. 2.1 Spectre du sable (ligne pointillée) et spectre de la peinture couleur sable (ligne continue).

une combinaison entre le spectre présent sur l'image et un spectre d'anomalie. Ce spectre d'anomalie est généré à partir d'un spectre présent dans une librairie spectrale. Les effets atmosphériques sont simulés sur ce spectre pour calculer la luminance qui serait vue à l'altitude du capteur. L'image testée pour ce cas étant principalement constituée de zones sableuses et rocheuses (site Railroad Valley, voir D), le spectre d'une peinture couleur sable a été utilisé pour réaliser les mélanges. La figure 2.1 illustre la similitude entre les deux spectres. 100 cibles sont insérées dans des proportions variant entre 10% et 90% de la surface du pixel. Les bandes de valeurs très faibles (absorption par la vapeur d'eau) sont retirées pour ne pas perturber la détection.

Deux critères classiques pour évaluer les performances de la détection sont la proportion de cibles correctement détectées ainsi que le nombre de fausses alarmes. Les fausses alarmes sont les pixels signalés comme étant des cibles alors qu'ils n'en sont pas. Tout système de détection est réglé en fonction d'un compromis entre la quantité de fausses alarmes par rapport au nombre de pixels correctement détectés. Ce compromis donne lieu au tracé d'une courbe appelée COR (Caractéristiques Opérationnelles du Récepteur ou *ROC curve*) servant à évaluer les performances du système de détection en traçant la probabilité de détection en fonction de la probabilité de fausse alarme.

La proportion de cibles insérées est faible par rapport au nombre de pixels dans l'image (100 cibles pour 65 536 pixels) puisqu'il s'agit d'une détection d'anomalies. De plus, l'algorithme de détection est très sensible aux réglages des paramètres (nombre de boucles, seuil . . .) et aux données de l'image. Des perturbations de même nature, mais dont la réalisation est aléatoire (bruit blanc ou déregISTRATION par exemple) donneront des résultats complètement

différents pour deux réalisations successives d'amplitude égale. La dispersion des résultats est donc assez importante et l'extrapolation des conclusions est délicate.

2.2.1.2 Classification

Les simulations pour la classification sont réalisées sur l'image Moffett du capteur AVIRIS (voir annexe D) qui présente des zones homogènes contrastant fortement entre elles (marais salants) dans une partie de l'image et une zone plutôt urbaine dans l'autre partie.

La classification réalisée est une classification supervisée. Des régions d'intérêt sont définies pour servir de base à la classification. Six régions sont finalement gardées. Les pixels sont classifiés par rapport à leur distance au sens du *Spectral Angle* (Eq. 2.22) ou de *Mahalanobis* (Eq. 2.25) aux pixels moyens des régions d'intérêt, une classification par *Maximum de Vraisemblance* est également effectuée. Pour la classification par *Spectral Angle* un seuil est fixé à un SAM de $0.15rad$. Certains pixels se trouveront à des distances supérieures, et ne seront donc pas classifiés. L'importance de cette classe de rejet sera précisée par la suite.

En l'absence de réalité terrain pour qualifier les résultats de la classification en absolu, on compare les *variations* de classification causées par les dégradations par rapport à l'image non dégradée. Pour chaque dégradation appliquée, deux valeurs sont utilisées : le nombre de pixels qui ne sont pas classifiés (dans le cas où on a une classe de rejet) et le nombre de pixels qui ont changé de classe.

La quantité de données étant assez importante (315 000 pixels sur 224 bandes) par rapport au nombre de classes, les résultats sont peu sensibles aux différentes réalisations des dégradations aléatoires dans le cas du bruit blanc.

2.2.1.3 Valeurs propres

Pour beaucoup d'applications, un prétraitement est d'abord effectué pour réduire le nombre de dimensions. Dans la plupart des cas, une projection sur les composantes principales (ACP ou PCA) est effectuée. Seules les composantes dont la valeur propre est supérieure à un seuil arbitraire sont conservées.

Hallidy [Hal99] a montré que la présence de bruit dans l'image provoque un aplatissement de la répartition des valeurs propres : la décroissance est moins rapide, il faudra donc garder plus de bandes.

On va donc également considérer ces valeurs propres comme une application sur laquelle les effets des distorsions devront être mesurés. Le critère finalement retenu est le nombre de valeurs propres que l'on doit conserver



Fig. 2.2 Moffett4 avec des zones uniformes : composition colorée à partir des bandes 90 (1211.33 nm), 40 (731.21 nm) et 20 (557.07 nm). Cette zone contient des marais salants et de la mer.



Fig. 2.3 Moffett3 plus contrastée : composition colorée à partir des bandes 90 (1211.33 nm), 40 (731.21 nm) et 20 (557.07 nm). Cette zone contient des bâtiments, des routes.

pour garder 99.9% de la somme des valeurs propres, ce qui correspond à 99.9% de l'énergie de l'image.

Cette application est assez stable, mais il est difficile d'évaluer la qualité de l'image directement à partir de ce critère. Selon les applications qui se trouvent après, une diminution du nombre des valeurs peut être considérée comme une variation positive ou négative.

2.2.2 Images hyperspectrales testées

Toutes les images testées dans cette partie sont des images AVIRIS. Les caractéristiques de ce capteur sont rappelées dans l'annexe E.2.1. Le choix s'est porté sur ce capteur pour son rapport signal sur bruit important sur les images fournies. En effet, un faible bruit sur les images originales est indispensable pour mesurer l'influence du bruit ajouté lors des simulations. Plusieurs images hyperspectrales sont sélectionnées, ces images représentent des zones variées, qui peuvent être uniformes ou hétérogènes (désert, ville, plans d'eau ...). Les images utilisées sont présentées dans l'annexe D.

Dans le cas de la détection, les simulations s'effectuent sur des échantillons de 256×256 pixels du site de Railroad Valley. Pour la classification, les simulations se font directement sur des vignettes d'images AVIRIS 256×256 (site Moffett) en conservant toutes les bandes. Ces simulations ont lieu sur des parties homogènes de l'image (zones de marais salant, uniformes) et sur des zones plus hétérogènes (zones de ville). Les résultats de référence pour les classifications par SAM sont présentés sur les figures 2.2 à 2.5.

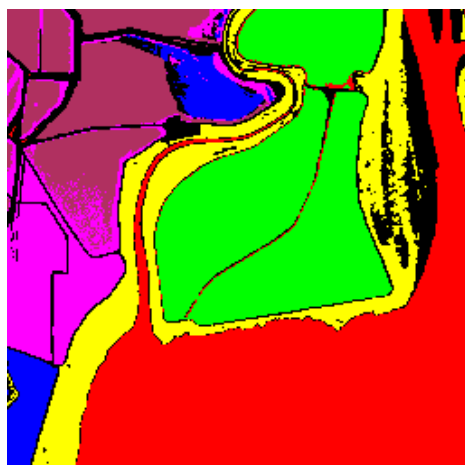


Fig. 2.4 Classification SAM de référence pour Moffett4, les régions en noir correspondent aux pixels non classifiés.

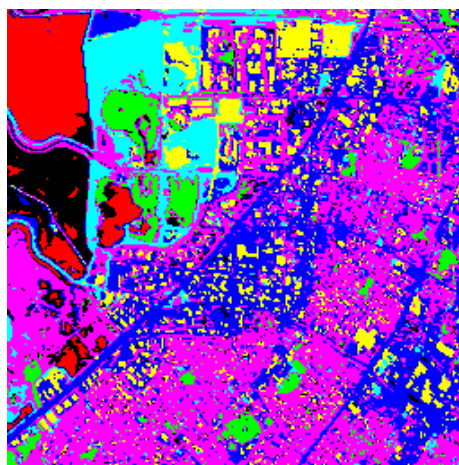


Fig. 2.5 Classification SAM de référence pour Moffett3, les régions en noir correspondent aux pixels non classifiés.

2.2.3 Dégradations appliquées

Différentes dégradations sont appliquées aux images afin de simuler au mieux les pertes dues à la compression ou à des défauts instrumentaux.

2.2.3.1 Bruit blanc gaussien additif

Le premier type de dégradation appliqué est un bruit blanc gaussien additif dont on modifie la variance. Ce bruit modélise les défauts d'acquisition de l'appareil au niveau électronique, ainsi que le bruit de quantification. On ajoute à l'image un tirage aléatoire de ce bruit blanc. Certains critères ou certaines applications posséderont une sensibilité différente à la réalisation du bruit.

2.2.3.2 Flou

La deuxième modification est un flou appliqué à l'image, selon la dimension spectrale, les dimensions spatiales ou selon les trois dimensions. Ce flou représente les défauts provoqués sur l'image par un défaut de FTM (taille de la tache image), par une compression par ondelettes (qui a tendance à rendre l'image floue) ou par une réduction du pas d'échantillonnage. Le flou est appliqué par un filtre moyenneur sur une fenêtre de taille variable (1 à 7 pixels de large selon les dimensions spatiales ou spectrale), la contribution des pixels voisins peut également être réduite (présence d'un coefficient d'atténuation sur le filtre).

2.2.3.3 Phénomène de Gibbs

Le troisième type de bruit ajouté à l'image est un phénomène de Gibbs qui provoque un rebond après les fortes transitions. Cet effet apparaît lorsqu'on applique un filtre passe-bas à un signal [Mal97]. Il est modélisé en appliquant un filtre de Wiener modifié [Oli98]. Le filtrage n'est réalisé que pour les images spatiales (longueur d'onde par longueur d'onde).

2.2.3.4 Compression

La quatrième dégradation utilisée est la compression par JPEG 2000. La norme est utilisée avec transformée en ondelettes pour décorrélérer les bandes spectrales entre elles. Différents débits sont utilisés pour produire des dégradations de forte ou faible amplitude.

2.2.4 Démarche

Un grand nombre de critères qualité ont été programmés. Les différentes images pour les différentes dégradations sont successivement évaluées par rapport aux images originales. Une évaluation est également réalisée par rapport aux applications de référence.

La comparaison entre les résultats de l'évaluation par les applications ou par les critères de qualité est souvent mise en relation par un facteur de corrélation. Des résultats fortement corrélés indiquent que le critère de qualité identifie l'impact des dégradations sur l'application. C'est la démarche la plus simple et la plus rapide à effectuer.

Dans [Bri04], Brill présente une méthode permettant de comparer des critères de qualité objectifs pour les séquences vidéos par rapport aux évaluations obtenues par un panel d'observateurs. Cette méthode est plus complète qu'une simple mesure de corrélation. Il est possible d'effectuer un parallèle entre les mesures subjectives des observateurs dans le cas de séquences vidéo et les résultats objectifs des applications dans le cas de l'hyperspectral.

Néanmoins, la méthode n'est pas directement utilisable, principalement parce que dans le cas des séquences vidéo, il y a une seule *application*, l'observateur humain, qui possède une variabilité statistique. Dans le cas des images hyperspectrales, il y a diverses applications qui requièrent des propriétés différentes. Il faut donc pouvoir qualifier plus précisément la nature des dégradations. Les nuages de points obtenus par la méthode complète de Brill (Fig. 2.6) sont assez délicats à interpréter et ne permettent pas de séparer les différents cas pour lesquels certains critères sont plus efficaces.

La méthode est donc modifiée pour donner des résultats interprétables dans le cas des images hyperspectrales. On appelle situation, une certaine dégradation avec un certain niveau sur une image donnée. Pour chaque situation i , un *score* est calculé pour un critère qualité (noté O_i , score objectif)

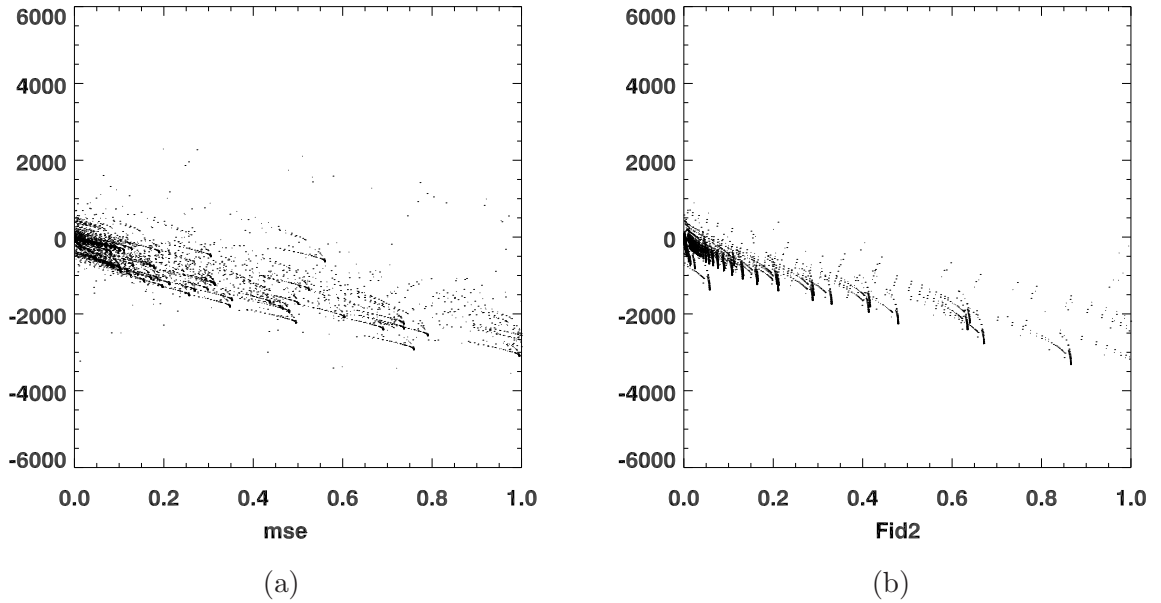


Fig. 2.6 Résultats obtenus par la méthode de Brill : (a) pour le MSE et (b) pour le F_λ . Les nuages obtenus doivent être le moins dispersés possible. Cette méthode ne fonctionne pas si les réponses ne sont pas linéaires.

et pour une application (noté S_i score subjectif). Environ 10 000 situations ont été testées et pour chaque cas, le score a été calculé pour diverses applications (temps de calcul de l'ordre de la minute ou de l'heure) et pour les 15 critères testés (temps de calcul de l'ordre de la seconde). Les courbes représentant directement les performances de l'application par rapport au critère qualité observé (points (O_i, S_i)) sont plus explicites et permettent de repérer facilement les dégradations faisant plus réagir certains critères (figures 2.7 à 2.14).

2.3 Résultats de l'étude

Une sélection des courbes obtenues pour tous les critères de qualité testés est présentée ci-dessous. Dans ce qui suit, sont expliqués la manière de lire ces courbes, ainsi que le type d'interprétation que l'on peut en tirer. La fin de ce chapitre détaillera les conclusions obtenues à partir des simulations effectuées.

2.3.1 Présentation des résultats

Les courbes sont tracées pour 15 critères de qualité, pour différents niveaux et types de dégradation. Ces résultats sont comparés aux performances

des applications sélectionnées.

Chaque courbe est tracée en représentant la valeur du critère qualité en abscisse et la performance de l'application en ordonnée. Chaque point correspond à une situation donnée, i.e. un type et un niveau de dégradation appliqué à une image. Pour faciliter la lecture, la valeur du critère qualité représentant une qualité maximum est représentée à gauche. L'image d'origine se trouve donc ainsi positionnée dans le coin supérieur gauche. Différents symboles sont utilisés pour les différents types de dégradation, permettant ainsi de distinguer les points forts des applications :

Bruit blanc	+
Flou Spectral	*
Flou Spatial	×
Effet de Gibbs	◇
Compression JPEG2000	□
Déréglage (Fig.2.9-2.10)	△

Un critère qualité répond correctement s'il donne une faible dispersion des points, si possible selon une direction oblique (Fig. 2.8). Si on obtient ce cas de figure, cela signifie que lorsque les performances de l'application baissent (i.e. l'ordonnée diminue), le critère qualité en rend bien compte et dans une bonne proportion. Au contraire, si les points sont dispersés, cela montre que le critère étudié va réagir plus fortement à certaines dégradations par rapport à d'autres (Fig. 2.7). Dans le cas où la réponse est concentrée le long d'un axe vertical (Fig. 2.7 courbe 1), c'est que le critère ne réagit pas à des dégradations qui sont pénalisantes pour l'application étudiée, il sous-estime l'importance de cette dégradation. À l'inverse, lorsque la réponse est concentrée le long d'un axe horizontal (Fig. 2.7 courbe 2), le critère surestime l'impact de la dégradation sur l'application.

2.3.2 Interprétation graphique et sensibilité des applications

Parmi les applications envisagées initialement (détection d'anomalies, classifications et calcul des valeurs propres), certaines s'avèrent ne pas avoir les propriétés nécessaires (stabilité, monotonie) pour conclure.

En effet, les performances de la détection sont très instables et dépendent fortement et irrégulièrement de la dégradation appliquée. Par exemple, dans le cas du bruit blanc (+), les performances sont très sensibles à la réalisation du bruit (Fig. 2.9). Cela peut s'expliquer par le fait qu'il n'y a qu'un nombre faible de cibles insérées dans l'image (environ 100 pour 10 000 pixels), des résultats statistiques sont difficilement fiables avec un nombre aussi faible d'échantillons. D'un autre côté, comme l'application est une application de détection d'anomalies, il n'est pas possible d'insérer un plus grand nombre de cibles. Les conclusions vont donc s'appuyer principalement sur les per-

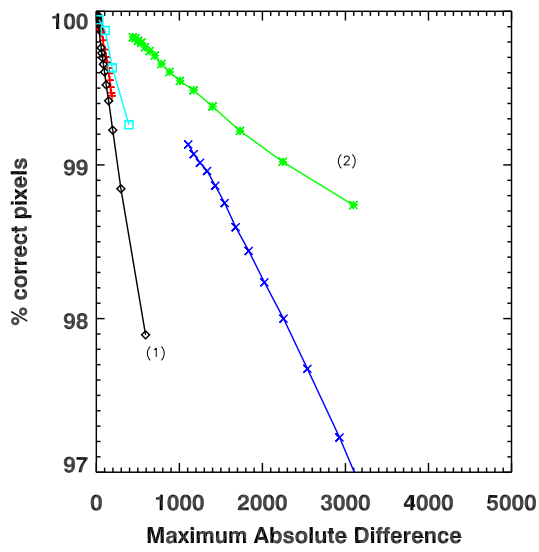


Fig. 2.7 MAD (Eq. 2.13) par rapport à la classification par SAM : sous estimation de l'effet de Gibbs (1) et surestimation du flou spectral (2). Bruit blanc (+), flou spectral (*), flou spatial (x), effet de Gibbs (◇) et compression JPEG2000 (□).

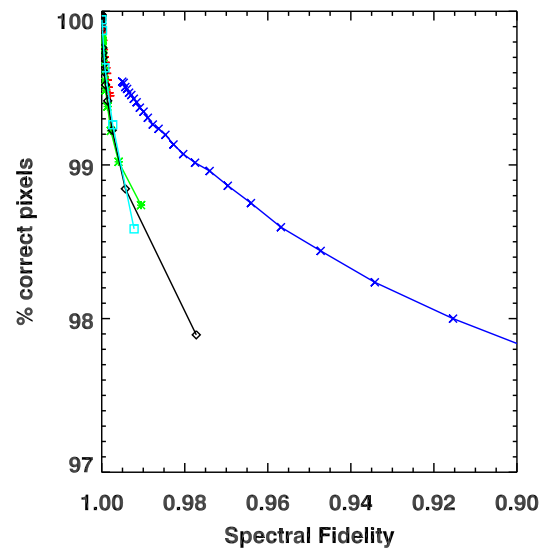


Fig. 2.8 F_λ (Eq. 2.38) par rapport à la classification par SAM : la dispersion pour toutes les dégradations est faible. La réaction de ce critère est semblable à l'application de classification : ce critère donne une estimation fiable de l'influence des dégradations. Bruit blanc (+), flou spectral (*), flou spatial (x), effet de Gibbs (◇) et compression JPEG2000 (□).

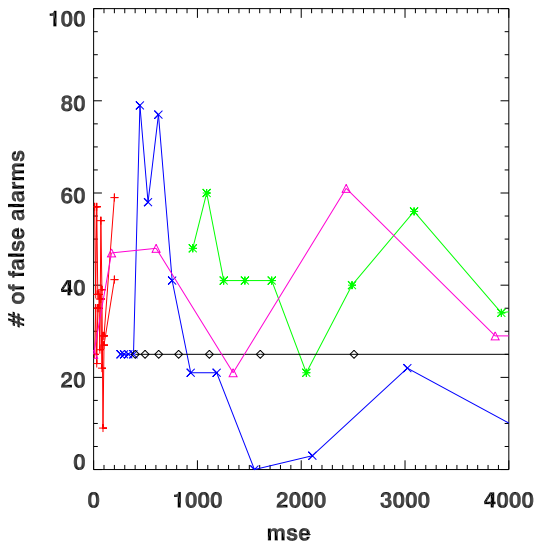


Fig. 2.9 MSE par rapport à la détection : les résultats sont très instables.

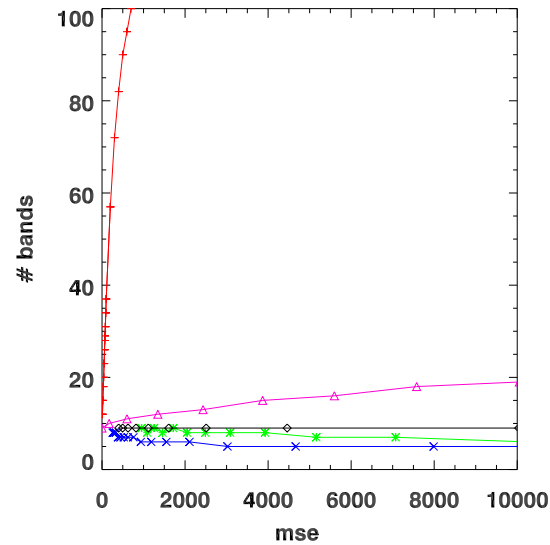


Fig. 2.10 MSE par rapport au nombre de composantes de l'ACP.

performances de l'application de classification qui sont beaucoup plus fiables et qui dépendent peu des différentes réalisations des dégradations aléatoires.

Pour l'analyse des résultats concernant l'ACP, on ne peut pas tirer de conclusion directe sur la performance des applications qui viendront après, mais juste avoir une idée de l'impact de la dégradation. On remarque principalement que l'ACP est très sensible à la présence de bruit blanc, moins sensible à la présence de flou, et complètement insensible à un phénomène de Gibbs (Fig. 2.10).

Ainsi, nos conclusions quant au choix de critères qualité vont s'appuyer essentiellement sur les résultats des méthodes de classification.

Les résultats pour la classification sont étudiés pour une même méthode de classification (SAM) sur deux images différentes pour s'assurer que les résultats seront applicables à d'autres images. Les deux images choisies sont deux parties de l'image Moffett acquise par AVIRIS. La première est une zone plutôt uniforme, tandis que la deuxième est une zone de ville très découpée.

D'autre part, pour vérifier que les résultats sont généralisables à d'autres méthodes de classification, les mesures sont effectuées pour la même image avec trois méthodes de classification différentes : SAM, Mahalanobis et Maximum de vraisemblance. Les résultats pour les classifications de Mahalanobis et Maximum de vraisemblance sont très similaires : très grande sensibilité à la présence de bruit blanc, peu de sensibilité au flou spectral, mais présence d'un effet de seuil important. En revanche, la classification par SAM est très peu sensible au bruit blanc et ne présente pas cet effet de seuil sur un flou

spectral (Tab. 2.1) .

	Bruit Blanc	Flou Spect.	Flou spat.	Gibbs	JPEG 2000.
ACP	***	*	**	0	**
SAM	*	*	**	**	*
Mahalanobis	***	0/**	**	*	**
Max Vrais.	**	0/***	**	**	**

Tab. 2.1 Sensibilité des applications aux bruits simulés. 0 : pas de sensibilité ou très faible; * : peu sensible; ** : sensible; *** : extrêmement sensible. 0/** montre un effet de seuil : insensible avant le seuil et très sensible après.

La structure des erreurs de classification est assez caractéristique lorsqu'il n'y a pas de classe de rejet. Même avec l'introduction d'un bruit blanc, les erreurs ne sont pas réparties aléatoirement sur l'image, elles se concentrent aux limites entre les zones. En effet, les pixels se trouvant entre deux zones ont de fortes chances d'être des pixels de mélange et possèdent des spectres moins caractéristiques. Ce sont donc ces pixels qui vont changer de classe en premier. Dans le cas de la classification par SAM, la présence de la classe de rejet permet d'atténuer cet effet.

2.3.3 Interprétation des critères de qualité

Les critères de qualité testés sont :

- MSE (Eq. 2.4), RRMSE (Eq. 2.8), MAD (Eq. 2.13), PMAD (Eq. 2.14), MAE (Eq. 2.3) qui sont calculés de manière globale sur l'image ;
- SAM (Eq. 2.22), SS (Eq. 2.26), SID (Eq. 2.43) et Pearson (Eq. 2.28) qui sont calculés pour chaque pixel (vu comme un vecteur). La valeur retournée pour l'image est la valeur maximum (pour SAM, SS et SID) ou la valeur minimum (pour Pearson) ;
- Les critères Q (Eq. 2.31) et F (Eq. 2.36) sont adaptés de différentes manières : ils sont soit calculés pour chaque pixel (vu comme un vecteur), la valeur finale étant le minimum entre les différents vecteurs (Eq. 2.38 et Eq. 2.33), soit calculés pour chaque bande spectrale, la valeur finale étant le minimum entre les différentes bandes (Eq. 2.39 et Eq. 2.34).

Les résultats obtenus sont très hétérogènes, certains critères surestimant ou sous-estimant beaucoup l'impact de certaines dégradations. L'impact du flou spectral (*) par exemple est largement surestimé par la plupart des critères : tous les critères (Fig. 2.11 par exemple) sauf F_λ surestiment cet impact. Dans le cas de la classification par Mahalanobis, le flou spectral (*) provoque un effet de seuil (Fig. 2.12). Certains critères sont très sensibles à la présence de bruit blanc (+) (RRMSE, PMAD, MAE et SID), alors que la classification par SAM n'y réagit quasiment pas (Fig. 2.13). $Q_{(x,y)}$ est complètement insensible aux dégradations touchant les plans spatiaux :

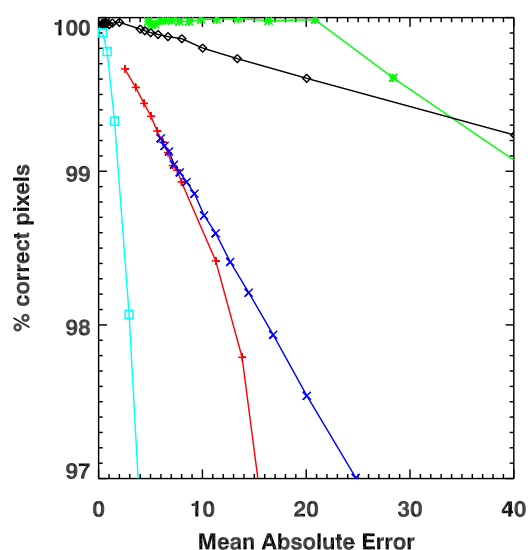


Fig. 2.11 MAE (Eq. 2.3) par rapport à la classification de Mahalanobis : le MAE est le critère le plus sensible à la présence de l'effet de Gibbs. Bruit blanc (+), flou spectral (*), flou spatial (x), effet de Gibbs (◇) et compression JPEG2000 (□).

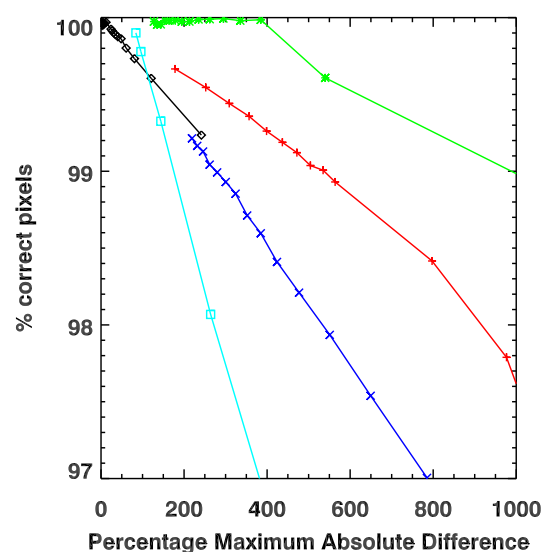


Fig. 2.12 PMAD (Eq. 2.14) par rapport à la classification de Mahalanobis : apparition d'un seuil pour le flou spectral. Bruit blanc (+), flou spectral (*), flou spatial (x), effet de Gibbs (◇) et compression JPEG2000 (□).

aucune sensibilité à un phénomène de Gibbs (◇) ou à un flou spatial (x), bien que ces dégradations déforment également le spectre (Fig. 2.14).

Deux types de critères peuvent être intéressants. Un critère réagissant uniformément à toutes les dégradations de la même manière que les applications serait idéal, mais comme on l'a vu dans la partie précédente, toutes les applications ne réagissent pas de la même manière. D'un autre côté, les critères réagissant fortement ou pas du tout à une dégradation sont intéressants pour permettre de qualifier la nature de la dégradation.

2.3.4 Évaluation quantitative

La représentation graphique des résultats permet de découvrir les tendances dominantes parmi les critères considérés. Ce n'est cependant pas suffisant, des comparaisons quantitatives sont nécessaires. Deux problèmes se posent pour cela. Les différents critères n'ont pas d'échelle commune entre eux et donc ne sont pas directement comparables. Appliquer une normalisation paraît difficile à cause du choix arbitraire du niveau de référence qui serait nécessaire. Ensuite, la sensibilité des différents critères à une dégradation n'est définie que par rapport aux autres dégradations. Par exemple, dire que le RRMSE est très sensible au flou spectral n'a de sens que parce qu'il est moins sensible relativement aux autres dégradations (Fig. 2.13). Pour sur-

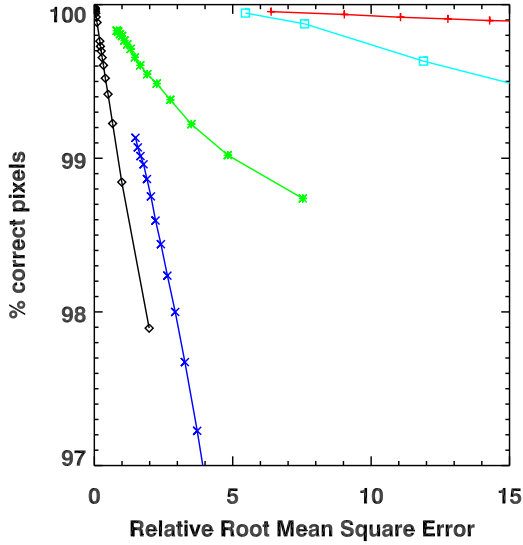


Fig. 2.13 RRMSE (Eq. 2.8) par rapport à la classification par SAM : le RRMSE surestime la dégradation en présence de bruit blanc. Bruit blanc (+), flou spectral (*), flou spatial (x), effet de Gibbs (\diamond) et compression JPEG2000 (\square).

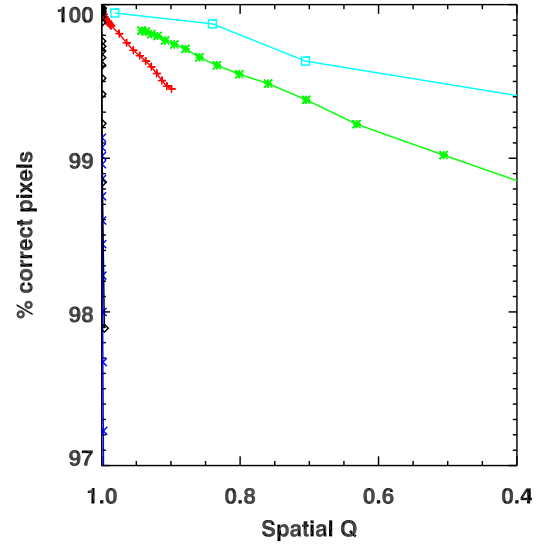


Fig. 2.14 $Q_{(x,y)}$ (Eq. 2.34) par rapport à la classification par SAM : aucune sensibilité à la présence de flou spatial. Bruit blanc (+), flou spectral (*), flou spatial (x), effet de Gibbs (\diamond) et compression JPEG2000 (\square).

monter ces difficultés, c'est la contribution relative de chaque dégradation qui est calculée.

Pour chaque critère de qualité C , une valeur moyenne du critère est calculée pour chaque dégradation. Chaque critère est donc caractérisé par \overline{C}_{BB} , $\overline{C}_{FlouSpect.}$, $\overline{C}_{FlouSpat.}$, \overline{C}_{Gibbs} et $\overline{C}_{JPEG2000}$ où la notation $\bar{\cdot}$ correspond à la valeur moyenne et l'indice indique le type de dégradation. Pour trouver le critère le plus sensible à chaque dégradation, les valeurs moyennes précédentes sont remplacées par leurs contributions relatives, i.e. $\overline{C}_{BB}/\overline{C}$, $\overline{C}_{FlouSpect.}/\overline{C}$, ... où

$$\overline{C} = \overline{C}_{BB} + \overline{C}_{FlouSpect.} + \overline{C}_{FlouSpat.} + \overline{C}_{Gibbs} + \overline{C}_{JPEG2000}. \quad (2.52)$$

Ces contributions relatives sont calculées pour chaque critère ainsi que pour diverses méthodes de classification. Les résultats sont donnés dans les tableaux 2.2 et 2.3.

Les valeurs signifient par exemple que la contribution dans le RRMSE (sur l'image moffett4, dans le tableau 2.3) :

- du bruit blanc est de 51.54%,
- du flou spectral est de 4.39%,
- du flou spatial est de 8.05%,
- de l'effet de Gibbs est de 0.86%,

	BruitB.	Fl.Spect.	Fl.Spat.	Ef.Gibbs	J2K
MSE	0.88%	12.19%	80.55%	2.77%	3.61%
MAD	0.73%	20.57%	71.48%	2.09%	5.13%
PMAD	24.31%	6.63%	16.31%	2.01%	50.73%
MSS	0.40%	7.93%	86.72%	2.51%	2.44%
RRMSE	44.15%	4.33%	6.63%	0.39%	44.50%
MSA	3.72%	10.06%	68.96%	7.13%	10.13%
MSID	9.40%	0.69%	50.84%	1.26%	37.81%
Pearson	0.27%	5.02%	88.68%	0.88%	5.15%
MAE	8.64%	18.49%	48.75%	11.42%	12.71%
$Q_{(x,y)}$	0.31%	42.82%	0.70%	0.02%	56.14%
Q_λ	0.05%	1.03%	97.11%	0.83%	0.99%
Q_m	0.24%	30.92%	28.24%	0.25%	40.35%
F	0.90%	12.20%	80.47%	2.79%	3.63%
F_λ	0.05%	0.34%	98.41%	0.52%	0.68%
$F_{(x,y)}$	2.59%	35.12%	15.77%	1.49%	45.03%

Tab. 2.2 Sensibilité des critères par rapport aux applications (sur moffett3).

	BruitB.	Fl.Spect.	Fl.Spat.	Ef.Gibbs	J2K
MSE	2.94%	21.62%	62.30%	5.40%	7.74%
MAD	1.53%	24.81%	64.60%	2.70%	6.36%
PMAD	23.43%	10.59%	19.20%	1.70%	45.09%
MSS	0.97%	9.69%	81.95%	2.98%	4.41%
RRMSE	51.54%	4.39%	8.05%	0.86%	35.16%
MSA	5.38%	9.42%	67.16%	6.74%	11.31%
MSID	17.08%	0.69%	48.85%	1.19%	32.18%
Pearson	0.68%	2.82%	85.58%	0.87%	10.06%
MAE	17.43%	22.30%	29.07%	14.84%	16.36%
$Q_{(x,y)}$	2.50%	35.57%	0.24%	0.02%	61.66%
Q_λ	0.14%	0.60%	95.80%	1.36%	2.11%
Q_m	2.20%	31.04%	12.74%	0.20%	53.82%
F	2.99%	21.61%	62.18%	5.44%	7.78%
F_λ	0.20%	0.61%	96.10%	0.85%	2.23%
$F_{(x,y)}$	2.38%	30.57%	21.12%	1.24%	44.69%
SAM cl.	4.24%	11.39%	60.37%	12.03%	11.97%
Mahal.	12.42%	0.41%	19.95%	1.57%	65.66%
Max.Like.	10.73%	0.01%	10.33%	2.76%	76.17%

Tab. 2.3 Sensibilité des critères par rapport aux applications (sur moffett4). Les minima et maxima sélectionnés pour chaque dégradation sont reportés en gras (la sélection est faite par rapport aux résultats sur les deux images). Lorsque deux critères sont proches de l'extremum, un seul est choisi de manière à minimiser le nombre total de critères.

- de JPEG 2000 est de 35.16%.

Le total est bien sûr de 100%. Ces valeurs dépendent du choix arbitraire des amplitudes des distorsions et doivent donc être comparées aux autres critères ainsi qu'aux applications. Pour le RRMSE, la comparaison avec la classification par SAM montre que le RRMSE surestime l'impact du bruit blanc (51.54% contre 4.24% d'influence pour la classification par SAM) et sous-estime l'effet de Gibbs (0.86% contre 12.03% pour la classification par SAM), confirmant l'interprétation graphique que l'on pouvait avoir sur la figure 2.13. Les extrema en gras dans le tableau 2.3 sont sélectionnés en fonction des résultats sur les deux images.

2.3.5 Critères de qualité proposés

Certains critères permettent de bien rendre compte de l'importance des dégradations causées quel que soit le type de ces dégradations. C'est le cas notamment du critère F_λ particulièrement bien en phase avec la classification par SAM. Toutefois, le critère reflétant le mieux les résultats de la classification par SAM est logiquement le critère MSA mais par contre, il ne permet pas de retrouver les sensibilités des autres applications.

On voit ainsi que les différentes applications envisagées réagissent différemment aux mêmes dégradations. Il n'est donc pas possible de sélectionner ou de définir un seul critère *magique* qui refléterait toutes les applications. Une combinaison d'un certain nombre de critères est nécessaire pour rendre compte de manière complète de l'influence des dégradations. Nous proposons de ne garder que 4 critères qui vont permettre de quantifier et de qualifier la dégradation sur l'image. Ces 4 critères sont : F_λ , RRMSE, MAE et $Q_{(x,y)}$ (Tab. 2.4). Ces critères sont sélectionnés comme étant les extrema de sensibilité pour les différentes dégradations. Pour renforcer cet ensemble de critères, le MAD est ajouté pour permettre de donner une borne supérieure à l'erreur sur l'ensemble du cube hyperspectral. Connaître l'erreur maximale est intéressant particulièrement dans le cas de la détection d'anomalies, pour s'assurer de ne pas rajouter ou d'éliminer des cibles.

	+ sensible	- sensible
Bruit blanc	RRMSE	F_λ
Flou spectral	$Q_{(x,y)}$	F_λ
Flou spatial	F_λ	$Q_{(x,y)}$
Effet de Gibbs	MAE	$Q_{(x,y)}$
JPEG 2000	$Q_{(x,y)}$	F_λ

Tab. 2.4 Critères sélectionnés pour leur sensibilité. A cet ensemble, on ajoute le MAD qui permet de borner l'erreur sur l'ensemble du cube.

2.4 Conclusion

Parmi les 15 critères de qualité étudiés, 5 ont été sélectionnés pour leurs propriétés. Comme nous l'avons vu, les 5 critères sélectionnés peuvent être utilisés pour caractériser la nature et le niveau de dégradation affectant une image ainsi que pour prédire les performances d'une application donnée. Cependant, seules des applications de classification ont fourni des résultats utilisables ici, la détection d'anomalies donnant des résultats trop aléatoires et l'analyse en composantes principales des résultats difficilement interprétables. Il pourrait être intéressant d'utiliser la même méthode pour d'autres applications : indice de végétation (NDVI), qualité de l'eau...

Le but de cette étude sur les critères qualité était de trouver des critères pertinents pour diverses applications hyperspectrales. Une méthode générale est définie pour pouvoir comparer la sensibilité de divers critères par rapport aux applications.

De plus cette étude démontre la **nécessité d'utiliser une combinaison de plusieurs critères plutôt qu'un seul**, chacun ayant une sensibilité différente selon les dégradations.

Dans la suite de cette étude, on va s'intéresser à la compression des images hyperspectrales. Afin de pouvoir comparer les résultats des méthodes de compression que nous allons proposer à des méthodes existantes, nous utiliserons des critères communs dans la littérature : PSNR ou SNR essentiellement. Ce n'est qu'en fin d'étude, lorsque nous aurons proposé une méthode de compression intéressante par rapport à celles existantes, que nous illustrerons les résultats en termes des critères sélectionnés dans le présent chapitre.

Chapitre 3

Compression des images hyperspectrales

3.1	JPEG 2000 pour l'hyperspectral	62
3.1.1	La norme JPEG 2000	62
3.1.2	Adaptation aux images hyperspectrales	63
3.2	Choix de la décomposition optimale	66
3.2.1	Décomposition anisotropique 3D en ondelettes	66
3.2.2	Optimisation débit-distorsion	69
3.2.3	Résultats sur les images 2D	74
3.2.4	Décomposition en ondelettes optimale pour l'hyperspectral	76
3.2.5	Décomposition fixe	76
3.3	Structures d'arbres	78
3.3.1	Idées principales	78
3.3.2	Principes généraux de EZW et SPIHT	81
3.3.3	Étude statistique pour le choix de la structure d'arbre	82
3.3.4	Adaptation de EZW	86
3.3.5	Adaptation de SPIHT	89
3.4	Comparaison des résultats	92
3.5	Variations avec la notation binaire signée	95
3.5.1	Un inconvénient de EZW	95
3.5.2	Utilisation de la notation binaire signée	95
3.5.3	Utiliser les dépendances spatiales	98
3.6	Conclusion	100



REVENONS maintenant sur la compression avec ce chapitre qui présente l'adaptation des méthodes de transformée en ondelettes pour les propriétés spécifiques de l'hyperspectral. Une première partie développe les possibilités du standard de compression JPEG 2000. La seconde partie s'attache à trouver la transformation en ondelettes optimale pour les images hyperspectrales. Enfin, une adaptation des structures d'arbres de zéros est faite pour appliquer les algorithmes EZW et SPIHT aux images hyperspectrales. Cette partie de l'étude a fait l'objet de publications [Chr06a, Chr06b, Chr06c, Chr07a, Chr07b].

3.1 JPEG 2000 pour l'hyperspectral

3.1.1 La norme JPEG 2000

JPEG 2000 est la norme la plus récente pour la compression d'image fixe. Une documentation complète est disponible sur le sujet, on peut voir par exemple [Tau02, Sko01]. Cette partie propose uniquement une présentation des points clés de la norme.

Cette norme a été définie pour fournir un cadre à une grande variété d'applications compressant les images avec différentes caractéristiques (images naturelles, images scientifiques multicomposantes, texte codé sur deux niveaux. . .) pour différentes utilisations (transmission en temps réel, archive, ressources limitées. . .).

Les principales fonctionnalités de JPEG 2000 sont, entre autres, le tuilage d'images de grande taille, le codage de régions d'intérêt (region of interest ou ROI) tout en proposant le codage d'images de dynamiques variées avec différents nombres de composantes. La partie 1 du standard [ISO02] comprend la plupart des fonctionnalités de JPEG 2000 pouvant être utilisées sans payer de royalties ou de frais de licence. La partie 2 contient les extensions [ISO04]. La plupart des implémentations actuelles de JPEG 2000 correspondent uniquement la partie 1.

Le codeur JPEG 2000 est basé sur le modèle de codage par transformée. Pour coder une image à une composante, une transformation multirésolution en ondelettes est d'abord appliquée. L'annexe A concerne les principaux aspects de l'utilisation de la transformée en ondelettes. Dans le cas d'une compression sans pertes, c'est l'ondelette Le Gall 5/3 qui est utilisée, sinon c'est l'ondelette de Daubechies 9/7. Pour la compression sans pertes, c'est une transformation d'entiers en entiers basée sur une implémentation en *lifting scheme* de la transformée en ondelettes [Dau98]. Ces transformations ont été choisies pour leur capacité à effectuer une décorrélation efficace d'une image.

La partie codage entropique de JPEG 2000 est assez complexe pour respecter la flexibilité demandée par la norme. Pour permettre le codage pro-

gressif ainsi que la résistance aux erreurs de transmission, le codage est basé sur un codage par blocs avec points de troncatures optimaux (Embedded Block Coding with Optimal Truncation ou EBCOT) [Tau00]. EBCOT est la raison principale des excellentes performances en terme de débit-distorsion de cette norme. Chaque sous-bande issue de la transformée en ondelettes est partagée en blocs de taille relativement faible (64×64 pixels en général). Chacun de ces blocs est ensuite codé indépendamment pour produire un train binaire progressif. Une quantification scalaire est utilisée pour la partie 1 de la norme tandis que la quantification vectorielle est disponible dans les extensions. Le codage entropique est réalisé par un codeur arithmétique contextuel. Les symboles sont codés à partir de modèles de probabilités pour 18 contextes différents. La norme ne spécifie pas d'allocation de débit et donc sur ce point, l'utilisateur est libre de faire ce qu'il veut. Néanmoins, une méthode populaire et largement utilisée est l'algorithme PCRD-opt (Post Compression Rate Distortion optimization) qui détermine le point optimal de troncature pour chacun des blocs [Tau00].

Pour les images multicomposantes à trois bandes (une image couleur classique par exemple), une transformation standard entre les couleurs est appliquée avant le codage par JPEG 2000. Cette transformation convertit l'espace naturel RGB (Red Green Blue) dans un espace moins corrélé YCrCb (une luminance et deux chrominances). Ensuite, le codeur JPEG 2000 classique est appliqué à chaque composante. L'algorithme d'allocation de débit (PCRD-opt) est ensuite utilisé de manière globale pour répartir le débit entre toutes les composantes.

3.1.2 Adaptation aux images hyperspectrales

Les parties 1 et 2 de la norme JPEG 2000 sont destinées au codage des images fixes en niveau de gris ou des images avec 3 bandes de couleurs et éventuellement un quatrième canal alpha (pour spécifier la transparence). Dans ces parties, aucune transformation intercomposante n'est définie à l'exception de la transformation couleur. Cependant, la partie 2 prévoit l'utilisation de transformations intercomposantes pouvant être précisées par l'utilisateur, avec, entre autres, la transformation en ondelettes. La partie 10, également connue sous le terme de JP3D, vise plutôt les images tridimensionnelles aussi isotropiques que possible [wg104]. Ces spécifications ne conviennent pas aux images hyperspectrales qui ne sont pas isotropiques et ont, par exemple, une corrélation bien plus forte dans la direction spectrale que dans les directions spatiales. Par conséquent, la partie JP3D du standard ne convient pas à la compression des images hyperspectrales. Nous proposons donc plutôt d'utiliser les extensions de la norme (partie 2) en introduisant l'utilisation de transformées intercomposantes avant d'appliquer le codage JPEG 2000.

L'implémentation de référence de JPEG 2000 est le *Verification Model*

(VM). Le VM est utilisé par le comité JPEG 2000 pour les expérimentations et évoluera au final en implémentation de la norme JPEG 2000 [wg101]. Pour cette étude, la version VM 9.1, la plus récente, est utilisée pour évaluer les performances de compression de la norme JPEG 2000 adaptée aux images hyperspectrales. Une correction est nécessaire pour pouvoir décompresser les trains binaires obtenus en utilisant la transformée intercomposante, cette correction est détaillée dans l'annexe C. Une autre implémentation récente et populaire de JPEG 2000, *Kakadu*, version 5.0 est également utilisée pour confirmer les résultats. Le détail des lignes de commande et des options utilisées est donné dans l'annexe C.

Comme il a été précisé, les parties 1 et 2 de la norme ne donnent pas de spécifications pour les cas où l'image possède plus de 3 bandes (transformation couleur). Cependant, le VM permet à l'utilisateur de fournir une matrice de transformation permettant une transformée en cosinus discrète (DCT) ou une transformée de Karhunen-Loeve (KLT). Cela est fait en utilisant l'option *-Mlin* du VM. Une autre option (*-Mtdt*) permet d'appliquer une transformée en ondelettes (DWT 9/7) entre les différentes composantes. Dans ces deux cas, la transformation 1D est appliquée sur la dimension spectrale avant d'utiliser l'encodage classique JPEG2000 sur les images résultantes. Une optimisation débit-distorsion par lagrangien est une option utile car les composantes ont des propriétés statistiques différentes après la compaction d'énergie réalisée par la transformation (option *-Flra*). Sans cette option, le VM a du mal à atteindre le débit visé sur certaines images.

La figure 3.1 compare les performances de plusieurs transformations intercomposantes : DCT, KLT et DWT en terme de PSNR (Eq. 2.11) en fonction du nombre de bits par pixel par bande (bpppb).

Introduire une décorrélation intercomposante des données avant le codage JPEG 2000 permet d'augmenter de manière importante les performances. La KLT donne les meilleurs résultats car elle est adaptée aux données et est optimale en terme de décorrélation et de concentration d'énergie. La DWT augmente significativement les résultats tandis que la DCT présente des performances plus faibles.

La KLT est spécifique à chaque image et les matrices de transformation doivent donc être recalculées en fonction des données. Ce calcul est coûteux en terme de complexité : il est nécessaire de calculer la matrice de covariance ainsi que les vecteurs propres. Cette complexité est dissuasive pour la plupart des applications particulièrement lorsque le nombre de bandes augmente. Dans le cas des images multispectrales une approche utilisant une KLT générique, calculée sur plusieurs images est envisagée [Thi06]. Dans ce cas, il n'est plus nécessaire d'effectuer la diagonalisation de la matrice de covariance, il suffit d'effectuer la transformation. Avec 224 bandes, l'opération reste encore impossible en implémentation matérielle comme le montre le tableau 3.1. L'approche par transformation en ondelettes paraît plus adaptée, surtout dans le cas hyperspectral où le nombre de composantes est impor-

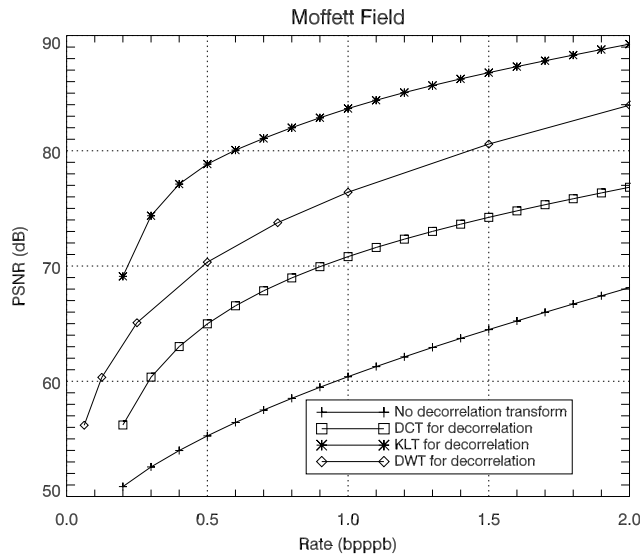


Fig. 3.1 Performances de la compression JPEG 2000 en utilisant de la décorrélation interbande par DCT, KLT, DWT ou sans décorrélation.

# bandes	Surface (mm^2)
16	6
64	99
128	400
256	1500

Tab. 3.1 Surface de silicium nécessaire pour l'implémentation matérielle (ASIC) de la transformée KLT (sans le calcul de la matrice de transformation) en fonction du nombre de bandes spectrales. La limite actuelle est de l'ordre de $110mm^2$ [Alc06].

tant.

L'algorithme JPEG 2000 nécessite des ressources importantes (principalement pour le codeur arithmétique et l'allocation de débit) ce qui n'est pas possible particulièrement dans le contexte d'une implémentation spatiale où les contraintes sont très fortes. A notre connaissance, il n'existe qu'une seule implémentation de JPEG 2000 adaptée aux contraintes spatiales [VB05]. L'auteur de cette implémentation indique que les performances sont significativement moins bonnes que *Kakadu* dans le cas de la compression avec pertes à cause de l'impossibilité d'implémenter l'optimisation débit-distorsion. Le Consultative Committee for Space Data Systems (CCSDS), un groupe de travail regroupant les principales agences spatiales mondiales (NASA, JAXA, ESA, CNES) a émis des recommandations pour les systèmes de compression bord [Yeh05]. La recommandation consiste à regrouper les coefficients de la transformée en ondelettes dans une structure similaire à celle des arbres de zéros plutôt que d'utiliser la norme JPEG 2000. Le but de cette implémentation est d'alléger la complexité vis-à-vis de la norme

JPEG 2000. Les performances de JPEG 2000 dans notre étude sont à voir plutôt comme une référence à atteindre, le but étant ici d'obtenir des performances proches de cette référence tout en gardant une complexité faible. La version de JPEG 2000 au fil de l'eau (*scan-based mode*), plus réaliste pour une implémentation spatiale donne elle-même des performances réduites par rapport à la référence.

La réduction de complexité par rapport à JPEG 2000 n'est pas obtenue directement par l'utilisation de la transformation en ondelettes selon les 3 directions, mais il apparaît que les ondelettes permettent l'utilisation d'outils performants et simples. C'est donc la transformée en ondelettes qui est choisie pour la décomposition.

3.2 Choix de la décomposition optimale

Avant d'adapter les arbres de zéros aux images hyperspectrales, il est nécessaire de définir une extension de la transformation en ondelettes performante pour les images hyperspectrales. La plupart des extensions actuelles sont basées sur des décompositions isotropiques [Tan03, Kim03], or, comme il a été montré dans les chapitres précédents, les données hyperspectrales ne sont clairement pas isotropiques. Dans le domaine de la compression vidéo, des structures anisotropiques sont utilisées avec succès [He03, Cho03]. Cependant, aucune justification théorique n'a été donnée concernant le choix de cette structure particulière et des décompositions plus efficaces pourraient exister. De toute façon, le meilleur choix pour les données vidéo n'est pas nécessairement le meilleur pour les images hyperspectrales à cause des propriétés statistiques différentes.

Le problème de la recherche de la décomposition en ondelettes optimale pour signaux à une dimension a été exploré dans plusieurs publications (par exemple [Coi90]). Pour les images naturelles 2D, les possibilités de décomposition ont souvent été restreintes à des *quadrees* (conduisant à des sous-bandes carrées) mais ont évolué avec les décompositions anisotropiques [Xu03]. Plusieurs critères ont été utilisés pour choisir la décomposition optimale : sélection basée sur l'entropie [Coi92] ou sur un compromis débit-distorsion [Ram93] par exemple. L'avantage principal du dernier choix est qu'il propose simultanément l'allocation de débit entre les différentes sous-bandes [Sho88].

3.2.1 Décomposition anisotropique 3D en ondelettes

Classiquement, pour les images 2D, la transformée en ondelettes est isotropique i.e. pour une sous-bande donnée, le niveau de décomposition dans la direction horizontale est le même que dans la direction verticale. Cette alternance entre décomposition des lignes et des colonnes conduit à des sous-

bandes carrées ¹, l'équivalent en 3D étant des cubes. C'est le cas de la décomposition multirésolution définie par Mallat ou de la décomposition en paquets d'ondelettes pour les images [Mal89]. Un exemple de la décomposition multirésolution classique est illustré sur la figure 3.2. Les coefficients gris représentent des valeurs proches de zéros, tandis que les noirs et blancs correspondent respectivement à des valeurs fortement négatives ou positives.

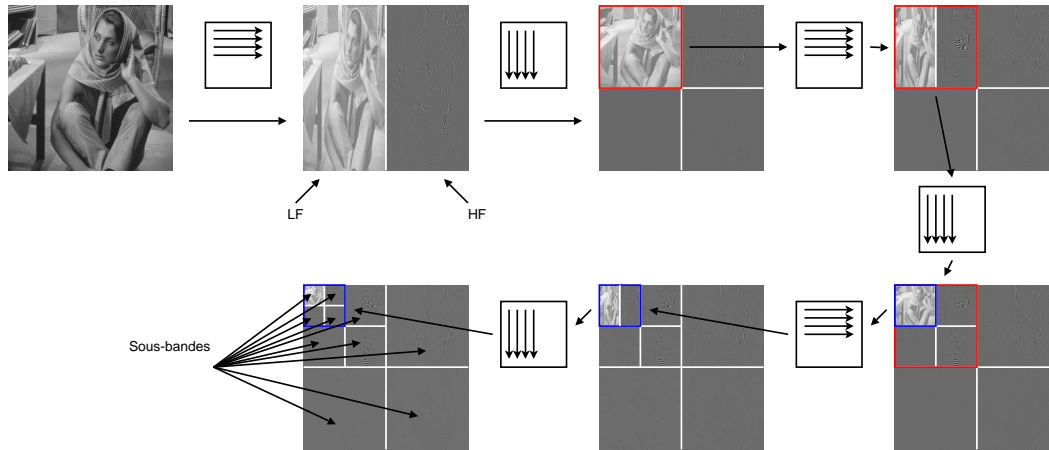


Fig. 3.2 Décomposition multirésolution classique : on décompose successivement suivant les lignes et les colonnes. LF est la partie basse fréquence (LF = *low frequency*) et HF la partie haute fréquence (HF = *high frequency*).

Le terme anisotropique est plus général que l'utilisation courante du terme *paquets d'ondelettes*. Dans la plupart des cas, le terme *paquets d'ondelettes* désigne une transformation en *quadtree* conduisant à des sous-bandes carrées. Cette utilisation est justifiée par le fait que les images 2D classiques ont des propriétés statistiques similaires dans toutes les directions. Un exemple de décomposition anisotropique est donné sur la figure 3.3. On s'aperçoit ainsi qu'une telle décomposition non isotropique permet d'isoler une sous bande contenant beaucoup d'énergie (Fig. 3.3, sous-bande 1), tandis que les autres sous-bandes contiennent une majorité de coefficients nuls (la sous-bande 2 par exemple). Ceci permettra un codage efficace de l'image.

On note $W_{i,j,k}^{p,q,r}$ une sous-bande de la décomposition 3D en ondelettes (Fig. 3.4) :

- i, j, k étant le niveau de décomposition selon, respectivement, les lignes, les colonnes et les spectres (déterminant la taille de la sous-bande).
- p, q, r étant respectivement l'index de ligne, de colonne et de spectre.

Cette notation est illustrée sur la décomposition multirésolution pour une image 2D dans l'annexe A.

Une relation peut être définie au niveau des espaces vectoriels des sous-

¹pourvu que l'image de départ soit carrée, sinon, les sous-bandes ont la même proportion hauteur/largeur

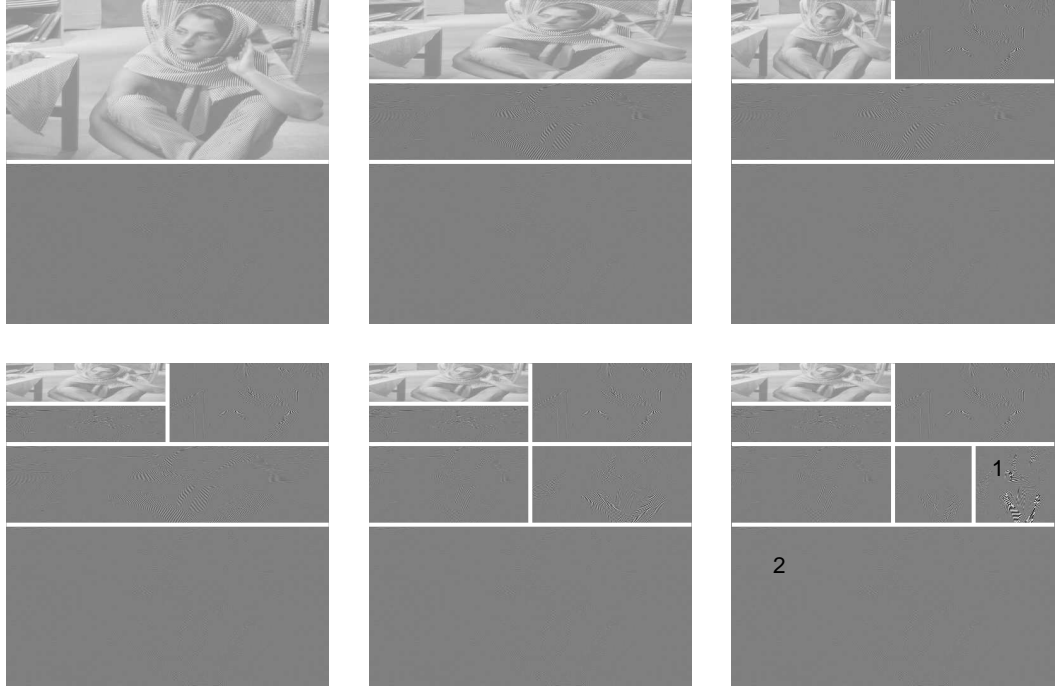


Fig. 3.3 Exemple de décomposition anisotrope.

bandes. Pour une décomposition selon les lignes, l'espace des ondelettes anisotropiques vérifie

$$W_{i,j,k}^{p,q,r} = W_{i+1,j,k}^{2p,q,r} \oplus W_{i+1,j,k}^{2p+1,q,r} \quad (3.1)$$

où \oplus est la somme directe de deux espaces vectoriels.

Pour une décomposition selon les colonnes, on a

$$W_{i,j,k}^{p,q,r} = W_{i,j+1,k}^{p,2q,r} \oplus W_{i,j+1,k}^{p,2q+1,r} \quad (3.2)$$

et selon les spectres

$$W_{i,j,k}^{p,q,r} = W_{i,j,k+1}^{p,q,2r} \oplus W_{i,j,k+1}^{p,q,2r+1} \quad (3.3)$$

À chaque étape de la décomposition, pour toutes les sous-bandes, il est possible de choisir la direction de la décomposition ce qui accroît la flexibilité de l'espace des décompositions. La décomposition multirésolution et les décompositions en paquets d'ondelettes sont toutes deux des cas particuliers de cette représentation.

Une représentation simple et bien adaptée pour ce type de décomposition peut être faite sous la forme d'arbre (Fig. 3.5 et 3.6). Chaque nœud porte un numéro, indiquant le sens de la décomposition (x , y ou λ), et deux branches, correspondant aux deux sous-bandes.

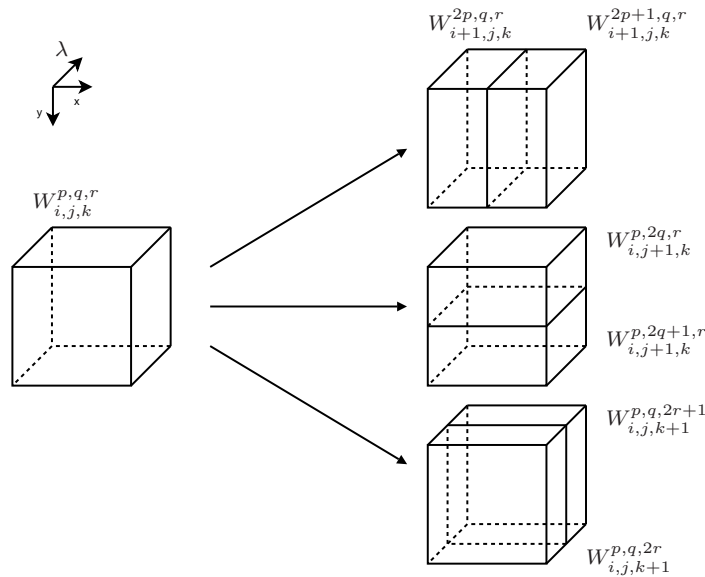


Fig. 3.4 Décomposition anisotrope et notations.

Il faut noter que la correspondance arbre-décomposition n'est pas bijective. Deux arbres différents peuvent donner la même décomposition. Par exemple, décomposer sur x puis sur *chacune* des deux sous-bandes décomposer sur y donne la même décomposition que décomposer d'abord sur y puis sur x pour *chacune* des deux sous-bandes. Les arbres correspondant à ces deux transformations seront différents. Par contre, un arbre donne bien une seule décomposition possible.

La représentation sous forme d'arbre se prête particulièrement bien à une programmation récursive.

3.2.2 Optimisation débit-distorsion

3.2.2.1 Le problème d'allocation

Le problème de l'allocation de débit, i.e. la distribution du budget de bits entre chaque sous-bande est un problème classique en compression de données. Shoham et Gersho ont traité le problème dans le cadre de la théorie débit-distorsion [Sho88]. Leur solution consiste à minimiser la distorsion sous une contrainte de débit en utilisant la méthode du lagrangien.

Dans le contexte de la décomposition en ondelettes, différents types de quantificateurs peuvent être utilisés pour les différentes sous-bandes. Chaque quantificateur donnera une valeur de débit et une valeur de distorsion pour chaque sous-bande (Fig. 3.7). On appelle S l'ensemble fini des combinaisons de quantificateurs pour les sous-bandes et B un élément de S . Un choix

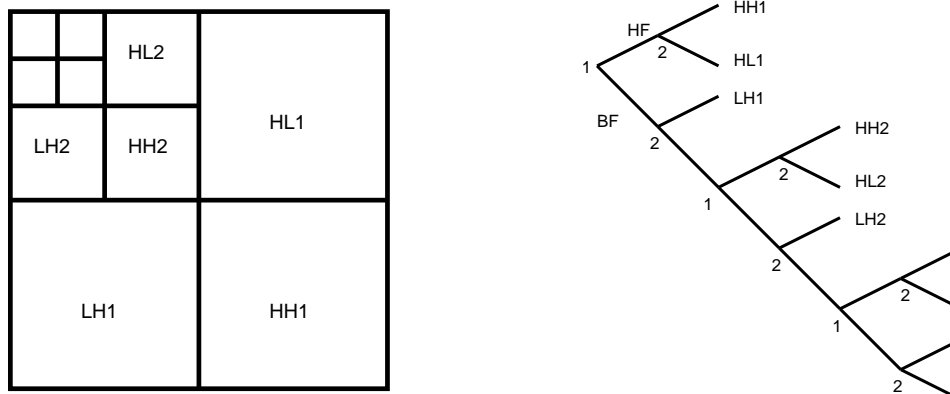


Fig. 3.5 Exemple de décomposition en ondelettes d'une image 2D classique (décomposition de Mallat). La décomposition est itérée sur la bande basse fréquence (en haut à gauche). L'arbre associé correspondant à cette décomposition est représenté à droite.

de B indiquera le quantificateur utilisé pour chacune des sous-bandes de la décomposition. Le problème est de minimiser la distorsion $D(B)$ sous la contrainte de débit total $R(B)$ dans le budget total R_c :

$$\min_{B \in \mathcal{S}} \{D(B)\} \text{ sous } R(B) \leq R_c. \quad (3.4)$$

En utilisant la méthode du lagrangien, on transforme cette minimisation sous contrainte en minimisation d'une fonction de coût J sans contrainte mais avec un paramètre λ_J

$$J(\lambda_J) = D + \lambda_J R. \quad (3.5)$$

Sous une hypothèse de codage indépendant des différentes sous-bandes et d'additivité pour les mesures de distorsion et de débit, on montre que l'optimal global est atteint lorsque toutes les sous-bandes sont à un même point de fonctionnement λ_J pour leur courbe débit-distorsion. Le problème devient alors :

$$\min \{D_k + \lambda_J R_k\} \text{ pour chaque sous-bande } k. \quad (3.6)$$

La preuve de l'équivalence entre le problème sous contrainte et le problème sans contrainte est simple et peut être trouvée dans [Sho88].

3.2.2.2 Algorithme

Le débit R_k^q pour chaque sous-bande est évalué en utilisant le codeur arithmétique défini dans [Mof98]. Le choix du codeur n'est pas critique ici. Ce sont les positions relatives des différents choix possibles qui sont importants, pas leur performances absolues. Des simulations avec d'autres mesures

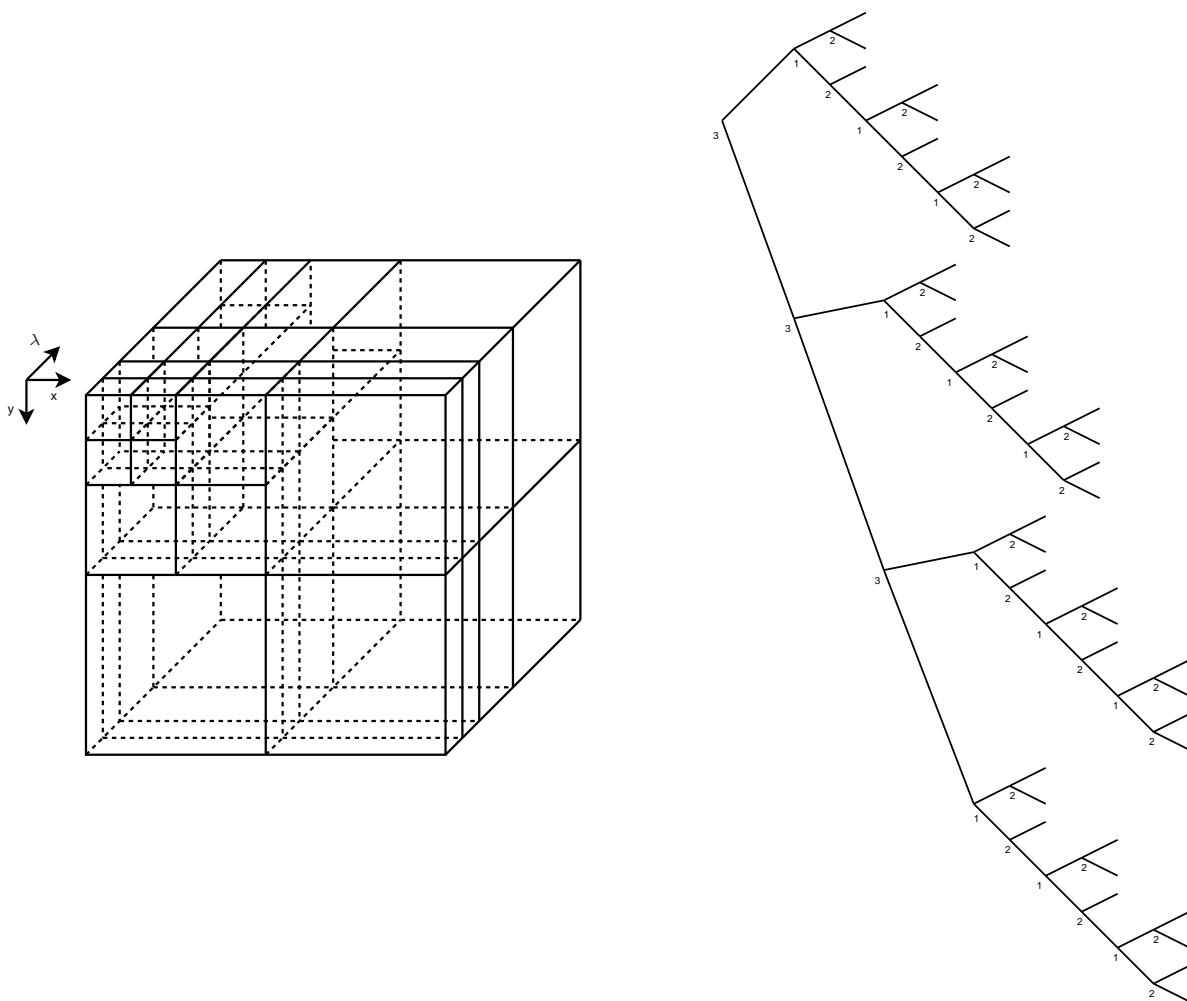


Fig. 3.6 Exemple de décomposition non isotropique d'une image hyperspectrale et son arbre de décomposition associé.

de débits comme l'entropie des coefficients des sous-bandes ou par une combinaison de *run length coding* et de codage de Rice conduisent à des résultats similaires. Le fait que les sous-bandes sont codées de manière indépendante est une supposition implicite ici.

La courbe débit-distorsion est calculée pour les 3 décompositions suivantes possibles (correspondant aux 3 directions). Une représentation illustrée sur la figure 3.8 est obtenue. Pour chaque valeur de λ_J , la fonction de coût J est calculée pour chacun des points de la courbe débit-distorsion. La décision de poursuivre ou non la décomposition est prise selon le coût le plus faible.

La recherche de type *bottom-up* est basée sur une fonction récursive et sur la propriété d'additivité de la fonction de coût J . On note $\mathcal{O}_{i,j,k}^{p,q,r}$ la base

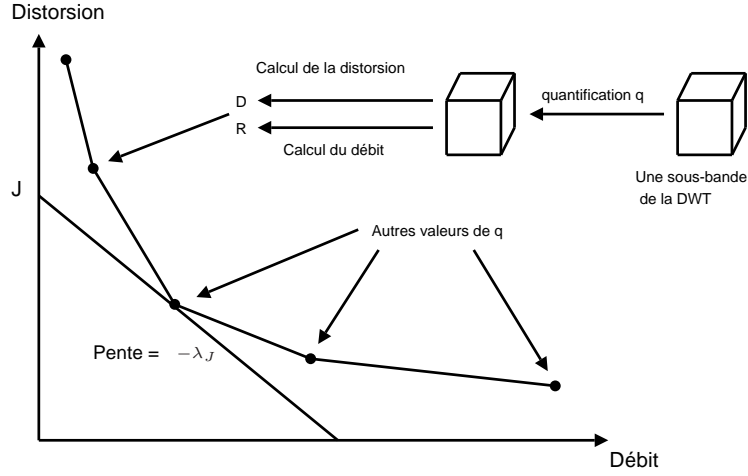


Fig. 3.7 Calcul de la courbe débit-distorsion (R-D) pour une sous-bande. Pour différents pas de quantification, la distorsion et le débit sont mesurés conduisant à différents points de fonctionnement pour former la courbe débit-distorsion.

optimale de $W_{i,j,k}^{p,q,r}$. Soit $\mathcal{B}_{i,j,k}^{p,q,r}$ la base de $W_{i,j,k}^{p,q,r}$ sans aucune transformation. On a alors une fonction de coût pour une base de représentation, $J(\lambda_J, \mathcal{B}) = \min\{D + \lambda_J R\} : R$ et D étant les points de fonctionnement de la sous-bande représentée sur la base \mathcal{B} .

J est une fonction de coût additive : pour deux bases orthogonales \mathcal{B}^1 et \mathcal{B}^2 , on a

$$J(\lambda_J, \mathcal{B}^1 \cup \mathcal{B}^2) = J(\lambda_J, \mathcal{B}^1) + J(\lambda_J, \mathcal{B}^2) \quad (3.7)$$

en utilisant les propriétés d'additivité des débits et des distorsions.

Proposition 1 (Cas 1D : Coifman, Wickerhauser). *Si J est une fonction additive de coût alors*

$$\mathcal{O}_i^p = \begin{cases} \mathcal{O}_{i+1}^{2p} \cup \mathcal{O}_{i+1}^{2p+1} & \text{si } J(\lambda_J, \mathcal{O}_{i+1}^{2p}) + J(\lambda_J, \mathcal{O}_{i+1}^{2p+1}) < J(\lambda_J, \mathcal{B}_i^p) \\ \mathcal{B}_i^p & \text{si } J(\lambda_J, \mathcal{O}_{i+1}^{2p}) + J(\lambda_J, \mathcal{O}_{i+1}^{2p+1}) \geq J(\lambda_J, \mathcal{B}_i^p) \end{cases} \quad (3.8)$$

Preuve: La meilleure base \mathcal{O}_i^p est soit égale à \mathcal{B}_i^p ou à l'union $\mathcal{B}^0 \cup \mathcal{B}^1$ des bases de W_{i+1}^{2p} et W_{i+1}^{2p+1} respectivement. Dans ce dernier cas, la propriété d'additivité (Eq. 3.7) implique que le coût sur \mathcal{O}_i^p est minimum si \mathcal{B}^0 et \mathcal{B}^1 minimisent le coût sur W_{i+1}^{2p} et W_{i+1}^{2p+1} . On a donc $\mathcal{B}^0 = \mathcal{O}_{i+1}^{2p}$ et $\mathcal{B}^1 = \mathcal{O}_{i+1}^{2p+1}$. Cela montre que \mathcal{O}_i^p est soit \mathcal{B}_i^p , soit $\mathcal{O}_{i+1}^{2p} \cup \mathcal{O}_{i+1}^{2p+1}$. La meilleure base est choisie en comparant le coût des deux possibilités. ■

Ce théorème, formulé à une dimension, peut être étendu au cas à 3 dimensions. A chaque étape on a alors 4 possibilités pour la meilleure base :

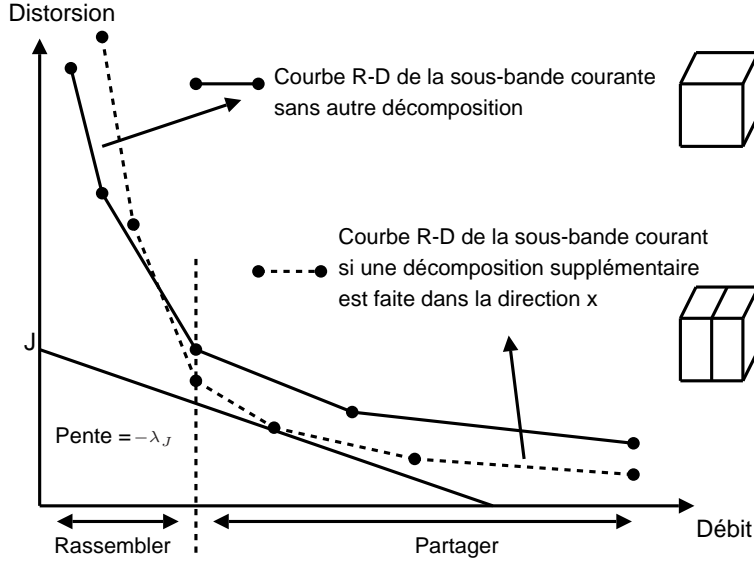


Fig. 3.8 Illustration de la décision de poursuivre ou non la décomposition. Pour plus de clarté, uniquement deux courbes débit-distorsion (sur les 4) sont représentées. Avec le λ_J présenté ici, la décision de poursuivre la décomposition selon la direction x sera prise.

Proposition 2 (Extension à 3 dimensions). *Si J est une fonction additive de coût alors*

$$\mathcal{O}_{i,j,k}^{p,q,r} = \begin{cases} \mathcal{O}_{i+1,j,k}^{2p,q,r} \cup \mathcal{O}_{i+1,j,k}^{2p+1,q,r} & \text{si } J_1 = \min\{J_0, J_1, J_2, J_3\} \\ \mathcal{O}_{i,j+1,k}^{p,2q,r} \cup \mathcal{O}_{i,j+1,k}^{p,2q+1,r} & \text{si } J_2 = \min\{J_0, J_1, J_2, J_3\} \\ \mathcal{O}_{i,j,k+1}^{p,q,2r} \cup \mathcal{O}_{i,j,k+1}^{p,q,2r+1} & \text{si } J_3 = \min\{J_0, J_1, J_2, J_3\} \\ \mathcal{B}_{i,j,k}^{p,q,r} & \text{sinon} \end{cases} \quad (3.9)$$

En notant

$$\begin{aligned} J_1 &= J(\lambda_J, \mathcal{O}_{i+1,j,k}^{2p,q,r}) + J(\lambda_J, \mathcal{O}_{i+1,j,k}^{2p+1,q,r}) \\ J_2 &= J(\lambda_J, \mathcal{O}_{i,j+1,k}^{p,2q,r}) + J(\lambda_J, \mathcal{O}_{i,j+1,k}^{p,2q+1,r}) \\ J_3 &= J(\lambda_J, \mathcal{O}_{i,j,k+1}^{p,q,2r}) + J(\lambda_J, \mathcal{O}_{i,j,k+1}^{p,q,2r+1}) \\ J_0 &= J(\lambda_J, \mathcal{B}_{i,j,k}^{p,q,r}) \end{aligned}$$

En pratique, l'algorithme est développé de manière récursive. On note J_0 le coût de la sous-bande courante sans réaliser de décomposition supplémentaire. Les coûts J_1 , J_2 et J_3 sont les coûts correspondant à une poursuite de la décomposition de la sous-bande courante respectivement selon les directions x , y ou λ . Les coûts J_1 , J_2 et J_3 sont calculés par un appel récursif à la fonction de calcul.

Algorithme 1. Recherche de la décomposition optimale

Fonction récursive : $cost(W_{i,j,k}^{p,q,r}, \lambda_J)$

- calcul du coût $J_0 = J(\lambda_J, \mathcal{B}_{i,j,k}^{p,q,r})$ par l'algorithme de Shoham et Gersho
- calcul du coût J_1
 - si la taille minimum n'est pas atteinte selon x : $J_1 = cost(W_{i+1,j,k}^{2p,q,r}, \lambda_J) + cost(W_{i+1,j,k}^{2p+1,q,r}, \lambda_J)$ par appels récursifs.
 - sinon $J_1 = \infty$
- calcul du coût J_2 : similaire à J_1
- calcul du coût J_3 : similaire à J_1
- retourne la valeur $\min\{J_0, J_1, J_2, J_3\}$

Fonction globale

- Pour chaque λ_J : appel de $cost(W_{0,0,0}^{0,0,0}, \lambda_J)$
- Courbe R-D complète pour l'image

Cet algorithme conduit à **une décomposition optimale différente pour chaque image et pour chaque débit visé**. Il est à noter que grâce à l'utilisation de la récursivité dans l'algorithme, les points de fonctionnement seront d'abord calculés pour les sous-bandes les plus petites et ensuite l'algorithme rassemblera ces valeurs pour prendre la décision de partager ou non la sous-bande. Cette recherche est exhaustive et ne conduit en aucun cas à un minimum local.

Cette recherche est similaire dans l'idée à ce qui est fait par Ramchandran dans [Ram93] avec une extension à un espace 3D anisotropique.

3.2.3 Résultats sur les images 2D

Pour illustration, la recherche de la décomposition optimale est d'abord appliquée à des images 2D classiques. Le débit est ici estimé avec un codeur arithmétique. Pour la plupart des images (comme Lena, Fig. 3.9), le gain apporté par la décomposition optimale ne justifie pas l'augmentation de complexité. Ce résultat est en accord avec le fait que la décomposition multi-résolution est largement utilisée dans les standards de compression tel que JPEG 2000. Cependant, lorsque l'image a un contenu fréquentiel plus marqué (comme Barbara, Fig. 3.10), la décomposition optimale arrive à grouper les coefficients de forte amplitude dans une sous-bande (Fig. 3.10 sous-bande 1) tandis qu'une sous-bande de taille importante (Fig. 3.10 sous-bande 2) arrive à regrouper un grand nombre de coefficients de faible amplitude. Dans ce cas, le gain peut atteindre 1.5 dB par rapport à la décomposition multi-résolution (pour des débits compris entre 0.2 et 2.0 bpp).

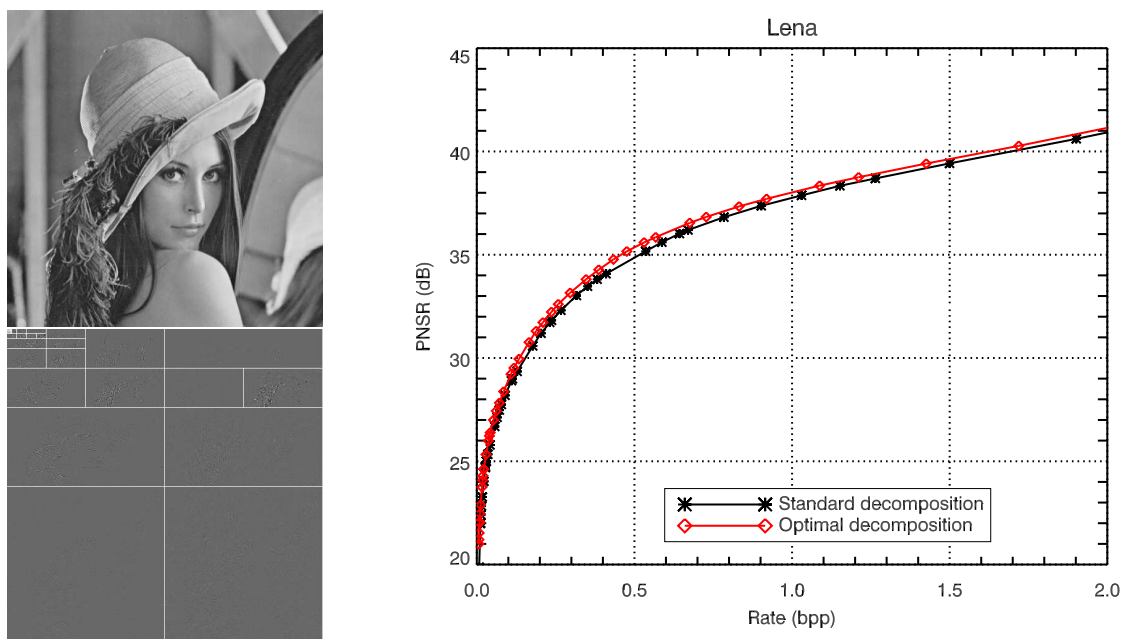


Fig. 3.9 Image Lena et la décomposition optimale obtenue pour un débit de 0.5 bit par pixel (bpp) (à gauche). Courbe débit-distorsion pour la décomposition classique et pour la décomposition optimale (droite). Dans le cas d'une image classique, la décomposition optimale n'apporte pas de gain significatif.

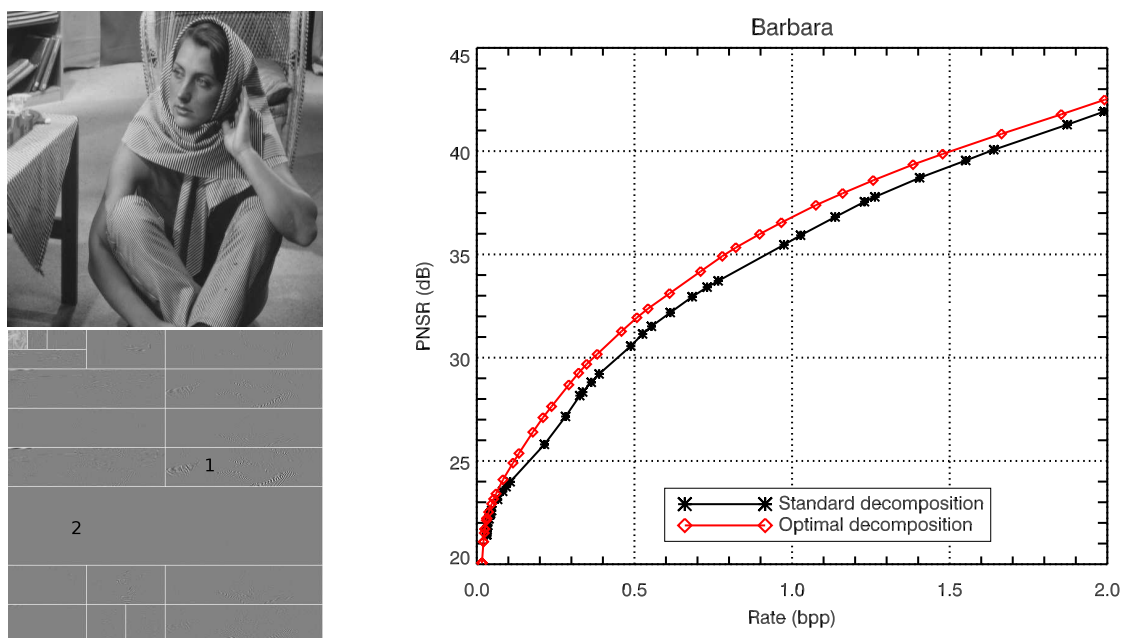


Fig. 3.10 Image Barbara et la décomposition optimale obtenue pour un débit de 0.9 bit par pixel (bpp) (à gauche). Courbe débit-distorsion pour la décomposition classique et pour la décomposition optimale (droite). Dans le cas d'une image avec des fréquences spatiales marquées, la décomposition optimale apporte un gain significatif.

3.2.4 Décomposition en ondelettes optimale pour l'hyperspectral

Les résultats présentés sur la figure 3.11 montrent que la décomposition anisotropique optimale amène une amélioration claire en terme de débit-distorsion. Ces résultats sont confirmés sur diverses images hyperspectrales. L'amélioration est d'environ 8 dB en terme de PSNR par rapport à une décomposition isotropique classique, pour des débits compris entre 0.1 et 4.0 bits par pixel par bande (bpppb). Si la comparaison est faite en terme de contrainte de qualité, par exemple pour un PSNR supérieur à 70 dB, le débit nécessaire passe de 1 bpppb à 0.5 bpppb, soit une réduction d'un facteur 2. L'importance de cette amélioration par rapport aux images classiques est due à la nature anisotropique des images hyperspectrales.

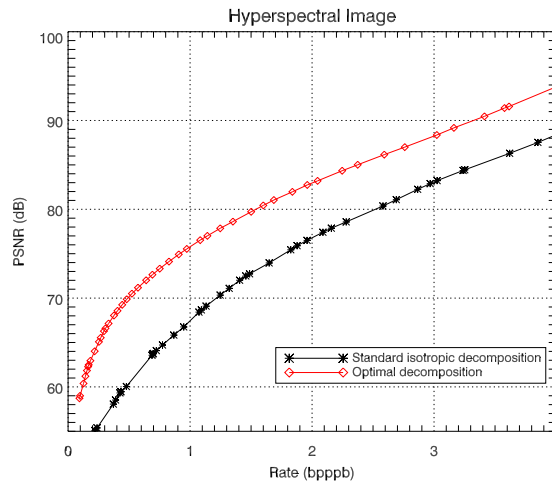


Fig. 3.11 Décomposition optimale sur des données hyperspectrales : la décomposition anisotropique optimale améliore clairement les performances (environ 8 dB), mais au prix d'une complexité importante.

3.2.5 Décomposition fixe

Il y a deux inconvénients principaux à cette décomposition optimale :

- Le coût calculatoire de la recherche de la décomposition ;
- La dépendance par rapport à l'image et au débit.

Le coût calculatoire de la méthode est important. Par exemple, la recherche de la décomposition optimale sur un cube hyperspectral de dimension $256 \times 256 \times 224$ avec une taille minimale de sous-bande de $8 \times 8 \times 7$ (un maximum de 5 niveaux de décomposition) nécessite de calculer la courbe débit-distorsion totale pour 250047 sous-bandes. Cette valeur est obtenue en calculant les combinaisons de toutes les tailles de sous-bandes possibles

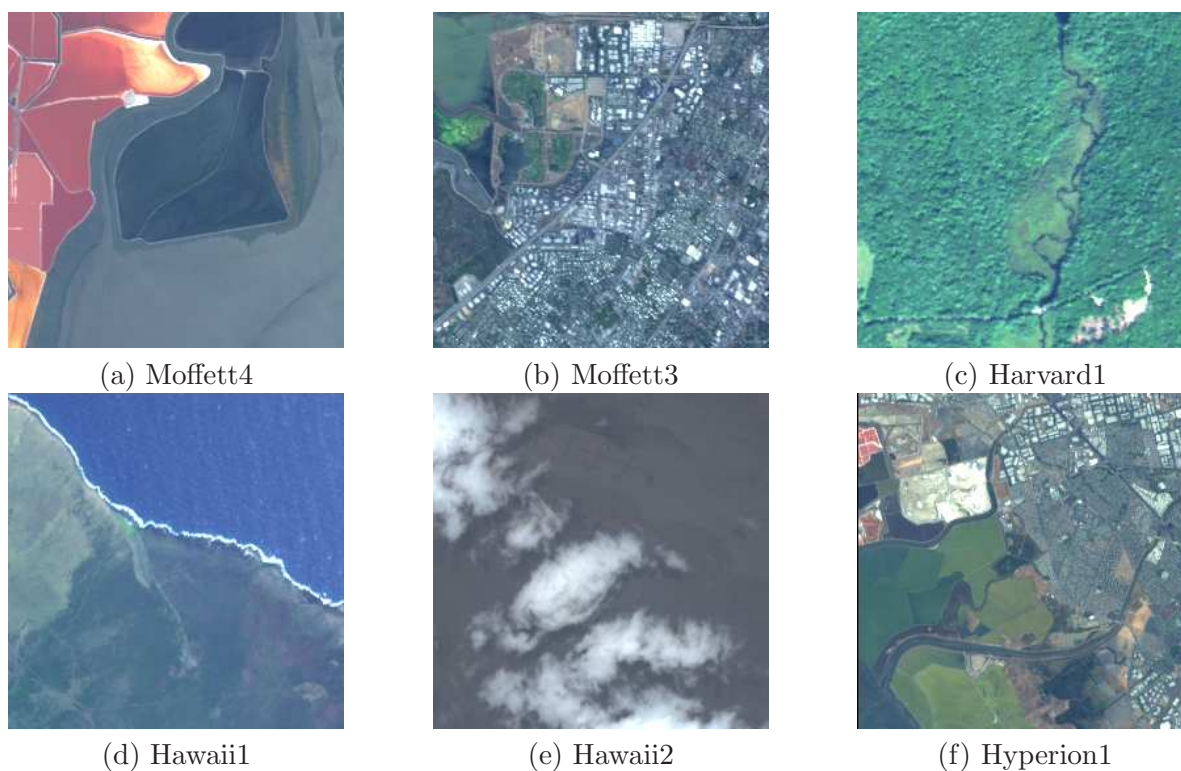


Fig. 3.12 Différentes images hyperspectrales utilisées pour les simulations. (a) et (b) sont différentes parties de la scène f970620t01p02_r03 du capteur AVIRIS sur le site de Moffett Field. (a) contient des zones uniformes avec des caractéristiques spectrales marquées. (b) est un mélange de ville (hautes fréquences spatiales). (c) provient de la scène f010903t01p01_r03 d'AVIRIS au dessus de Harvard Forest, cette image contient principalement de la végétation. (d) et (e) viennent de la scène f000414t01p03_r08 à Hawaï. (d) contient deux zones contrastées (minérale et océan) séparées par une ligne de côte. (e) a été sélectionnée pour illustrer une scène contenant des nuages, une forte dynamique et des contrastes importants. (f) provient du capteur spatial Hyperion (EO1H0440342002212110PY) et couvre une zone similaire à celle des images (a) et (b).

dans les trois directions : pour une direction, on a $2^0 + 2^1 + \dots + 2^5 = 63$ tailles possibles, en combinant dans les 3 dimensions, on a $63^3 = 250047$ sous-bandes possibles.

La dépendance de la décomposition par rapport à l'image et au débit pose des problèmes en terme d'implémentation de la transformation. En général, on préfère des transformations indépendantes des données.

Le but est donc de trouver une décomposition suffisamment proche de la décomposition optimale pour donner des performances quasi-optimales, tout en conservant une forme assez générale qui lui permette de rester valable pour une grande variété d'images. La structure de la décomposition optimale pour différentes images à différents débits, ainsi que la localisation de la corrélation résiduelle pour la décomposition isotropique conduit à essayer une décomposition particulière. Cette décomposition régulière consiste à appliquer une décomposition multirésolution standard sur les spectres suivie d'une décomposition multirésolution 2D sur les images résultantes (l'ordre des opérations étant réversible). Cette décomposition a montré de bonnes performances dans plusieurs travaux [Kim00, Wan04a], mais uniquement avec une justification empirique. Une justification théorique en terme d'entropie apparaît dans [Pen06].

La décomposition obtenue est présentée sur la figure 3.13. Cette décomposition est comparée avec la décomposition isotropique classique, les coefficients en gris représentent les valeurs proches de 0, les coefficients en blanc sont les coefficients positifs tandis qu'en noir sont les coefficients négatifs. Comme on peut le voir sur la figure 3.14, cette décomposition fixe est presque aussi performante que la décomposition optimale. La régularité de cette structure ainsi que les faibles perspectives de gain possibles à chercher une autre décomposition conduisent à choisir cette décomposition.

3.3 Structures d'arbres

3.3.1 Idées principales

Une des faiblesses possibles de JPEG 2000 est qu'il n'utilise pas les relations existantes entre les localisations des coefficients significatifs entre les différentes sous-bandes. D'après Taubman [Tau02], le bénéfice venant du choix du point de troncature compense le fait que les relations entre les coefficients ne sont pas utilisées. Cette conclusion peut être différente dans le cas des images hyperspectrales, la corrélation entre les sous-bandes étant anormalement élevée.

Les arbres de zéros sur les coefficients d'ondelettes ont été développés pour utiliser la relation qui existe entre la position des singularités dans les différentes sous-bandes. Après la transformée en ondelettes, on observe que la localisation des coefficients significatifs est similaire entre les différentes

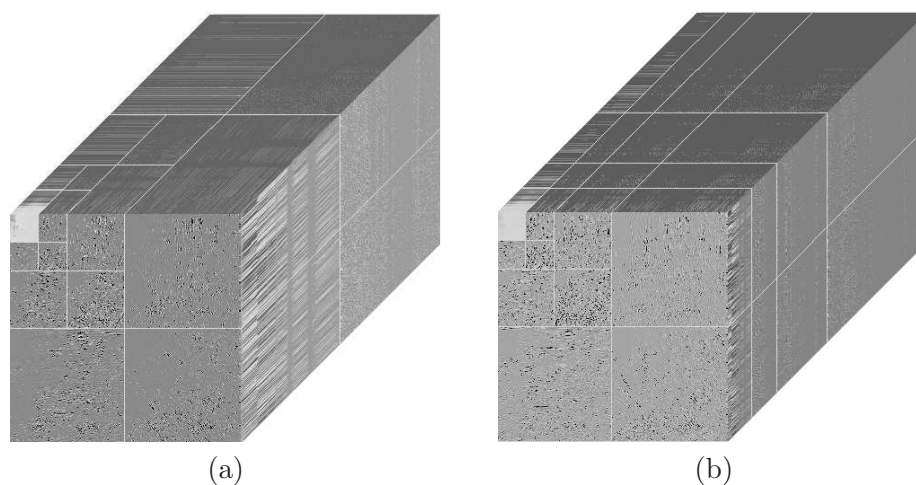
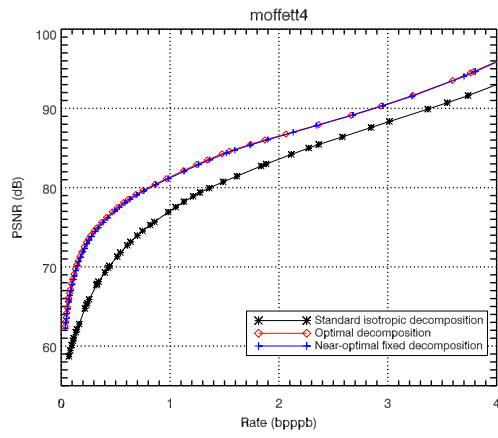


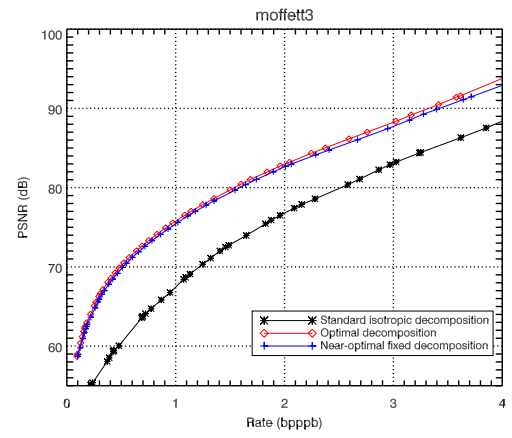
Fig. 3.13 Décomposition isotropique classique (a) et décomposition anisotrope (b) avec 3 niveaux de décomposition (les simulations sont faites avec 5 niveaux). Pour la décomposition isotropique, on peut remarquer qu'il reste une corrélation importante entre les coefficients dans les basses fréquences spectrales et spatiales (lignes de coefficients consécutifs de valeur similaire). Cette corrélation résiduelle est plus faible pour la décomposition anisotrope.

sous-bandes même si leurs amplitudes sont décorréliées (Fig. 3.15). La propriété à l'origine des arbres de zéros est que **si un coefficient n'est pas significatif dans une sous-bande alors le coefficient à la même position dans une sous-bande de plus haute fréquence sera lui aussi probablement non significatif**.

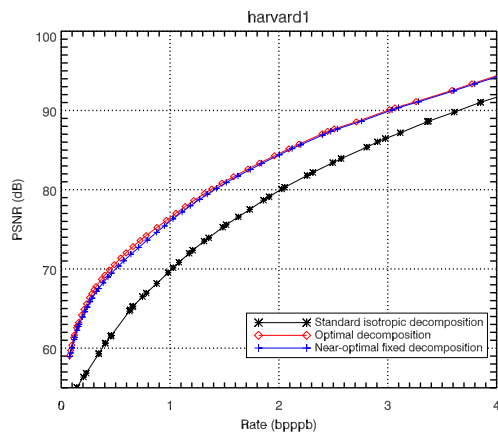
Cette idée a été exploitée avec succès par Shapiro avec EZW (Embedded Zerotree of Wavelet coefficients) [Sha93]. Une amélioration remarquable a été apportée quelques années plus tard par Said et Pearlman avec SPIHT (Set Partitioning In Hierarchical Trees) [Sai96]. Toutefois, les performances des arbres de zéros sont supérieures à ce qu'on pouvait en attendre [Mar01] et ce succès n'est pas complètement expliqué. L'idée de SPIHT a ensuite été reprise pour donner des solutions progressives en résolution [Dan03], résistantes aux erreurs de transmission [Cho05a] ou généralisées pour des structures 3D, principalement pour la compression vidéo [Kim00, He03] ou pour les images médicales [Bil00, Xio03]. Sur le modèle de SPIHT, d'autres codeurs ont ensuite vu le jour comme SPECK [Isl99, Pea04] qui tire parti des similarités entre les coefficients adjacents dans une même sous-bande (blocs de zéros plutôt que des arbres de zéros). SPECK a ensuite été adapté en SBHP [Chr00] pour concurrencer EBCOT [Tau00] dans la norme JPEG 2000 ainsi qu'en EZBC [Hsi01] utilisant un codage arithmétique contextuel évolué pour augmenter les performances. LTW [Oli03], amélioré dans [Guo06] propose une approche différente pour réduire la complexité. Plus récemment, SPECK a été adapté pour fournir un codage progressif en résolution [Xie05].



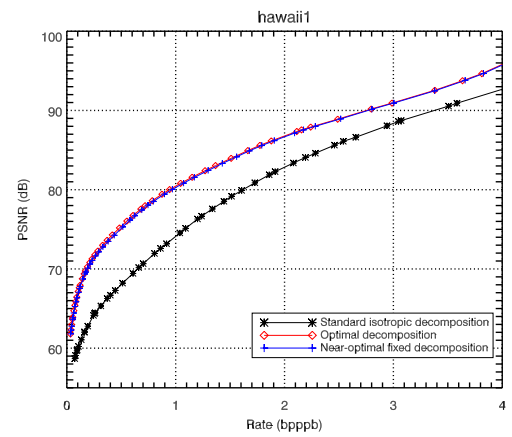
(a) moffett4



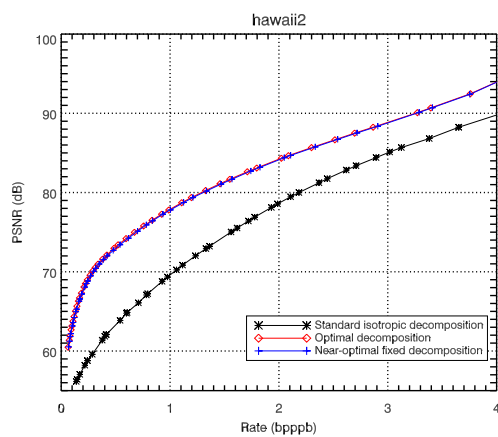
(b) moffett3



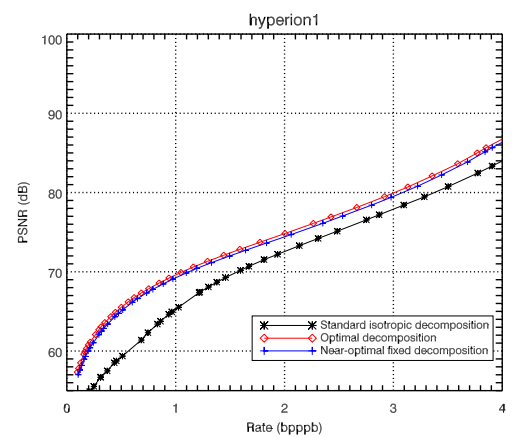
(c) Harvard1



(d) Hawaii1



(e) Hawaii2



(f) hyperion1

Fig. 3.14 Résultats sur des données hyperspectrales : la décomposition anisotropique optimale améliore clairement les performances. La décomposition anisotropique fixe permet d'obtenir des performances proches de la décomposition optimale tout en gardant une complexité raisonnable.

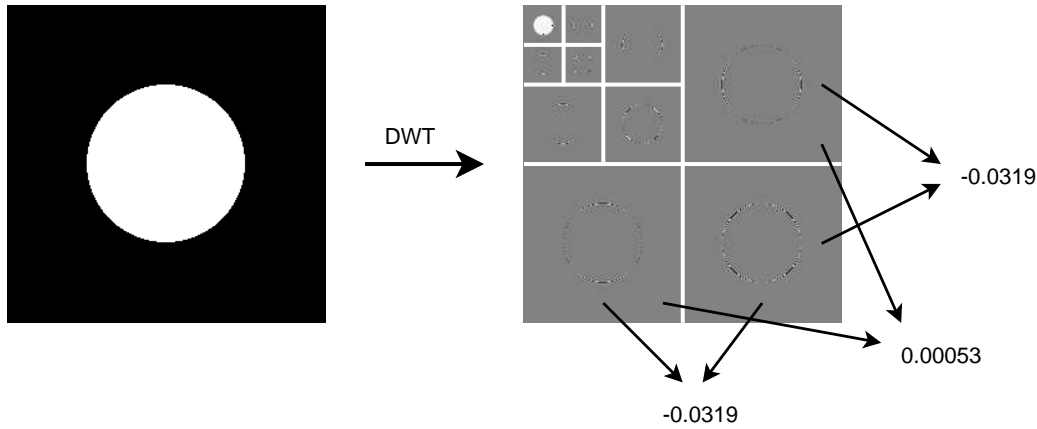


Fig. 3.15 Illustration du principe des arbres de zéros : même si les sous-bandes sont décorréélées (les coefficients de corrélation entre les bandes HL, LH et HH sont très faibles), les endroits où les singularités apparaissent sont les mêmes dans les différentes sous-bandes.

3.3.2 Principes généraux de EZW et SPIHT

Au moment de sa publication, l'algorithme EZW de Shapiro, utilisant les arbres de zéros, produisait les meilleures performances en terme de débit-distorsion tout en ayant une complexité faible. Cette structure d'arbres de zéros a été généralisée quelques années après par Said et Pearlman pour donner l'algorithme SPIHT.

Ces deux algorithmes possèdent des propriétés qui les rendent particulièrement attractifs dans le domaine de la compression spatiale embarquée. Ils produisent tous deux un **train binaire emboîté** : le préfixe d'un train binaire produit par EZW (ou SPIHT) est lui-même un train binaire EZW (ou SPIHT) valide conduisant à une image décompressée avec une qualité plus faible. Ces deux algorithmes atteignent ce résultat avec un niveau modeste de complexité.

La question de la complexité algorithmique est un point critique pour la définition des algorithmes embarqués, la puissance de calcul étant très limitée. Cela est particulièrement vrai dans le cas des satellites. Le fait d'avoir un train binaire emboîté est intéressant pour éviter les dépassements des mémoires et garantir la meilleure qualité d'image étant données les ressources disponibles à bord (mémoire et calcul).

Pour assurer cette propriété de train binaire emboîté, la compression procède par plans de bits. On note $c_{x,y,\lambda}$ le coefficient de la transformée en ondelettes de la colonne x , de la ligne y et du plan spectral λ . On définit la suite de seuils T_{K-1}, \dots, T_0 , tels que $T_k = T_{k+1}/2$. Le seuil initial est choisi pour que $|c_{x,y,\lambda}| < 2T_{K-1}$ pour tous les coefficients d'ondelettes. Pour des raisons pratiques de représentation binaire, on choisit T_{K-1} comme une puissance de 2. Le coefficient $c_{x,y,\lambda}$ est significatif au plan de bits k si

Plan de bits	T_k	7	30	180
7	128	0	0	1
6	64	0	0	0
5	32	0	0	1
4	16	0	1	1
3	8	0	1	0
2	4	1	1	1
1	2	1	1	0
0	1	1	0	0

Tab. 3.2 Exemple de codage en plan de bits.

$|c_{x,y,\lambda}| \geq T_k$. Les plans de bits sont codés les uns après les autres permettant de réduire la distorsion à chaque étape.

Un exemple de codage par plan de bits est présenté dans le tableau 3.2. On considère des coefficients d'ondelettes de valeurs 7, 30 et 180. 7 n'est pas significatif avant le plan de bits 2, 30 avant le 4 et 180 le devient dès le plan de bits 7. Après qu'un coefficient ait été marqué comme significatif, les bits suivants doivent être codés. Le signe des coefficients est codé séparément.

Pour *chaque plan de bits*, le codage suit une structure d'arbres de zéros, les arbres étant définis le long des différentes sous-bandes de la transformée en ondelettes.

La définition d'un arbre de zéros varie en fonction des algorithmes. Dans [Cho05b] est développée l'idée des arbres de zéros de degré k . Un arbre de degré 0 est un arbre dont tous les coefficients sont égaux à zéros (Fig. 3.16), un arbre de degré 1 est un arbre dont tous les coefficients sauf la racine sont égaux à zéros (Fig. 3.17) et un arbre de degré 2 est un arbre dont tous les coefficients sauf la racine et ses enfants directs sont égaux à zéros (Fig. 3.18). EZW utilise des arbres de degré 0 tandis que SPIHT utilise des arbres de degré 1 et 2.

Un exemple de la première passe des algorithmes EZW et SPIHT est détaillé dans l'annexe B.

3.3.3 Étude statistique pour le choix de la structure d'arbre

Étant donné la décomposition optimale précédente, plusieurs structures d'arbre peuvent être définies. On peut utiliser le lien entre les sous-bandes spatiales, entre les sous-bandes spectrales ou les deux. L'avantage principal des arbres de zéros est leur capacité de coder un grand nombre de coefficients à zéro (pour le plan de bits courant) en n'utilisant que très peu de symboles. **La structure d'arbre optimale est donc celle qui maximise la longueur des arbres de zéros tout en ne laissant qu'un nombre très faible de zéros isolés.** Quelle que soit la structure choisie, les coefficients significatifs seront les mêmes. La différence dans l'efficacité des structures

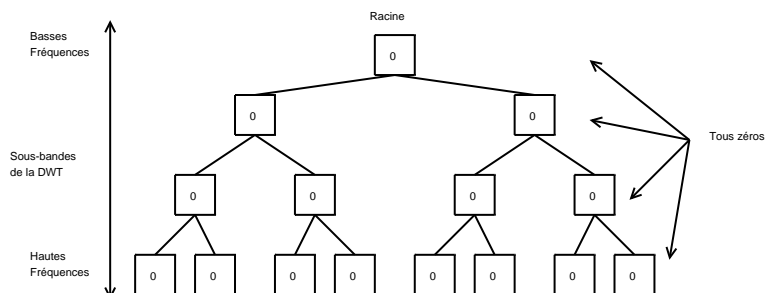


Fig. 3.16 Arbre de zéros de degré 0

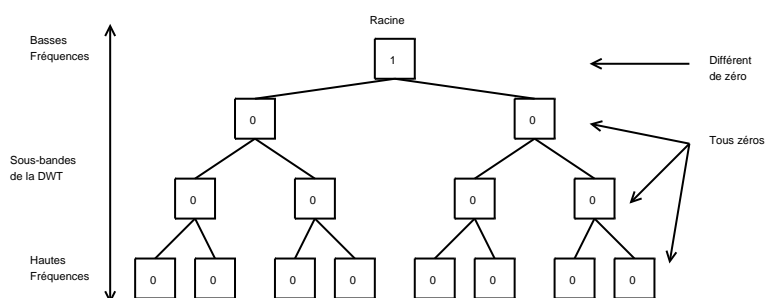


Fig. 3.17 Arbre de zéros de degré 1

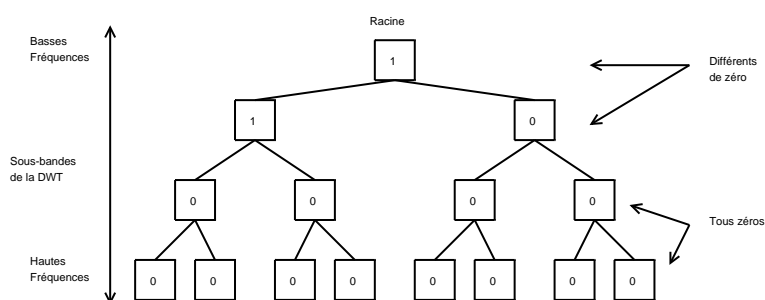


Fig. 3.18 Arbre de zéros de degré 2

d'arbres est uniquement due à leur capacité à rassembler les coefficients nuls.

Il y a trois structures régulières d'arbre possibles :

- Utiliser uniquement la relation spatiale entre les coefficients : arbres spatiaux ;
- Utiliser uniquement la relation spectrale entre les coefficients : arbres spectraux ;
- Utiliser les deux relations : arbres 3D.

Le choix optimal n'est pas évident : mettre plus de coefficients dans un arbre permet d'encoder potentiellement plus de zéros avec un seul symbole, mais augmente également le risque d'avoir un coefficient significatif détruisant l'arbre. Des choix ont été faits précédemment mais sans justifications comme dans [He03].

Des statistiques sont calculées pour la transformée d'une image hyperspectrale avec 5 niveaux de décomposition et sont présentées dans les tableaux 3.3, 3.4 et 3.5. Ces tableaux donnent le nombre de coefficients significatifs pour chaque plan de bits (qui doivent être codés de toute façon et sont indépendants de la structure choisie), le nombre de zéros isolés (IZ) ainsi que le nombre d'arbres de zéros (ZTR). On considère ici les arbres de degré 0. Pour un plan de bits donné k , tous les coefficients en dessous du seuil T_k sont à zéro. Le but du codage par arbre de zéros est d'inclure le maximum de coefficients dans un arbre de 0 en utilisant un seul symbole (ZTR dans la terminologie EZW comme on le verra plus tard). Les coefficients en dessous du seuil ne pouvant pas être inclus dans un arbre de zéros sont codés avec un symbole (IZ). Les deux symboles, ZTR et IZ, sont utilisés pour coder les coefficients à zéros. Le but de la structure d'arbre est de minimiser le nombre de symboles utilisés ou de maximiser le nombre de 0 codés avec un seul symbole. Pour comparer l'efficacité des différentes structures, le nombre moyen de 0 codés par un symbole (ZTR ou IZ) est donné ci-dessous dans les trois tableaux 3.3, 3.4, 3.5.

Dans ces tableaux le terme *moyenne* désigne le nombre de 0 codés en moyenne par un des symboles IZ ou ZTR. Cette valeur est obtenue par :

$$\text{moyenne} = \frac{\text{nbr coeff} - \text{nbr coeff significatifs}}{\text{nbr IZ} + \text{nbr ZTR}} \quad (3.10)$$

Par exemple, pour une structure d'arbre 3D (tab. 3.3), pour le plan de bits 16, 610 symboles sont significatifs, les autres $((256*256*224)-610=14679454)$ sont codés par $94+1960 = 2054$ symboles, ce qui donne une moyenne de 7146.76 zéros codés par symbole. Cette valeur est à comparer avec l'arbre spatial pour lequel 15745 symboles sont nécessaires (932.32 zéros codés par symbole).

Ces tableaux montrent que le choix de la structure 3D est plus efficace et utilise moins de symboles pour coder les coefficients à 0 quelque soit le plan de bits. Cette structure correspond à la structure illustrée sur la figure 3.20.

Plan de bits	Significatif	IZ	ZTR	Moyenne
19	99	0	745	19704.65
18	183	0	997	14724.05
17	249	2	1195	12263.84
16	610	94	1960	7146.76
15	1143	264	3595	3803.81
14	2975	975	9619	1385.41
13	9052	2678	28191	475.27
12	25525	6797	66564	199.76
11	64646	14587	122718	106.44
10	141869	29824	189039	66.43
9	276589	56712	268876	44.24
8	499196	104615	372197	29.74
7	860482	189221	517928	19.54
6	1461127	376246	718656	12.07
5	2439789	691967	926441	7.56
4	3940555	1299399	1015978	4.64
3	6053237	2130317	1098716	2.67
2	8950166	2860847	800219	1.57
1	11795923	2079228	341052	1.19
0	13673387	777902	156241	1.08

Tab. 3.3 Statistiques pour une structure d'arbre 3D.

Plan de bits	Significatif	IZ	ZTR	Moyenne
19	99	0	14534	1010.04
18	183	0	14702	998.50
17	249	2	14842	988.94
16	610	41	15704	932.32
15	1143	294	17256	836.41
14	2975	1877	22564	600.51
13	9052	10995	35115	318.17
12	25525	39488	53933	156.87
11	64646	95909	78453	83.82
10	141869	179019	112395	49.89
9	276589	297178	164097	31.23
8	499196	480696	245862	19.52
7	860482	787705	358326	12.06
6	1461127	1323093	473480	7.36
5	2439789	2019657	543211	4.78
4	3940555	2783606	543079	3.23
3	6053237	3597503	583521	2.06
2	8950166	4056970	316510	1.31
1	11795923	2557907	111127	1.08
0	13673387	918152	56742	1.03

Tab. 3.4 Statistiques pour une structure d'arbre spatial.

Plan de bits	Significatif	IZ	ZTR	Moyenne
19	99	0	458752	32.00
18	183	0	458752	32.00
17	249	0	458755	32.00
16	610	57	459065	31.97
15	1143	124	459261	31.95
14	2975	566	459575	31.90
13	9052	1251	460365	31.78
12	25525	2659	462871	31.48
11	64646	4930	472529	30.61
10	141869	12290	501134	28.31
9	276589	32602	568224	23.97
8	499196	84838	696563	18.15
7	860482	210635	895402	12.49
6	1461127	511498	1145031	7.98
5	2439789	1016474	1336894	5.20
4	3940555	1751396	1346870	3.47
3	6053237	2510335	1216711	2.31
2	8950166	3104591	763072	1.48
1	11795923	2154989	323541	1.16
0	13673387	804094	142971	1.06

Tab. 3.5 Statistiques pour une structure d'arbre spectral.

Cette structure est différente de celle utilisée dans [Tan03] (présentée sur la figure 3.19)

3.3.4 Adaptation de EZW

La partie délicate dans le codage par plan de bits est le codage de la carte des coefficients significatifs, i.e. l'ensemble des décisions binaires pour savoir si un coefficient est significatif pour un seuil donné T_k . Les algorithmes EZW et SPIHT fournissent des moyens efficaces pour coder cet ensemble. Pour la plupart des coefficients (à l'exception des sous-bandes de plus basse et de plus haute fréquences), avec la structure d'arbre optimale précédente, un coefficient $c_{x,y,\lambda}$ a deux descendants spectraux $c_{x,y,2\lambda}$ et $c_{x,y,2\lambda+1}$ et quatre descendants spatiaux $c_{2x,2y,\lambda}$, $c_{2x+1,2y,\lambda}$, $c_{2x,2y+1,\lambda}$ et $c_{2x+1,2y+1,\lambda}$.

Chaque plan de bits est codé en deux passes. La première passe, appelée *significance pass*, code la carte des coefficients significatifs pour le plan de bits courant. La structure d'arbre de zéros permet de réduire le coût pour le codage de la carte en utilisant les relations entre les sous-bandes. La deuxième passe, *refinement pass*, code un bit pour chaque coefficient ayant été déclaré comme significatif à un autre plan de bits.

Comme il a été montré par Shapiro, il est utile de coder le signe des coefficients significatifs en même temps que la carte. En pratique 4 symboles différents sont utilisés : racine d'un arbre de zéros (Zero Tree Root ou ZTR), zéro isolé (Isolated Zero ou IZ), coefficient positif (POS) ou coefficient négatif

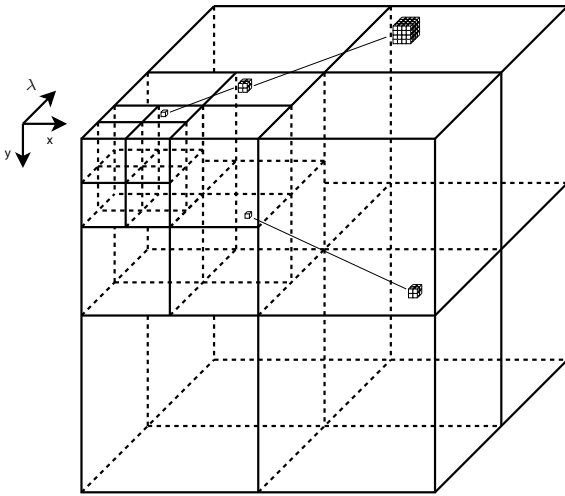


Fig. 3.19 Relation entre les coefficients dans le cas d'une décomposition isotropique classique.

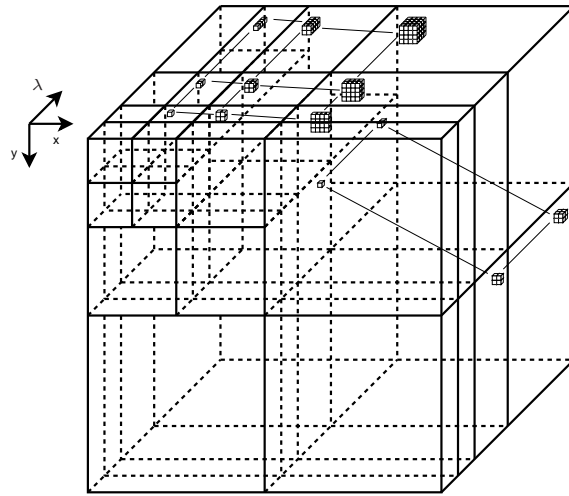


Fig. 3.20 Arbre 3D définissant la relation entre les coefficients dans le cas de la décomposition anisotrope. Il est à noter que les descendance spatiales et spectrales se croisent donnant une relation redondante. La relation parent-enfant n'est plus une relation mono-parentale.

(NEG). Chacun de ces symboles peut être codé en utilisant 2 bits. IZ signifie que le coefficient courant est en dessous du seuil mais qu'au moins un de ses descendants est au dessus (ce coefficient ne peut pas être inclus dans un arbre de zéros). Le symbole ZTR indique que le coefficient courant est en dessous du seuil et que tous ses descendants sont également en dessous du seuil (ou alors sont déjà déclarés comme significatifs et seront de toute façon traités durant la *refinement pass*). Le parcours des coefficients est effectué sous-bande par sous-bande (Fig. 3.21 pour le cas 2D) de telle manière qu'aucun coefficient ne soit vu avant un de ses parents. Au sein de chaque sous-bande, les coefficients sont parcourus en zigzag. Le codage est fait par un alphabet à quatre symboles (POS, NEG, ZTR and IZ).

Algorithme 2. EZW

Pour chaque plan de bits :

- *Significant pass* : Pour tous les coefficients non significatifs, coder un symbole ZTR, IZ, POS ou NEG selon le cas. Les coefficients sont traités dans l'ordre indiqué sur la figure 3.21.
- *Refinement pass* : écrire un bit pour tous les coefficients ayant été déclarés comme significatifs (sauf ceux du dernier plan de bits), ce bit correspond à la valeur du coefficient dans le plan de bits courant.

■

Un exemple de déroulement de l'algorithme EZW est détaillé dans l'annexe B.

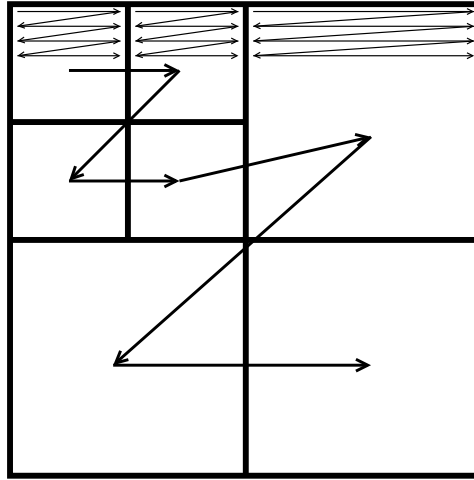


Fig. 3.21 Ordre de parcours des coefficients pour EZW.

Il est important de noter que les coefficients ayant déjà été notés comme significatifs seront traités durant la *refinement pass* et peuvent donc être inclus sans dommage dans un arbre de zéros. Une attention particulière doit être portée sur ce point lors de la programmation de l'algorithme. En effet, dans le cas d'image 3D, un coefficient peut avoir deux parents (un spatial, un spectral), conduisant à une situation de croisement d'arbres (tree crossing) illustré sur la figure 3.22. Ce phénomène est propre à la structure d'arbre redondante utilisée ici.

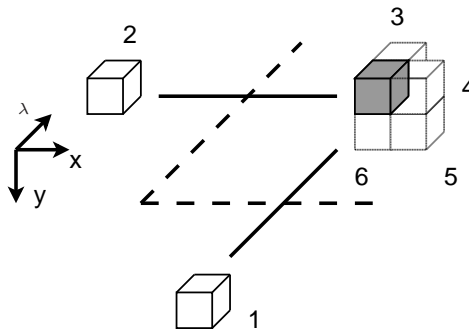


Fig. 3.22 Illustration du tree crossing : le coefficient en gris peut être considéré comme le descendant spectral du coefficient 1 avec le coefficient 3 ou comme le descendant spatial du coefficient 2 avec 4, 5 et 6.

Les performances sur l'image *moffett3* sont détaillées dans le tableau 3.6 pour la structure d'arbre 3D ainsi que pour la structure plus classique d'arbre spatial. Un codeur arithmétique peut être appliqué sur le train binaire en sortie du codeur. Le codeur arithmétique utilisé est

décrit dans [Mof98] et disponible dans un but de recherche à l'adresse http://www.cs.mu.oz.au/~alastair/arith_coder/. Le codeur arithmétique (pour les résultats noté AC) est appliqué directement sur le train EZW et ne prend en compte aucun contexte. Comme on peut le voir dans le tableau 3.6, la structure d'arbre 3D amène une amélioration significative par rapport à l'arbre spatial.

	1.0 bpppb	0.5 bpppb
EZW-3D	73.77 dB	67.98 dB
EZW-3D-AC	75.54 dB	69.25 dB
EZW-spat	71.94 dB	66.31 dB
EZW-spat-AC	74.87 dB	68.84 dB

Tab. 3.6 Performances de EZW sur *moffett3* (PSNR). *3D* correspond à l'utilisation de la structure d'arbre 3D, *spat* à l'utilisation de la structure d'arbre spatial, *AC* signale l'utilisation d'un codeur arithmétique. La structure d'arbre 3D amène une amélioration significative.

3.3.5 Adaptation de SPIHT

L'algorithme SPIHT est présenté par Said et Pearlman dans [Sai96]. Les principales propriétés de EZW sont préservées : codage progressif et faible complexité. Cependant, quelques différences conduisent à une amélioration pour les images 2D classiques. L'algorithme SPIHT maintient trois listes de coefficients : la liste des coefficients significatifs (List of Significant Pixels ou LSP), la liste des coefficients non significatifs (List of Insignificant Pixels ou LIP) et la liste des ensembles non significatifs (List of Insignificant Sets ou LIS). $\mathcal{O}(x, y, \lambda)$ est l'ensemble des enfants de (x, y, λ) (un seul niveau de descendance), $\mathcal{D}(x, y, \lambda)$ est l'ensemble de tous les descendants et $\mathcal{L}(x, y, \lambda) = \mathcal{D}(x, y, \lambda) - \mathcal{O}(x, y, \lambda)$ est l'ensemble des descendants à l'exception des enfants (Fig. 3.23). La fonction $S_t(x, y, \lambda)$ est égale à 0 si tous les descendants de (x, y, λ) sont en dessous du seuil T_t (arbre de zéros) et 1 dans le cas contraire.

La première différence à noter par rapport à EZW est que toute sortie est binaire. La seconde différence est que l'ordre de traitement des coefficients est dépendant des données. Alors que les coefficients sont traités en zigzag dans chaque sous-bande pour EZW (Fig. 3.21), le système de liste de SPIHT laisse l'ordre entièrement dépendant des données. Les coefficients sont traités selon leur position dans les listes. Une autre différence à noter est la relation parent-enfant pour la sous-bande de plus basse fréquence (Fig. 3.24). Dans le cas 2D pour EZW, chaque coefficient pour la sous-bande basses fréquences (LL) possède 3 descendants (dans LH, HL et HH). Pour SPIHT, un descendant sur quatre n'a pas de descendant tandis que les autres en ont 4. La définition des arbres est aussi sensiblement différente car SPIHT considère deux types d'arbres de zéros : le type A où tous les descendants ne

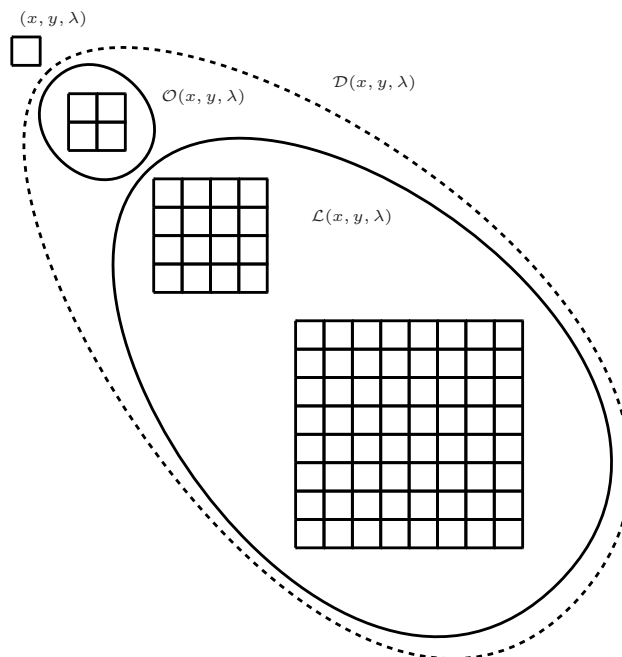


Fig. 3.23 Terminologie SPIHT pour les descendants.

sont pas significatifs (arbre de degré 1) et le type B où tous les descendants, à l'exception d'au moins un des enfants, ne sont pas significatifs (arbre de degré 2). Il est à noter que dans les deux cas, rien n'est dit sur la valeur du coefficient à la racine qui peut être significatif. Un exemple du déroulement de SPIHT dans le cas d'images 2D est donné dans l'annexe B.

Sur les images hyperspectrales avec la relation entre les coefficients définis précédemment, l'algorithme SPIHT se déroule selon :

Algorithme 3. *SPIHT Anisotropique*

Initialisation :

- Pour chaque plan de bits t :
- $LSP = \emptyset$
- LIP : tous les coefficients sans parent (coefficients de la LLL)
- LIS : tous les coefficients de la LIP qui ont des descendants (marqués comme type A, par défaut)

Sorting pass :

Pour chaque coefficient (x, y, λ) de la LIP

- Écrire $S_t(x, y, \lambda)$
- Si $S_t(x, y, \lambda) = 1$, déplacer (x, y, λ) dans la LSP et écrire le signe de $c_{x,y,\lambda}$

Pour chaque coefficient (x, y, λ) de la LIS

- Si le coefficient est de type A
 - Écrire $S_t(\mathcal{D}(x, y, \lambda))$
 - Si $S_t(\mathcal{D}(x, y, \lambda)) = 1$ alors

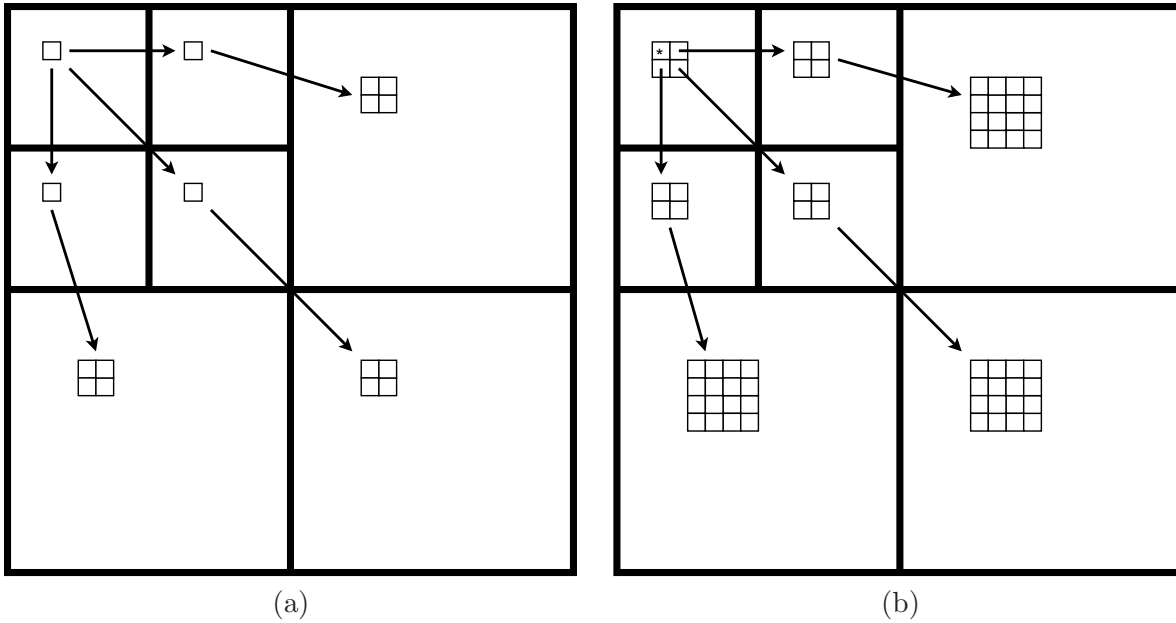


Fig. 3.24 Différences pour la descendance des coefficients basses fréquences entre EZW (a) et SPIHT (b).

- Pour tout $(x', y', \lambda') \in \mathcal{O}(x, y, \lambda)$: écrire $S_t(x', y', \lambda')$; si $S_t(x', y', \lambda') = 1$, ajouter (x', y', λ') à la LSP et écrire le signe de $c_{x', y', \lambda'}$ sinon, ajouter (x', y', λ') à la fin de la LIP. (Étape critique pour le problème du tree-crossing).
- Si $\mathcal{L}(x, y, \lambda) \neq \emptyset$, déplacer (x, y, λ) à la fin de la LIS comme une entrée de type B
- Sinon, retirer (x, y, λ) de la LIS
- Si l'entrée est de type B
 - Écrire $S_t(\mathcal{L}(x, y, \lambda))$
 - Si $S_t(\mathcal{L}(x, y, \lambda)) = 1$
 - Ajouter tous les $(x', y', \lambda') \in \mathcal{O}(x, y, \lambda)$ à la fin de la LIS comme entrée de type A
 - Retirer (x, y, λ) de la LIS

Refinement pass :

- Pour tous les coefficients (x, y, λ) de la LSP à l'exception de ceux ajoutés au cours de la dernière sorting pass : Écrire le t^{eme} bit le plus significatif de $c_{x, y, \lambda}$
- Décrémenter t et retourner à la sorting pass. ■

Le décodeur est obtenu en remplaçant *écrire* par *lire* dans l'algorithme précédent.

Le phénomène de tree-crossing a un impact plus grand dans le cas de SPIHT que dans le cas de EZW. Durant l'algorithme, un coefficient peut être traité avant qu'un de ses parents ne le soit, un soin particulier doit être pris pour éviter de traiter le même coefficient plusieurs fois. On garde donc en

	1.0 bpppb	0.5 bpppb
SPIHT-3D	73.55 dB	68.35 dB
SPIHT-3D-AC	73.89 dB	68.83 dB
SPIHT-spat	75.74 dB	69.97 dB
SPIHT-spat-AC	75.92 dB	70.05 dB

Tab. 3.7 Performances de SPIHT sur *moffett3* en PSNR. *3D* correspond à l'utilisation de la structure d'arbre 3D, *spat* à l'utilisation de la structure d'arbre spatial, *AC* signale l'utilisation d'un codeur arithmétique. Le phénomène de tree-crossing cause plus de problèmes que dans le cas de EZW, la structure d'arbre 3D ne donne pas de bons résultats dans ce cas.

mémoire le fait qu'un coefficient a déjà été traité. Cette information n'a pas besoin d'apparaître dans le train binaire de sortie, le décodeur partageant les états du codeur, il sera capable de faire la même opération. Le phénomène du tree-crossing a un impact plus important probablement parce que la destruction d'un arbre (par apparition d'un coefficient significatif) a des conséquences plus importantes pour SPIHT, qui essaie de maintenir des arbres au maximum (par le biais des arbres de degrés 1 et 2), que pour EZW. Grâce aux arbres de degrés 1 et 2, SPIHT tire également plus parti du lien spectral apparaissant uniquement pour les coefficients de basse fréquence spatiale.

Le codeur arithmétique utilisé pour EZW (voir paragraphe précédent) peut être intégré en sortie de cette extension de SPIHT.

D'après le tableau 3.7, on peut remarquer que la structure d'arbre 3D donne des résultats plus faibles que ce qu'on attendait et n'apporte pas d'amélioration par rapport à l'arbre spatial. Ces résultats surprenants sont dus au problème du tree-crossing qui est plus problématique dans le cas de SPIHT (on ne contrôle pas l'ordre de traitement des coefficients). Sur la figure 3.25 sont présentées les performances de SPIHT en utilisant la structure d'arbres 3D tandis que sur la figure 3.26, SPIHT2 correspond au codeur utilisant la structure d'arbre spatial. Tous des résultats sont commentés dans le paragraphe suivant.

3.4 Comparaison des résultats

JPEG 2000, ainsi que les adaptations de EZW et SPIHT sont appliqués sur les données hyperspectrales du capteur AVIRIS du JPL/NASA ainsi que sur le capteur spatial Hyperion de la NASA (en conservant toutes les bandes spectrales pour AVIRIS et en enlevant les bandes noires pour Hyperion : 224 bandes dans les deux cas). Les images sélectionnées sont présentées sur la figure 3.12. Une des scènes est également utilisée dans d'autres publications [Tan03].

La comparaison entre l'adaptation de JPEG 2000 (présenté dans la partie 3.1.2) et l'adaptation faite dans [Ruc05] ont des performances similaires.

Les adaptations de EZW et de SPIHT sont comparées avec JPEG 2000. EZW est utilisé avec la structure d'arbre 3D. SPIHT est d'abord comparé avec la structure d'arbre 3D (Fig. 3.25) puis est utilisé avec la structure d'arbre spatial (Fig. 3.26). Les résultats obtenus avec JPEG 2000 consistent en une étape de compression en utilisant plusieurs niveaux de qualité (*quality layers*) [ISO02]. 20 niveaux de qualité sont définis de 0.1 bpppb à 2.0 bpppb. La décompression est faite aux niveaux de qualité correspondants. Les performances de EZW et SPIHT sont calculées avec et sans codeur arithmétique. Les performances sont tracées en terme de PSNR (Eq. 2.11). L'image est complètement compressée, le train binaire sauvegardé sur disque puis décodé. Le PSNR est mesuré entre l'image originale et l'image en sortie (pas sur la distorsion de l'image transformée).

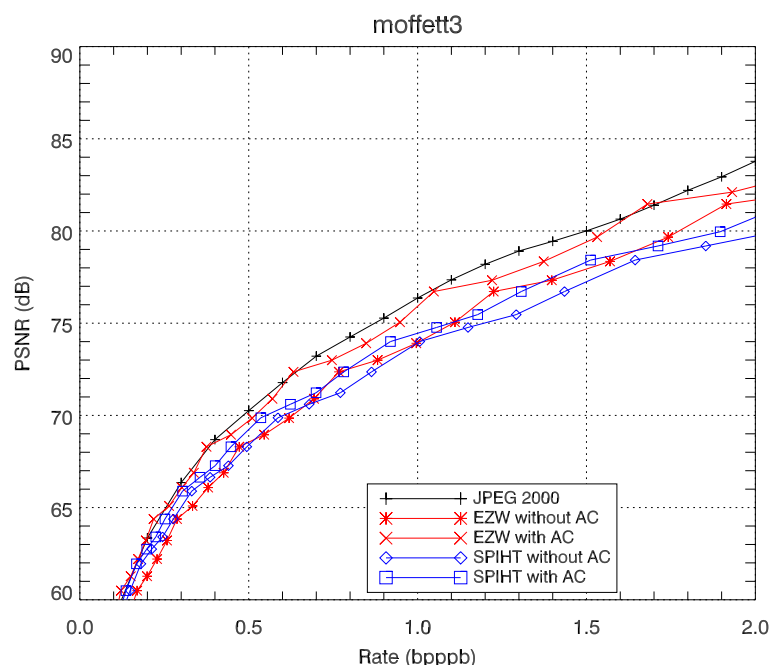
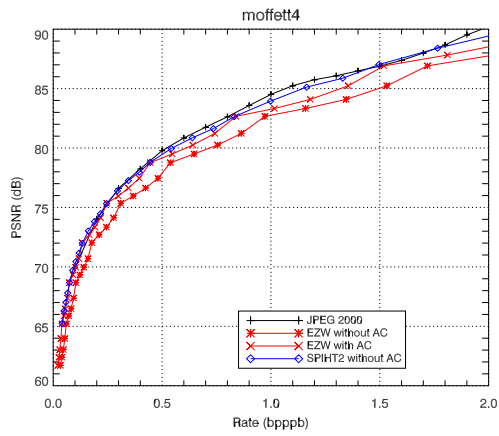
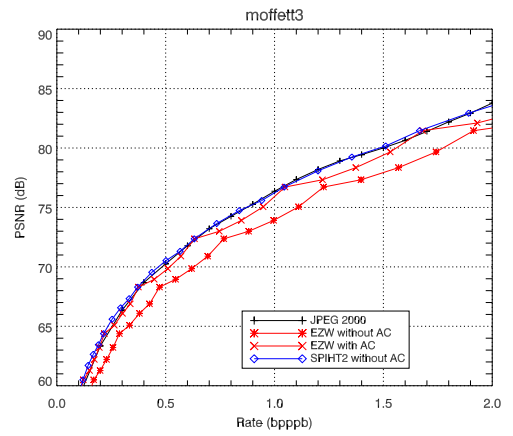


Fig. 3.25 Comparaison des performances de compression entre JPEG 2000 et EZW (resp. SPIHT) avec et sans codeur arithmétique.

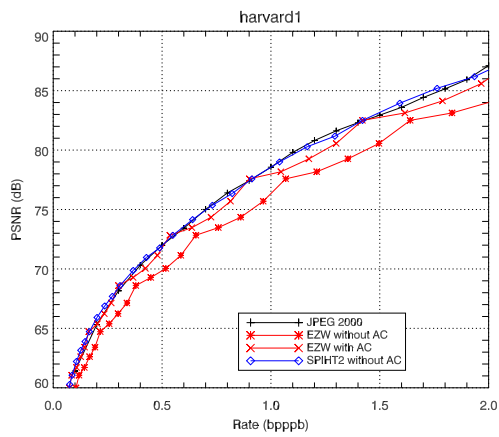
Les performances de EZW-3D sont bien meilleures lorsque la décompression atteint la fin d'un plan de bits. Les *pics* de performance correspondent aux points d'inflexion sur la figure 3.25. Cette caractéristique n'apparaît pas pour SPIHT. Utilisé avec un codeur arithmétique, EZW-3D produit des performances similaires à JPEG 2000 et même meilleures à certains débits. Les performances de SPIHT-3D sont étonnamment mauvaises pour des forts débits et tendent à dépasser EZW-3D pour des débits plus faibles. Cela peut être expliqué par le fait que SPIHT utilisé avec une structure d'arbre 3D est particulièrement sensible au *tree-crossing*.



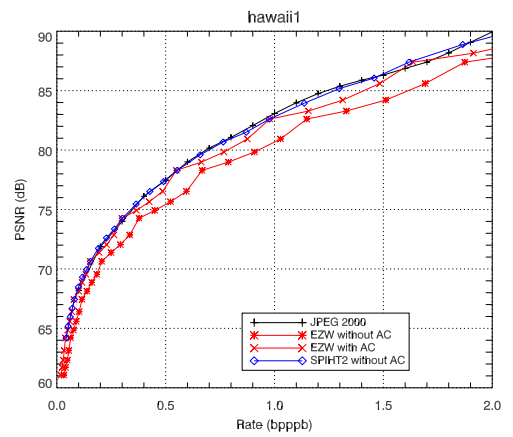
(a) moffett4



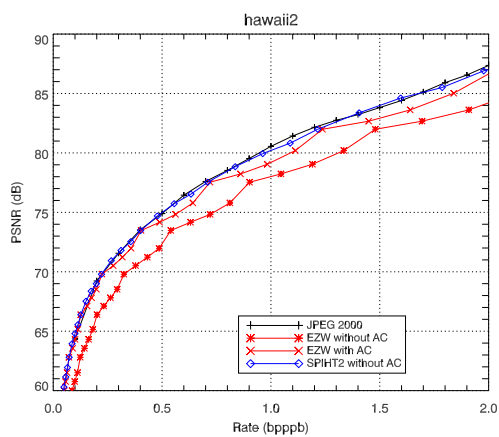
(b) moffett3



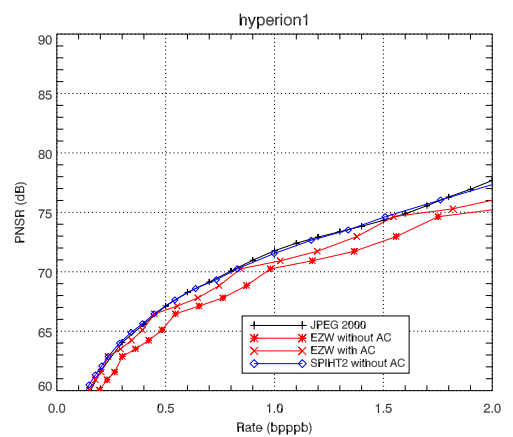
(c) Harvard1



(d) Hawaii1



(e) Hawaii2



(f) hyperion1

Fig. 3.26 Résultats sur 6 images hyperspectrales différentes. SPIHT-spat donne des performances très similaires à JPEG 2000 pour tout débit. Les résultats de SPIHT-spat sont montré ici sans codeur arithmétique brut, celui-ci n'améliorant pas significativement les résultats. EZW-3D ne présente pas d'aussi bonnes performances, mais avec un codeur arithmétique et un choix de troncature optimal, les performances sont semblables à JPEG 2000 et SPIHT-spat.

La solution la plus simple pour éviter le *tree-crossing* est d'utiliser la structure d'arbre spatial définie auparavant. Cette modification conduit à une structure similaire à celle trouvée dans [He03] et apporte une amélioration, comme on peut le voir sur la figure 3.26 (SPIHT-spat). Dans le cas de SPIHT, le codeur arithmétique apporte une amélioration faible. Les résultats sont donc présentés sans codeur arithmétique ce qui présente en plus l'avantage de réduire la complexité du codeur. Cette solution n'est pas entièrement satisfaisante car on peut voir que la relation spectrale entre les pixels n'est pas complètement utilisée. Cependant, conserver la structure 3D et tirer parti du *tree-crossing* demanderait de grandes modifications sur l'algorithme SPIHT.

3.5 Variations avec la notation binaire signée

3.5.1 Un inconvénient de EZW

Un des défauts de EZW est la mémoire requise pour mémoriser les coefficients qui ont été notés comme significatifs. Ces coefficients seront traités pendant la *refinement pass*, il n'est donc pas nécessaire de les traiter pendant la *significance pass*. Au moins un bit de mémoire est nécessaire pour chaque coefficient de l'image. Pour une image hyperspectrale de $256 \times 256 \times 224$, en comptant un bit de mémoire pour signaler la position des coefficients significatifs, on aura besoin de garder 14.7 Mbits en mémoire durant la compression. Si la compression est faite plan de bits par plan de bits (en gardant uniquement le plan de bits courant en mémoire), conserver cette information double la mémoire nécessaire. Une solution pour éliminer ce besoin de mémoire est de supprimer la *refinement pass*. Dans cette situation, on applique uniquement la *significant pass* pour chaque plan de bits. Un coefficient est considéré comme non significatif si le bit de ce coefficient dans ce plan de bits est 0. Il est considéré comme significatif sinon. Cependant, ce changement provoque une perte de performances de plus de 2 dB (Tableau 3.8).

Débit (bpppb)	3D-EZW		Sans <i>refinement pass</i>	
	MSE	PSNR	MSE	PSNR
1.0	106.15	76.07	193.73	73.46
0.5	445.22	69.84	685.49	67.97

Tab. 3.8 Effet de la suppression de la *refinement pass*. Les résultats sont pour l'image Moffett3.

3.5.2 Utilisation de la notation binaire signée

Comme nous l'avons vu, les bonnes performances des techniques par arbres de zéros proviennent principalement de leur capacité à coder une

grande quantité de zéros avec un seul symbole. Si tous les plans de bits sont codés par une *significance pass*, la probabilité d'avoir des zéros dans les plans de bits inférieurs devient proche de 0.5. D'autre part, ces zéros tendent à être distribués aléatoirement empêchant ainsi les arbres de zéros de les coder efficacement.

Une stratégie possible pour augmenter les performances des arbres de zéros est d'augmenter la proportion de zéros dans chaque plan de bits. La notation binaire signée permet de réaliser cela. La notation binaire signée d'un nombre n est une suite de chiffres $a = (\dots, a_2, a_1, a_0)$ avec $a_i \in \{-1, 0, 1\}$ tel que $n = \sum_{i=0}^{\infty} a_i 2^i$.

Le nombre 119, par exemple, en notation binaire classique est $(0, 1, 1, 1, 0, 1, 1, 1)$ car il est égal à $1 * 2^6 + 1 * 2^5 + 1 * 2^4 + 1 * 2^2 + 1 * 2^1 + 1 * 2^0$. Si -1 est utilisé en plus de 1 et 0, le nombre 119 peut être représenté par $(1, 0, 0, 0, -1, 0, 0, -1)$ car il est égal à $1 * 2^7 - 1 * 2^3 - 1 * 2^0$.

La représentation binaire signée pour un nombre donné n'est pas unique. En général, l'intérêt est d'avoir une représentation qui contient un maximum de 0. Cette représentation est obtenue en considérant la solution de poids de Hamming minimum. Le poids de Hamming d'une représentation est égal au nombre de chiffres non nul dans la représentation a . Dans l'exemple précédent, le poids de Hamming de la représentation de 119 en notation binaire classique est 6 tandis qu'il est de 3 dans la notation binaire proposée.

Dans [Arn93] un algorithme est donné pour obtenir la représentation binaire signée de poids de Hamming minimum :

Algorithme 4. *Représentation binaire signée*

```

t = 0
a = (... , a2, a1, a0), la notation binaire classique du nombre à convertir
Tant que (... , at+2, at+1, at) ≠ (... , 0, 0, 0)
- Si at ≠ 0
  - b = (... 0, sgn(at), -2*sgn(at), 0, ... , 0) (les éléments non nuls à t,
    t + 1)
  - c = a + b
  - si ct+1 = 0
    - a = c
  - t = t + 1
retourner a ■

```

Cet algorithme est simple, mais ce n'est pas le plus efficace en terme de complexité. Des algorithmes plus rapides existent pour réaliser cette conversion en nombre binaire signés de poids de Hamming minimum. On peut citer par exemple [Pro00, Joy00, Oke04].

Cependant la notation binaire signée de poids minimum n'est pas unique. En général, la notation binaire signée est utilisée pour l'exponentiation rapide. La forme non-adjacente (NAF), où les éléments différents de zéros

sont séparés par au moins un zéro, est unique et possède les propriétés nécessaires pour l'exponentiation rapide. La plupart des algorithmes de conversion conduisent à cette forme.

Dans notre cas, si le poids de Hamming minimum est une condition requise (le maximum de zéros), il n'est pas sûr que la NAF ait un avantage par rapport aux autres possibilités. Deux possibilités sont comparées dans la suite en utilisant la transformation $(\dots, 1, 0, -1, \dots) \rightarrow (\dots, 0, 1, 1, \dots)$ et de même $(\dots, -1, 0, 1, \dots) \rightarrow (\dots, 0, -1, -1, \dots)$. Nous appellerons cette seconde forme AF (Adjacent Form). Ces deux formes donnent le même nombre de zéros. Des exemples de représentation binaire signée pour le nombre 349 sont donnés dans le tableau 3.9.

t	9	8	7	6	5	4	3	2	1	0
2^t	512	256	128	64	32	16	8	4	2	1
Binaire	0	1	0	1	0	1	1	1	0	1
NAF	1	0	-1	0	-1	0	0	-1	0	1
AF	0	1	0	1	1	0	0	0	-1	-1

Tab. 3.9 Exemple de représentations pour le nombre 349

Pour mesurer l'efficacité à accroître la proportion de coefficients à zéros, on calcule cette proportion après le premier bit significatif. Pour la transformée en ondelettes de moffett3, le nombre moyen de bits après le premier bit significatif, le nombre de zéros après ce premier bit significatif et la proportion de bits à zéro sont détaillés dans le tableau 3.10. La notation binaire signée permet d'accroître de manière significative la proportion de zéros pour les plans de bits inférieurs : plus de 60% de zéros contre 50% avant. Ces résultats confirment ce qui était attendu pour la représentation binaire signée.

Notation	Nombre moy. de bits après le 1 ^{er} sig.	Nombre de bits à zéro	Proportion de bits à zéro
Binaire	2.72	20 490 955	51.28%
NAF	3.12	29 263 791	63.83%
AF	2.85	25 507 573	61.03%

Tab. 3.10 La proportion de bits à zéros après le premier bit significatif.

EZW est implémenté en utilisant la notation binaire signée (NAF et AF) et chaque plan de bits est traité en utilisant uniquement la *significant pass*. Cependant, même si on observe un gain de 1 dB en utilisant la notation binaire signée (Tableau 3.11), cette amélioration n'est pas suffisante pour récupérer la baisse de performances due à la suppression de la *refinement pass*. On ne retrouve pas les performances du tableau 3.8. Dans ce cas, on ne constate pas de différences entre les formes NAF et AF.

Débit (bpppb)	Binaire		NAF		AF	
	MSE	PSNR	MSE	PSNR	MSE	PSNR
1.0	193.73	73.46	149.07	74.60	151.76	74.52
0.5	685.49	67.97	549.56	68.93	553.10	68.90

Tab. 3.11 EZW avec traitement séparés de chaque plan de bits (sans *refinement pass*).

3.5.3 Utiliser les dépendances spatiales

Cette dernière version du codeur EZW ne prend pas en compte la valeur des coefficients voisins dans le même plan de bits. Un moyen simple de les prendre en compte est d'utiliser un codeur arithmétique contextuel. On considère uniquement trois coefficients dans le même plan de bits : ces coefficients sont ceux précédant le pixel courant dans les trois directions du cube hyperspectral.

On considère également la valeur du coefficient à la même place dans le plan de bits précédent. Dans le cas de la NAF, cette dépendance est facile à prendre en compte : si un coefficient dans le plan de bits précédent est 1 ou -1 , on sait que le coefficient courant est 0. Dans le cas de la forme AF, cette règle n'existe pas. On aurait donc à doubler le nombre de contextes selon les cas où le coefficient dans le plan de bits précédent appartient aux ensembles $\{0\}$ ou $\{-1, 1\}$. On choisit donc la forme NAF pour laquelle le contexte est plus simple.

Soient η_s , η_l et η_b , les coefficients précédents dans les trois directions :

- $\eta_s(i, j, k) = (i - 1, j, k)$
- $\eta_l(i, j, k) = (i, j - 1, k)$
- $\eta_b(i, j, k) = (i, j, k - 1)$

Comme les plans de bits sont pris en compte séparément, η_s , η_l et η_b font partie de l'ensemble $\{-1, 0, +1\}$. On considère la fonction qui, à une combinaison de voisinage, associe une valeur η définie par $\eta = \eta_s + 3\eta_l + 9\eta_b$. Cette fonction est une bijection entre tous les voisinages possibles et les entiers entre -13 et 13 .

On peut donc étudier la probabilité d'avoir une des valeurs -1 , 0 ou 1 selon les valeurs voisines. La courbe de probabilité est représentée sur la figure 3.27. Ces probabilités sont calculées pour l'image moffett3 de $256 \times 256 \times 224$ pour tous les plans de bits pour la forme NAF : plusieurs millions de bits sont donc pris en compte. On peut voir qu'un voisinage se distingue clairement des autres en terme de probabilité : quand $\eta = 0$ ($\eta_s = \eta_l = \eta_b = 0$). Avec ce voisinage, la probabilité d'avoir un 0 pour le coefficient courant est très élevée.

Le contexte pour le codeur arithmétique est donc séparé en deux cas : $\eta = 0$ et $\eta \neq 0$.

On peut noter également que dans le cas de la NAF, une valeur différente de 0, pour un coefficient dans un plan de bits donné, sera nécessairement suivie par un coefficient à 0 dans le plan de bits suivant (d'où son nom

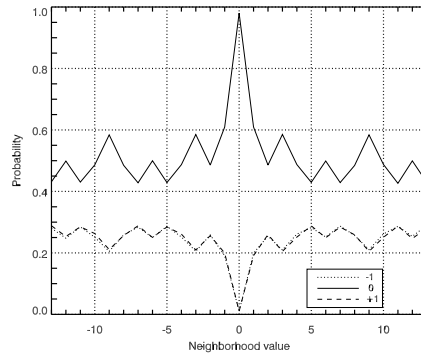


Fig. 3.27 Probabilité d'avoir les valeurs -1 , 0 ou 1 pour le coefficient courant selon les valeurs des voisins pour la forme NAF. Les 27 voisinages possibles sont présentés en abscisse selon la valeur de η .

de *non-adjacent form*). Dans ce cas, il n'est pas nécessaire de sortir une information. Cet avantage n'apparaît pas pour la forme AF qui ne donne pas d'aussi bonnes performances. Les performances obtenues sont présentées dans le tableau 3.12

Débit (bpppb)	Non contextuel		Contextuel	
	MSE	PSNR	MSE	PSNR
1.0	149.07	74.60	121.38	75.49
0.5	549.56	68.93	457.77	69.72

Tab. 3.12 EZW avec un codage indépendant pour chaque plan de bits en forme NAF avec et sans codage contextuel.

On nomme cette dernière version de EZW sans *refinement pass* utilisant une forme NAF et un codage arithmétique contextuel 3D-EZW-NAF.

La comparaison entre 3D-EZW-NAF et 3D-EZW est présentée dans le tableau 3.13 pour l'image moffett3. Les courbes débit-distorsion sont illustrées sur la figure 3.28.

Débit (bpppb)	3D-EZW		3D-EZW-NAF	
	MSE	PSNR	MSE	PSNR
1.0	106.15	76.07	121.38	75.49
0.5	445.22	69.84	457.77	69.72
0.25	1407.34	64.85	1514.81	64.53
0.125	3933.86	60.38	4402.34	59.89

Tab. 3.13 Comparaison entre 3D-EZW et 3D-EZW-NAF sur l'image moffett3.

Les performances de 3D-EZW-NAF sont très proches de celle de 3D-EZW. En terme de complexité calculatoire, une estimation précise serait requise, mais un moyen simple d'avoir une première estimation est de comparer les temps d'exécution. Le temps de codage est similaire entre 3D-EZW

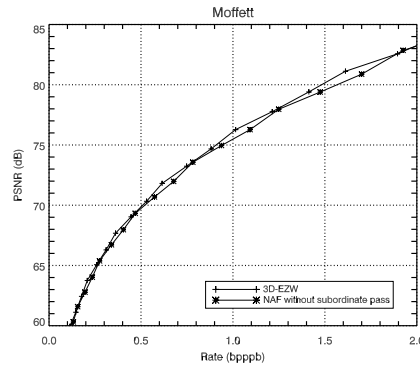


Fig. 3.28 Comparaison des performances entre 3D-EZW et 3D-EZW-NAF.

et 3D-EZW-NAF : environ 100 s pour les deux versions. La conversion en notation binaire signée n'est pas optimisée dans notre cas (ajoutant environ 24 s) mais pourrait être optimisée en utilisant un algorithme plus performant disponible dans la littérature. Comme l'une des utilisations majeures de la notation binaire signée est d'accélérer l'exponentiation rapide, une conversion rapide ne sera pas un problème.

Il est à noter que l'intérêt principal de cet algorithme est qu'il est facilement parallélisable sans compromis de rapidité/complexité. On peut imaginer utiliser différentes unités de codage pour coder chaque plan de bits. Chacune de ces unités prendra en entrée un plan de bits et sortira la portion du train binaire correspondant à ce plan de bits.

3.6 Conclusion

Dans ce chapitre, différentes méthodes de compression pour les images hyperspectrales sont développées puis comparées. En premier lieu, une méthode est définie pour trouver la décomposition optimale (au sens débit-distorsion) en ondelettes anisotropiques 3D pour les images hyperspectrales. Cette méthode justifie l'utilisation d'une décomposition fixée particulière. Sur cette décomposition, plusieurs structures d'arbre peuvent être définies. Une étude statistique sur la proportion d'arbre de zéros est faite pour différentes structures possibles et conduit à un choix de structure d'arbre 3D. Ensuite, différents algorithmes de compression basés sur cette transformation et cette structure d'arbre sont définis. La première méthode est une adaptation de l'algorithme EZW, la seconde une adaptation de SPIHT. Leurs résultats sont comparés au standard JPEG 2000. Même si l'optimisation débit-distorsion incluse dans le standard JPEG 2000 rend difficile son implémentation dans le contexte spatial, ses performances peuvent être vues comme un objectif à atteindre.

Cette étude monte l'intérêt des arbres de zéros adaptés sur une décomposition anisotropique pour la compression des images hyperspectrales. Avec une complexité faible, EZW pour une structure d'arbre 3D présente des performances très proches de JPEG 2000 tout en produisant un train binaire emboîté. SPIHT-spat donne de bonnes performances même sans codeur arithmétique. Ces propriétés sont particulièrement intéressantes dans le contexte de la compression bord des images hyperspectrales. Des améliorations sont toujours possibles, particulièrement dans le cas de SPIHT si on arrive à trouver un moyen de tirer parti de la structure d'arbre 3D.

La notation binaire signée, particulièrement la *non-adjacent form*, a montré de bonne capacité à compenser la suppression de la *refinement pass*. Cette compensation a permis à un algorithme simplifié de donner des performances presque aussi bonnes que l'algorithme de départ. Cette utilisation originale de la notation binaire signée semble prometteuse et pourrait être appliquée à d'autres algorithmes de compression.

Chapitre 4

Vers une plus grande flexibilité

4.1	Décomposition et arbre utilisés	108
4.2	Le codage par groupes	111
4.2.1	Pourquoi?	111
4.2.2	Comment?	111
4.3	Permettre la progression en résolution	113
4.3.1	Introduction de la progression	113
4.3.2	Illustration sur un exemple	115
4.3.3	Permutation des progressions	117
4.4	Les désavantages du codage par groupes.	119
4.4.1	Conservation de la progression en qualité	119
4.4.2	Organisation en couches et débit-distorsion.	120
4.4.3	Connaître la distorsion pendant la compression	121
4.4.4	La formation du train binaire final	122
4.5	Résultats	123
4.5.1	Données	123
4.5.2	Performances de compression	124
4.5.3	Flexibilité	126
4.5.4	Codage de régions d'intérêt.	127
4.6	Conclusions	129



O FFRIR plus de flexibilité à l'utilisateur d'un algorithme de compression est probablement ce qui va diriger l'évolution des futurs algorithmes [Pea01b]. On souhaite ainsi pouvoir décoder une image à une résolution plus faible en accédant à un minimum de bits dans le train binaire, on désire également accéder à une partie de l'image (en spatial ou en spectral) sans avoir à décoder toute l'image. Ces propriétés sont obtenues tout en réduisant les ressources (mémoire) nécessaires à la compression et à la décompression et en posant les bases pour une compression au fil de l'eau. Cette partie de l'étude a fait l'objet de publications [Chr06d, Chr06e].

Les propriétés de codage progressif en qualité (Fig. 4.2), en résolution (Fig. 4.3) et de résistance aux erreurs (Fig. 4.4) sont illustrées sur une image classique 2D (Fig. 4.1). Ces propriétés sont intéressantes pour des images satellites. En effet, un codage progressif en résolution est utile lors de la distribution ou de la visualisation des images : il n'est pas toujours nécessaire d'avoir l'image à pleine résolution que ce soit en spatial ou en spectral. Par exemple, la détection de nuages dans les images se fait couramment sur des images à faible résolution (*quicklook*). La résistance aux erreurs est particulièrement séduisante pour des applications de type sondes lointaines où le débit disponible est très faible et pour lesquelles la reprogrammation des prises de vues pose des problèmes particuliers (temps de transmission). Certains travaux sont consacrés à ces adaptations comme [Oli03, Cho05c, Xie05].



Fig. 4.1 Barbara et détail.



Fig. 4.2 Barbara et codage progressif en qualité (débit 0.05, 0.1, 0.5 bpp).



Fig. 4.3 Barbara et codage progressif en résolution (1/8, 1/4, 1/2).

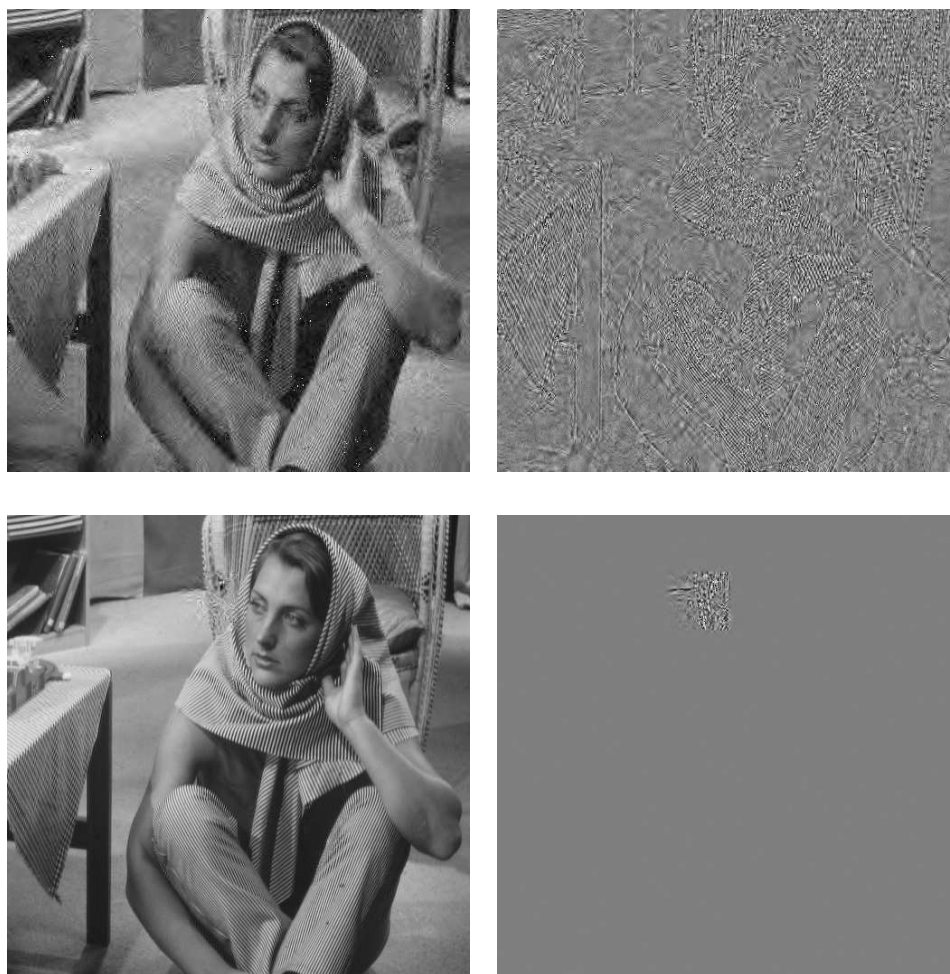


Fig. 4.4 Barbara et résistance aux erreurs : sans dispositif particulier (en haut) et avec la stratégie de résistance aux erreurs décrite dans la suite (en bas). L'image décodée est représentée à gauche et la différence par rapport à l'image d'origine à droite. L'erreur ne touche qu'un seul octet du train binaire.

4.1 Décomposition et arbre utilisés

Selon les résultats du chapitre précédent, la structure d'arbre spatial couplée à la décomposition anisotrope donne les résultats les plus performants pour SPIHT. C'est donc cette structure qui va être utilisée dans ce chapitre (Fig. 4.5).

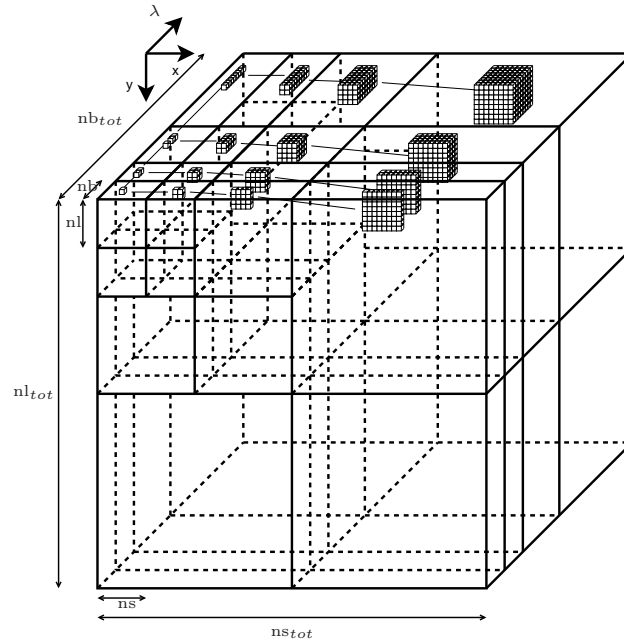


Fig. 4.5 Illustration de la structure d'arbre utilisée. Tous les descendants pour un coefficient (i, j, k) avec i et k impairs et j pair sont représentés.

Rappelons que $\mathcal{O}(i, j, k)$ note l'ensemble des enfants du coefficient (i, j, k) du cube hyperspectral après la décomposition présentée sur la figure 4.5. On sépare ses enfants entre les enfants spatiaux $\mathcal{O}_{spat}(i, j, k)$ et les enfants spectraux $\mathcal{O}_{spec}(i, j, k)$. On appelle ns le nombre d'échantillons (colonnes) dans la sous-bande LLL et nl le nombre de lignes (ns et nl sont les dimensions en spatial de la sous-bande LLL). On note ns_{tot} et nl_{tot} le nombre total de coefficients sur une ligne ou une colonne (taille du cube hyperspectral). Ces notations sont illustrées sur la figure 4.5. On note aussi d_{spat} et d_{spec} le nombre de décompositions de la transformée en ondelettes dans les directions spatiales et spectrale respectivement. Sur la figure 4.5, on a $d_{spat} = d_{spec} = 3$ par souci de clarté ; en pratique, pour les résultats présentés et sauf mention contraire, on prend $d_{spat} = d_{spec} = 5$.

Pour la descendance spatiale, on a une relation de descendance semblable au SPIHT original, comme illustré sur la figure 4.6. À part dans les

sous-bandes de plus basse fréquence et de plus haute fréquence, on a

$$\mathcal{O}_{spat}(i, j, k) = \{(2i, 2j, k), (2i + 1, 2j, k), (2i, 2j + 1, k), (2i + 1, 2j + 1, k)\}.$$

Les sous-bandes de plus haute fréquence n'ont pas de descendance :

$$\mathcal{O}_{spat}(i, j, k) = \emptyset \text{ si } i \geq \frac{ns_{tot}}{2} \text{ ou } j \geq \frac{nl_{tot}}{2}$$

Et pour les plus basses fréquences ($i < ns$ et $j < nl$), les coefficients sont rassemblés par groupes de 2×2 (en spatial) comme dans le SPIHT original (Fig. 4.6). On a alors :

- si i pair et j pair :

$$\mathcal{O}_{spat}(i, j, k) = \emptyset$$

- si i impair et j pair :

$$\mathcal{O}_{spat}(i, j, k) = \{(i + ns, j, k), (i + ns + 1, j, k), (i + ns, j + 1, k), (i + ns + 1, j + 1, k)\}$$

- si i pair et j impair :

$$\mathcal{O}_{spat}(i, j, k) = \{(i, j + nl, k), (i + 1, j + nl, k), (i, j + nl + 1, k), (i + 1, j + nl + 1, k)\}$$

- si i impair et j impair :

$$\mathcal{O}_{spat}(i, j, k) = \{(i + ns, j + nl, k), (i + ns + 1, j + nl, k), (i + ns, j + nl + 1, k), (i + ns + 1, j + nl + 1, k)\}$$

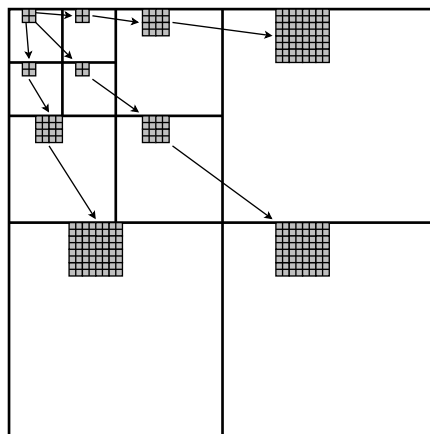


Fig. 4.6 Structure de groupe équivalente en 2D : tous les coefficients en gris appartiennent au même groupe. Dans l'algorithme décrit par la suite, une structure équivalente en 3D est utilisée.

La descendance spectrale de $\mathcal{O}_{spec}(i, j, k)$ est définie de manière similaire, mais seulement pour les plus basses fréquences spatiales. On définit de manière similaire nb comme étant le nombre de coefficients en spectral dans la sous-bande LLL et nb_{tot} le nombre de coefficients dans la direction spectrale pour tout le cube.

Si $i \geq ns$ ou $j \geq nl$ on a $\mathcal{O}_{spec}(i, j, k) = \emptyset$.

Sinon, sauf pour les plus hautes et plus basses fréquences, on a

$$\mathcal{O}_{spec}(i, j, k) = \{(i, j, 2k), (i, j, 2k + 1)\} \text{ si } nb \leq k < \frac{nb_{tot}}{2}.$$

Pour les plus hautes fréquences, il n'y a pas de descendance :

$$\mathcal{O}_{spec}(i, j, k) = \emptyset \text{ si } k \geq \frac{nb_{tot}}{2}$$

et pour les basses fréquences ($k < nb$), les coefficients sont groupés par 2 (en spectral) par analogie avec la construction de SPIHT :

- si $i < ns, j < nl, k$ pair (et $k < nb$) : $\mathcal{O}_{spec}(i, j, k) = \emptyset$;
- si $i < ns, j < nl, k$ impair (et $k < nb$) : $\mathcal{O}_{spec}(i, j, k) = \{(i, j, k + nb), (i, j, k + nb + 1)\}$.

Dans le cas d'un nombre impair de coefficients au niveau de la sous-bande LLL (si nb est un nombre impair), la définition précédente est légèrement modifiée et les coefficients du dernier plan spectral de la sous-bande LLL auront un seul descendant. Le regroupement des pixels dans la LLL en groupes de 2×2 en spatial et par 2 en spectral donne naturellement un groupe de $2 \times 2 \times 2$ en 3D.

Avec ces définitions concernant la descendance, tous les coefficients du cube hyperspectral appartiennent à un arbre et un seul. Chacun des coefficients est le descendant d'un unique coefficient racine situé dans la sous-bande LLL (avec la structure représentée sur la figure 4.5). Il est à noter que tous les coefficients appartenant au même arbre correspondent à une zone précise de l'image originale dans les 3 dimensions.

On peut calculer le nombre maximum de descendants pour un coefficient racine (i, j, k) pour 5 niveaux de décomposition spatial et spectral ($d_{spat} = d_{spec} = 5$). Le nombre de descendants est maximum lorsque k est impair et que, au moins i ou j est impair. Dans ce cas, on a

$$1 + 2 + 2^2 + \dots + 2^5 = 2^6 - 1$$

descendants spectraux et pour chacun d'eux on a

$$1 + 2^2 + (2^2)^2 + (2^3)^2 + \dots + (2^5)^2 = 2^0 + 2^2 + 2^4 + \dots + 2^{10} = (2^{12} - 1)/3$$

descendants spatiaux. On peut différencier le nombre de décompositions en spectral et en spatial, avec d_{spec} le nombre de décompositions dans la

direction spectrale et d_{spat} dans la direction spatiale, on obtient la formule générale :

$$n_{desc} = (2^{d_{spec}+1} - 1) \frac{2^{2(d_{spat}+1)} - 1}{3} \quad (4.1)$$

Le nombre de coefficients maximum dans un arbre est au plus 85995 (pour $d_{spec} = 5$ et $d_{spat} = 5$).

4.2 Le codage par groupes

4.2.1 Pourquoi ?

Pour permettre l'accès aléatoire à une portion quelconque de l'image hyperspectrale, il est nécessaire de coder séparément les différentes zones de l'image. Coder séparément les différentes portions d'une image a plusieurs avantages. D'abord, c'est une condition requise pour la compression au fil de l'eau. Ensuite, cette séparation permet d'utiliser des paramètres de compression différents en fonction de la zone de l'image, cette propriété rend possible le codage par régions d'intérêt (ROI) et le rejet (ou le codage à très basse qualité) de portions inutiles de l'image. Pour une image satellite, une partie inutile peut être une zone de nuages, pour une image médicale de type résonance magnétique, la zone inutile peut être un organe voisin de celui qui doit être observé. Un autre avantage est que les erreurs de transmission ont un impact plus limité si les zones de l'image sont codées séparément : l'erreur affecte uniquement une partie de l'image. Enfin, un des facteurs limitant de l'algorithme SPIHT est la quantité de mémoire nécessaire pour stocker les listes lors de la compression. Si le codage est fait sur des portions de l'image, le nombre maximum de coefficients en mémoire est réduit de manière très importante réduisant la mémoire requise.

Bien sûr, il y a également un désavantage à faire du codage par groupe et celui-ci sera détaillé par la suite. On souligne que seul le codage est fait par groupe, la transformée étant, elle, appliquée sur toute l'image. Il n'y a donc pas d'effet de bloc comme on peut le trouver dans le standard JPEG original.

4.2.2 Comment ?

Avec la structure d'arbre définie précédemment, des groupes apparaissent naturellement. Un groupe \mathcal{G}_k est défini par 8 coefficients de la sous-bande LLL formant un cube de $2 \times 2 \times 2$ coefficients ainsi que tous leurs descendants. Grouper les coefficients par 8 permet de tirer parti des similarités des coefficients dans un voisinage. L'idée est similaire au codage des coefficients par groupe de 2×2 qui figure dans le brevet de SPIHT [Pea98]

(voir Fig. 4.6) mais son exploitation est différente. Dans le brevet de SPIHT, ce groupement permet d'inclure un codage de Huffman car les valeurs des 4 coefficients sont fortement liées (corrélation spatiale résiduelle). Les résultats ci-après ne prennent pas en compte cette possibilité d'amélioration.

Un autre avantage de ce groupement est que le nombre de coefficients dans chaque groupe sera le même, la seule exception étant le cas où au moins une des dimensions de la sous-bande LLL est impaire. Le nombre de coefficients dans chaque groupe peut être calculé. Dans un groupe racine de $2 \times 2 \times 2$ (Fig. 4.7), on a 3 coefficients qui ont un ensemble complet de descendants (coefficients 5, 6 et 7) dont le nombre est donné par (4.1), 3 ont seulement des descendants spatiaux (coefficients 1, 2 et 3), 1 a uniquement des descendants spectraux (coefficient 4) et le dernier n'a pas de descendant (coefficient 0). Le nombre de coefficients dans un groupe, qui est lié à la quantité de mémoire nécessaire, sera donc finalement de $262144 = 2^{18}$ (pour une décomposition à 5 niveaux en spectral et en spatial).

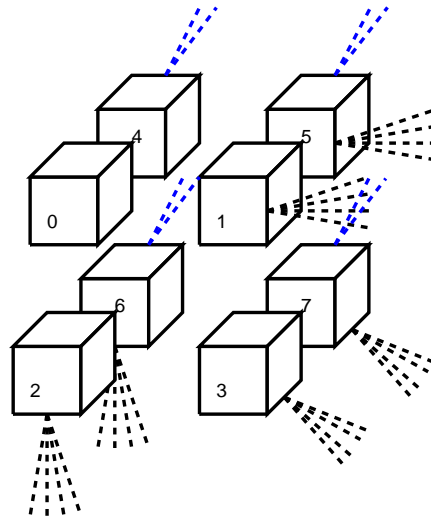


Fig. 4.7 Descendance des différents coefficients d'un groupe \mathcal{G}_k de $2 \times 2 \times 2$ coefficients de la sous-bande LLL.

Chacun de ces groupes \mathcal{G}_k va être codé en utilisant une version modifiée de l'algorithme SPIHT décrit dans la partie suivante. Il faut souligner que ces groupes n'ont rien à voir avec la notion de blocs utilisés par JPEG 2000. Ce dernier regroupe en blocs uniquement des pixels appartenant à la même sous-bande de la transformée en ondelettes et uniquement dans un seul plan spectral. L'algorithme SPIHT utilisant cette séparation en groupes pour le codage pour permettre l'accès aléatoire sera dénoté RA (*Random Access*) par la suite.

4.3 Permettre la progression en résolution

4.3.1 Introduction de la progression

On ne rappelle pas les détails de l'algorithme SPIHT original qui peuvent être trouvés dans la section 3.3.5. Un exemple pratique est décrit dans l'annexe B. Pour un codage progressif en résolution, on considère, ce qui est classique [Kim00, Dan03], que la sous-bande basse fréquence de la transformée en ondelettes est une version sous-échantillonnée de l'image. Pour obtenir une image à basse résolution, il suffit alors d'appliquer la transformée en ondelettes inverse (IDWT) sur les coefficients des sous-bandes de basse fréquence.

Durant le codage, SPIHT ne fait pas de distinction entre les différents niveaux de résolution. Pour pouvoir fournir différents niveaux de résolution, on doit traiter séparément chaque résolution. Pour permettre cela, on garde 3 listes pour chaque niveau de résolution R_r . Quand $r = 0$, seulement le plus haut niveau de la pyramide de la transformée en ondelettes sera traité (les coefficients de LLL). Pour une décomposition à 5 niveaux en spatial et en spectral, un total de 36 niveaux de résolution sera disponible comme il est illustré sur la figure 4.8 (pour des raisons de clarté sont représentés sur cette figure uniquement 3 niveaux de décomposition, soit 16 niveaux de décomposition). Chaque niveau R_r conserve 3 listes en mémoire : LSP_r , LIP_r et LIS_r .

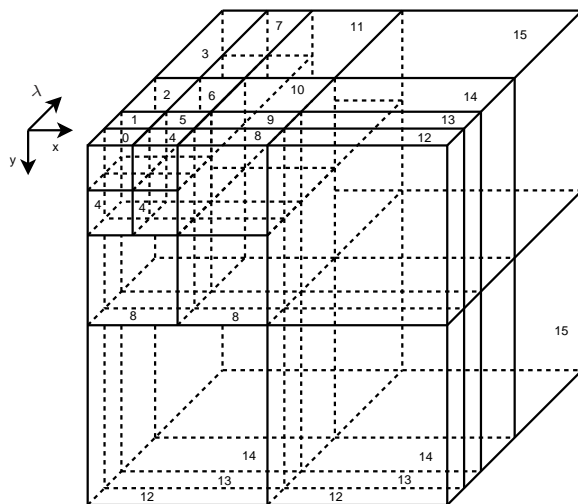


Fig. 4.8 Numérotation des résolutions. Si une image à une résolution plus faible est demandée (en spectral ou en spatial), seules les sous-bandes correspondant à la demande sont décodées.

Des difficultés apparaissent avec cette organisation. Si la priorité est donnée à un codage progressif en résolution (par rapport au codage progressif en

qualité), des précautions supplémentaires doivent être prises. Les différentes possibilités pour les ordres de progression sont détaillées dans la partie suivante. On note R_{r_d} la résolution des descendants de R_r : par exemple, la résolution descendante de R_8 est R_{12} pour l'exemple de la figure 4.8. Dans le cas le plus compliqué, lorsque tous les plans de bits pour une résolution R_r sont traités avant la résolution des descendants r_d (*full resolution scalability*), le dernier élément à traiter pour LSP_{r_d} , LIP_{r_d} et LIS_{r_d} pour chaque plan de bit t doit être enregistré. Ce problème est illustré sur un exemple dans le paragraphe 4.3.2.

Les détails de l'algorithme sont donnés ci-dessous. On rappelle que $S_t(i, j, k) = 0$ si tous les descendants sont inférieurs à 2^t , que les arbres de type A correspondent à des arbres de zéros de degré 1 (Fig. 3.17) et que les arbres de type B correspondent à des arbres de zéros de degré 2 (Fig. 3.18).

Algorithme 5. *Resolution scalable 3D SPIHT (3D-SPIHT-RS)*

Initialisation :

- Initialiser t au nombre de plans de bits :
- $LSP_0 = \emptyset$
- LIP_0 : tous les coefficients sans aucun parent (les 8 coefficients racine du groupe)
- LIS_0 : tous les coefficients de la LIP_0 ayant des descendants (7 sur les 8, comme un seul n'a pas de descendant).
- Pour $r \neq 0$, $LSP_r = LIP_r = LIS_r = \emptyset$

Sorting pass :

Pour chaque r variant de 0 à la résolution maximale

Pour chaque t variant du nombre de plans de bits à 0

- Pour tout (i, j, k) de LIP_r qui a été ajouté à un seuil strictement supérieur au t courant
 - Écrire $S_t(i, j, k)$
 - Si $S_t(i, j, k) = 1$, déplacer (i, j, k) dans LSP_r à la bonne position et écrire le signe de $c_{i,j,k}$
- Pour tout (i, j, k) de LIS_r qui a été ajouté à un plan de bits supérieur au plan de bits courant
 - Si l'entrée est de type A
 - Écrire $S_t(\mathcal{D}(i, j, k))$
 - Si $S_t(\mathcal{D}(i, j, k)) = 1$ alors
 - Pour tout $(i', j', k') \in \mathcal{O}(i, j, k)$: écrire $S_t(i', j', k')$; Si $S_t(i', j', k') = 1$, ajouter (i', j', k') à LSP_{r_d} et écrire le signe de $c_{i',j',k'}$ sinon, ajouter (i', j', k') à la fin de LIP_{r_d} .
 - Si $\mathcal{L}(i, j, k) \neq \emptyset$, déplacer (i, j, k) à la bonne position de LIS_r comme une entrée de type B
 - Sinon, enlever (i, j, k) de LIS_r
 - Si l'entrée est de type B
 - Écrire $S_t(\mathcal{L}(i, j, k))$
 - Si $S_t(\mathcal{L}(i, j, k)) = 1$

- Ajouter tous les $(i', j', k') \in \mathcal{O}(i, j, k)$ à la bonne position de LIS_{r_d} comme entrée de type A
- Enlever (i, j, k) de LIS_r

Refinement pass :

- Pour tout (i, j, k) de LSP_r qui a été ajouté à un seuil strictement supérieur au t courant : écrire le t^{eme} bit le plus significatif de $c_{i,j,k}$.

■

Ce nouvel algorithme donne strictement la même quantité de bits que la version originale de SPIHT. Les bits sont juste organisés dans un ordre différent. Avec la structure en groupe, la mémoire nécessaire est grandement réduite. La progression en résolution, même si elle utilise plus de listes, n'augmente pas la mémoire nécessaire car les coefficients sont juste répartis entre les différentes listes.

4.3.2 Illustration sur un exemple

Le déroulement de l'algorithme est illustré sur un exemple simple (Fig. 4.9). Les termes de codage progressif en qualité et codage progressif en résolution seront précisés dans le paragraphe 4.3.3.

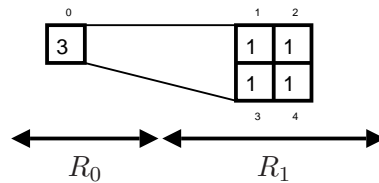


Fig. 4.9 Illustration des codages progressifs en qualité et en résolution pour 5 coefficients.

- Les coefficients 1, 2, 3 et 4 sont des descendants de 0.
- Le coefficient 0 à la résolution R_0 a la valeur 3.
- Les coefficients 1, 2, 3 et 4 à la résolution R_1 ont la valeur 1.
- Deux niveaux de résolution R_0 et R_1 .
- Deux plans de bits : $t = 1$ avec un seuil de 2 et $t = 0$ avec un seuil de 1.

Regardons les différences entre le codage progressif en qualité et le codage progressif en résolution.

4.3.2.1 Codage progressif en qualité

- A la fin de R_0 pour $t = 1$: le seul élément significatif est le coefficient 0.

$LSP_0 = \{0\}$	$LSP_1 = \emptyset$
$LIP_0 = \emptyset$	$LIP_1 = \emptyset$
$LIS_0 = \{0\}$	$LIS_1 = \emptyset$

- A la fin de R_1 pour $t = 1$: aucun élément significatif dans R_1 , pas de changement.

$$\begin{array}{ll} \text{LSP}_0 = \{0\} & \text{LSP}_1 = \emptyset \\ \text{LIP}_0 = \emptyset & \text{LIP}_1 = \emptyset \\ \text{LIS}_0 = \{0\} & \text{LIS}_1 = \emptyset \end{array}$$
- A la fin de R_0 pour $t = 0$: le coefficient 0 n'est plus la racine d'un arbre de 0.

$$\begin{array}{ll} \text{LSP}_0 = \{0\} & \text{LSP}_1 = \{1,2,3,4\} \\ \text{LIP}_0 = \emptyset & \text{LIP}_1 = \emptyset \\ \text{LIS}_0 = \emptyset & \text{LIS}_1 = \emptyset \end{array}$$
- A la fin de R_0 pour $t = 0$: pas de changement.

$$\begin{array}{ll} \text{LSP}_0 = \{0\} & \text{LSP}_1 = \{1,2,3,4\} \\ \text{LIP}_0 = \emptyset & \text{LIP}_1 = \emptyset \\ \text{LIS}_0 = \emptyset & \text{LIS}_1 = \emptyset \end{array}$$

On voit que dans ce cas, le parcours est très semblable au SPIHT classique et aucune précaution particulière n'est nécessaire.

4.3.2.2 Codage progressif en résolution

- A la fin de R_0 pour $t = 1$: le seul élément significatif est le coefficient 0.

$$\begin{array}{ll} \text{LSP}_0 = \{0\} & \text{LSP}_1 = \emptyset \\ \text{LIP}_0 = \emptyset & \text{LIP}_1 = \emptyset \\ \text{LIS}_0 = \{0\} & \text{LIS}_1 = \emptyset \end{array}$$
- A la fin de R_0 pour $t = 0$: le coefficient 0 n'est plus la racine d'un arbre de 0.

$$\begin{array}{ll} \text{LSP}_0 = \{0\} & \text{LSP}_1 = \{1,2,3,4\} \\ \text{LIP}_0 = \emptyset & \text{LIP}_1 = \emptyset \\ \text{LIS}_0 = \emptyset & \text{LIS}_1 = \emptyset \end{array}$$
- A la fin de R_1 pour $t = 1$: Le point critique est ici ! Il ne faut surtout pas traiter les éléments de la LSP_1 qui ont été ajoutés pendant le seuil t_0 .

$$\begin{array}{ll} \text{LSP}_0 = \{0\} & \text{LSP}_1 = \{1,2,3,4\} \\ \text{LIP}_0 = \emptyset & \text{LIP}_1 = \emptyset \\ \text{LIS}_0 = \{0\} & \text{LIS}_1 = \emptyset \end{array}$$
- A la fin de R_0 pour $t = 0$: pas de changement.

$$\begin{array}{ll} \text{LSP}_0 = \{0\} & \text{LSP}_1 = \{1,2,3,4\} \\ \text{LIP}_0 = \emptyset & \text{LIP}_1 = \emptyset \\ \text{LIS}_0 = \emptyset & \text{LIS}_1 = \emptyset \end{array}$$

Pour éviter de traiter les éléments de la LSP_1 ajoutés au seuil $t = 0$, on est obligé d'ajouter un marqueur indiquant quand traiter les éléments durant la compression. Il n'y a pas d'incidence bien sûr sur la taille du train binaire.

4.3.3 Permutation des progressions

Dans la partie précédente, nous avons présenté la version la plus délicate de l'algorithme. Traiter d'abord une résolution complètement avant de passer à la suivante (Fig. 4.10 (b)) demande plus de précautions que de traiter les coefficients plan de bits par plan de bits (Fig. 4.10 (a)). Le train binaire obtenu par cet algorithme est illustré sur la figure 4.11 et nommé *organisation progressive en résolution*. On notera cet algorithme **SPIHT-RARS** (RS = *Resolution Scalable*), **le but de cette version étant de fournir le maximum de flexibilité et notamment la séparation des résolutions spatiale et spectrales**. Si la progression en résolution n'est plus une priorité et que c'est plutôt une progression en qualité qui est demandée, les boucles 'for' de l'algorithme sur les résolutions et les plans de bits peuvent être inversées. On traitera alors complètement un plan de bits avant de passer à la résolution suivante. Dans ce cas, le train binaire est différent et illustré sur la figure 4.12. Cette organisation est nommée *progressive en qualité*. Cette progression en qualité donnera un train binaire très proche du SPIHT original, l'ordre de traitement au sein d'un plan de bit aura juste la contrainte supplémentaire de traiter les résolutions les unes après les autres. On notera toujours cet algorithme **SPIHT-RA** même si le codage et le décodage peuvent être effectués en sélectionnant la résolution pour souligner que **l'objectif principal de cette version est d'obtenir la meilleure qualité en lisant uniquement le début du train binaire**. La différence du sens de parcours est illustré sur la figure 4.10.

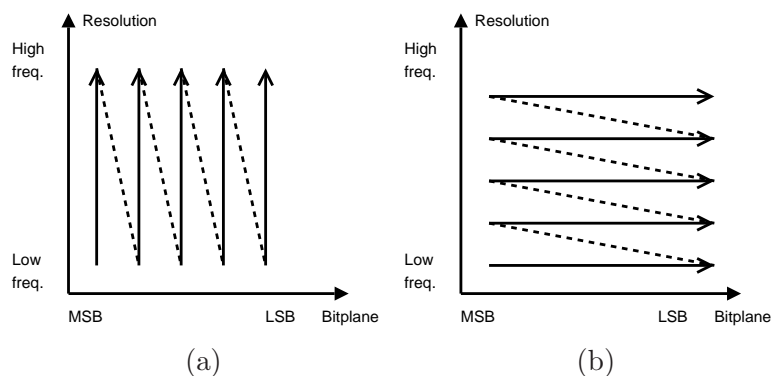


Fig. 4.10 Ordre de traitement selon la progression en qualité (a) ou en résolution (b). (a) correspond à la version SPIHT-RA de l'algorithme tandis que (b) correspond à SPIHT-RARS.

L'algorithme décrit ci-dessus possède une grande flexibilité et une même image peut être codée jusqu'à un niveau arbitraire de résolution ou jusqu'à un certain plan de bits selon les deux ordres possibles des boucles. Le décodeur peut aller jusqu'au même niveau pour décoder les images. Cependant, une propriété intéressante à avoir est de pouvoir coder l'image une seule fois

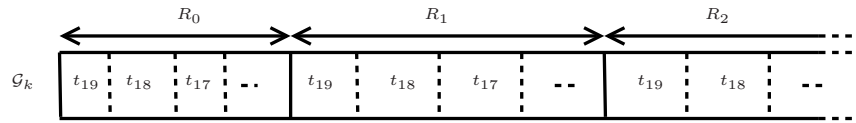


Fig. 4.11 Train binaire pour une progression en résolution. Ce train binaire correspond au codage d'un groupe. Les t_i correspondent aux plans de bits et les R_i aux résolutions.

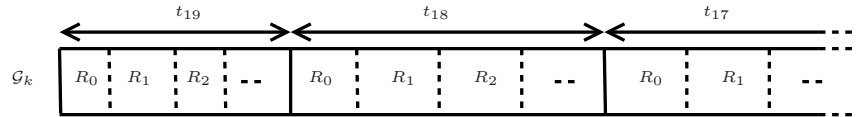


Fig. 4.12 Train binaire pour une progression en qualité. Ce train binaire correspond au codage d'un groupe. Les t_i correspondent aux plans de bits et les R_i aux résolutions.

avec toutes les résolutions et tous les plans de bits et de choisir seulement durant l'étape de décodage les résolutions et les plans de bits à décodé. Certaines applications peuvent avoir besoin d'une image basse résolution avec une forte précision radiométrique tandis que d'autres auront besoin d'une résolution forte et se contenteront d'une radiométrie moins précise.

Lorsque l'organisation progressive en résolution est utilisée (Fig. 4.11), il est facile d'interrompre le décodage à une certaine résolution, mais si tous les plans de bits ne sont pas nécessaires, on a besoin d'un moyen pour passer au début de la résolution 1 une fois que la résolution 0 est décodée jusqu'au plan de bits souhaité. Le problème est similaire avec l'organisation progressive en qualité (Fig. 4.12) en échangeant les termes résolutions et plan de bits dans la description du problème.

Pour résoudre ce problème, on introduit un système d'en-tête de groupes qui décrit la taille de chaque portion du train binaire. La nouvelle structure est décrite sur les figures 4.13 et 4.14. Le coût de cet en-tête est négligeable : la taille de chaque portion (en bits) est codée sur 24 bits¹. Éventuellement, comme les plus faibles résolutions (respectivement les plus hauts plans de bits) ne comportent en général que quelques bits, on peut décider de les décodé complètement (coût de décodage faible) et donc de ne pas utiliser d'en-têtes pour ces faibles résolutions. Uniquement la taille de grands morceaux (au dessus de 10000 bits en général) est alors sauvegardée. Cela correspond à environ 10 valeurs de 24 bits à écrire dans le train binaire par groupe. Le coût de cet en-tête est conservé en dessous de 0.1%.

Comme dans [Dan03], on aurait pu utiliser de simples marqueurs pour identifier le début d'une nouvelle résolution ou d'un nouveau plan de bits. Les marqueurs ont l'avantage d'être plus courts qu'un en-tête codant la taille complète du groupe suivant. Cependant, les marqueurs rendent obligatoire la lecture complète du train binaire (pour trouver le marqueur), le décodeur ne

¹ce qui est suffisant pour coder la taille de morceaux jusqu'à 16 Mbits ($2^{24} = 16\ 777\ 216$).

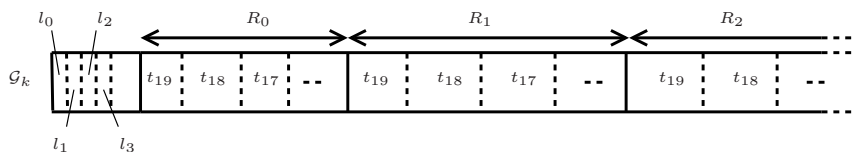


Fig. 4.13 Organisation progressive en résolution avec en-têtes. Les en-têtes permettent de passer directement à la résolution 1 lorsque le décodage de la résolution 0 arrive au plan de bits visé. l_i est la taille totale en bits de la résolution R_i .

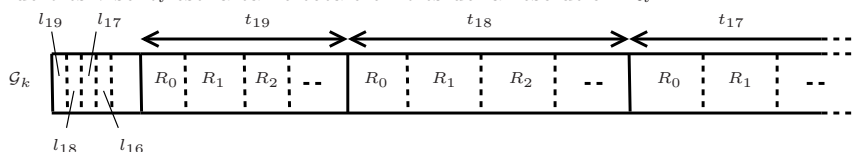


Fig. 4.14 Organisation progressive en qualité avec en-têtes. Les en-têtes permettent de passer directement à un plan de bits inférieur sans avoir à décoder toutes les résolutions pour le plan de bits courant. l_i est la taille totale en bits du plan de bits correspondant au seuil $t = i$.

peut pas se rendre directement à l'endroit désiré. Comme le coût de codage des en-têtes reste faible, cette solution est choisie.

4.4 Les désavantages du codage par groupes

Comme expliqué dans le paragraphe 4.2, le codage progressif est appliqué sur des groupes \mathcal{G}_k de l'image transformée. Ces groupes ont été définis dans le paragraphe 4.2 comme des groupements de $2 \times 2 \times 2$ coefficients de la sous-bande LLL ainsi que tous leurs descendants.

4.4.1 Conservation de la progression en qualité

Le problème de traiter les différentes parties d'une image séparément réside toujours dans l'allocation de débit pour chacune de ces parties. Un débit fixe pour chaque groupe n'est généralement pas une bonne décision car la complexité varie probablement à travers l'image. Si une progression en qualité est nécessaire pour l'image complète, il faut pouvoir mettre les bits les plus significatifs d'un groupe avant de finir le précédent. Cela peut être obtenu en coupant le train de binaire de chaque groupe pour les entrelacer. Avec cette solution, la progression en qualité ne sera plus disponible au niveau du bit à cause de l'organisation en groupes et à la séparation spatiale, mais un compromis avec des couches de qualité peut être trouvé (équivalent des *quality layers*), ce qui est présenté dans le paragraphe suivant. Cette organisation présente évidemment plus d'intérêt pour la version SPIHT-RA.

4.4.2 Organisation en couches et débit-distorsion

L'idée des couches de qualité est de fournir dans le même train binaire différents débits. Par exemple, un train binaire peut contenir deux niveaux de qualité : un correspondant à un débit de 1.0 bit par pixel (bpp) et un autre à 2.0 bpp. Si le décodeur a besoin d'une image à un débit de 1.0 bpp, uniquement le début du train binaire est transmis et décodé. Si une image de plus haute qualité à 2.0 bpp est nécessaire, la première couche est transmise, décodée et ensuite précisée grâce aux informations de la deuxième couche.

Comme le train binaire pour chaque groupe est progressif, on peut simplement choisir les points de coupure pour chaque groupe et chaque couche pour arriver au bon débit avec la qualité optimale pour toute l'image. On rappelle une fois encore que l'optimisation doit être globale et non locale car la qualité varie entre les groupes. Sous des contraintes spécifiques de mémoire, on peut optimiser localement.

Une méthode d'optimisation par lagrangien [Sho88] donne le point optimal de coupure pour chaque groupe \mathcal{G}_k . Comme le train binaire de chaque groupe est progressif, choisir un point de coupure différent conduit à un débit R_k différent et à une distorsion D_k différente. Comme les groupes sont codés indépendamment, leurs débits sont additifs et le débit final est $R = \sum R_k$. La mesure de distorsion doit être choisie comme additive pour avoir la distorsion finale $D = \sum D_k$. Une bonne mesure est l'erreur quadratique. Soit c un coefficient de la DWT originale de l'image et \tilde{c} le coefficient correspondant dans la reconstruction, alors

$$D_k = \sum_{c \in \mathcal{G}_k} (c - \tilde{c})^2 \quad (4.2)$$

L'optimisation par lagrangien [Sho88] nous dit qu'étant donné un paramètre λ , le point de coupure optimal pour chaque groupe \mathcal{G}_k est celui qui minimise la fonction de coût $J(\lambda) = D_k + \lambda R_k$. Ces points de coupures sont illustrés sur la figure 4.15. Pour chaque λ et chaque groupe \mathcal{G}_k , cette optimisation nous donne un point de fonctionnement optimal $(R_k^\lambda, D_k^\lambda)$. Le débit total pour un λ donné est $R^\lambda = \sum R_k^\lambda$ et la distorsion totale $D^\lambda = \sum D_k^\lambda$. En faisant varier le paramètre λ , un débit visé arbitraire peut être atteint.

Cette optimisation conduit à entrelacer les trains binaires pour les différents groupes. Après le codage de chaque groupe, on a besoin de conserver les données en mémoire pour pouvoir faire cette optimisation. Il peut paraître coûteux de devoir garder les données codées en mémoire, mais il doit être souligné que pour obtenir un train binaire progressif en qualité, il est nécessaire de conserver soit l'image complète, soit l'image complète codée en mémoire. Garder l'image codée est plus économique que de garder l'image d'origine. Cette contrainte n'est pas compatible avec un codage au fil de l'eau, dans ce cas, on sera obligé de réaliser une optimisation locale en traitant des groupes de lignes et d'utiliser un buffer pour *lisser* les différences

entre ces groupes.

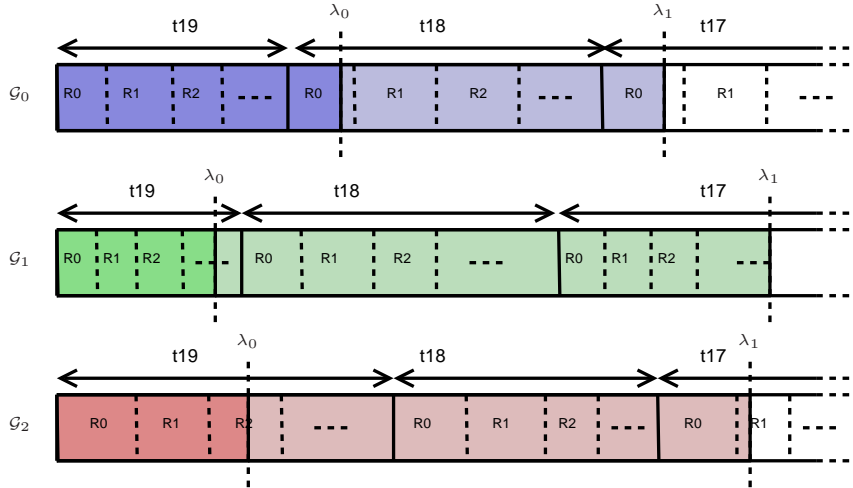


Fig. 4.15 Un train binaire progressif est généré pour chaque groupe \mathcal{G}_k . L’algorithme débit-distorsion sélectionne les différents points de coupure correspondant à différentes valeurs du paramètre λ . Le train binaire obtenu est illustré sur la figure 4.16.

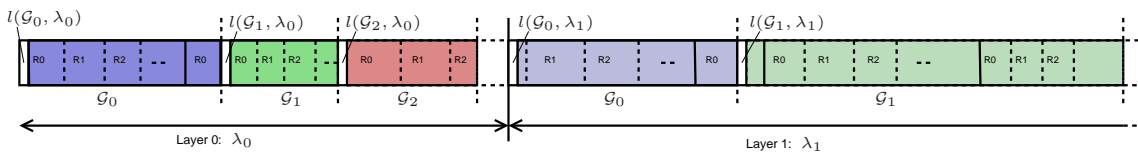


Fig. 4.16 Les trains binaires sont entrelacés selon différentes couches de qualité. Pour permettre un accès aléatoire aux différents groupes, la longueur en bit pour chaque morceau correspondant à un groupe \mathcal{G}_k et un niveau de qualité λ_q est donné par $l(\mathcal{G}_k, \lambda_q)$.

4.4.3 Connaître la distorsion pendant la compression

Dans la partie précédente, la connaissance de la distorsion pour chaque point de coupure pour chaque groupe était implicite. Comme le train binaire pour un groupe est en général de quelques millions de bits, il n’est pas concevable de garder l’information de distorsion pour chaque point de coupure en mémoire. Seulement quelques centaines de points de coupure potentiels sont mémorisés avec le débit et la distorsion correspondant.

Connaître le débit pour un point de coupure est la partie facile : il suffit de compter le nombre de bits précédant ce point. La distorsion demande plus de calcul. Cette distorsion n’est pas recalculée complètement pour chaque point, mais mise à jour durant la compression (*distorsion tracking*). On considère le moment où le compresseur va ajouter un bit de précision pour le coefficient

c dans le plan de bit t . Soit c_t la nouvelle approximation de c apportée par ce nouveau bit. Soit c_{t+1} l'approximation précédente (au plan de bits $t + 1$).

SPIHT utilise un quantificateur à zone morte donc si le bit de précision est 0 on a $c_t = c_{t+1} - 2^{t-1}$ et si ce bit est 1 on a $c_t = c_{t+1} + 2^{t-1}$. En notant D_{apres} la distorsion totale après que le bit ait été ajouté et D_{avant} la distorsion avant, on a :

– avec un bit de précision de 0 :

$$\begin{aligned} D_{apres} - D_{avant} &= (c - c_t)^2 - (c - c_{t+1})^2 \\ &= (c_{t+1} - c_t)(2c - c_t - c_{t+1}) \\ &= 2^{t-1} \left(2(c - c_{t+1}) + 2^{t-1} \right) \end{aligned} \quad (4.3)$$

ce qui donne

$$D_{apres} = D_{avant} + 2^{t-1} \left(2(c - c_{t+1}) + 2^{t-1} \right) \quad (4.4)$$

– avec un bit de précision de 1 :

$$D_{apres} = D_{avant} - 2^{t-1} \left(2(c - c_{t+1}) - 2^{t-1} \right) \quad (4.5)$$

Comme ce calcul peut être fait en utilisant uniquement des décalages et des additions, le coût calculatoire reste faible. D'autre part, l'algorithme n'a pas besoin de connaître la distorsion initiale comme on aurait pu s'y attendre. La méthode d'optimisation débit-distorsion reste valable si on remplace le terme *distorsion* par *réduction de distorsion* dans ce qui précède. Cette valeur de distorsion peut être très élevée et doit être conservée en interne sur un entier de 64 bits. Comme on l'a vu précédemment, on a 2^{18} coefficients dans un groupe, pour certains d'eux (dans la sous-bande LLL), leur valeur peut atteindre 2^{20} (avec le gain de la DWT). Les entiers sur 32 bits étant trop court, 64 bits semble un choix raisonnable et reste largement valable pour le pire des cas.

L'évaluation de la distorsion est faite dans le domaine transformé, directement sur les coefficients d'ondelettes. Cette évaluation est valide uniquement si la transformée est orthogonale. L'ondelette 9/7 de [Ant92] est quasi-orthogonale donc l'erreur réalisée par l'évaluation de la distorsion dans le domaine transformé reste suffisamment faible.

4.4.4 La formation du train binaire final

En général, on cherche à spécifier un débit R pour un niveau de qualité plutôt que de donner un paramètre λ qui n'a pas de sens pratique. Pour obtenir un débit visé, il faut trouver la bonne valeur de λ qui nous permettra d'obtenir le débit total visé $R(\lambda) = R$.

Proposition 3 ([Sho88]). *Soit λ_1 et λ_2 deux paramètres de Lagrange tels que $\lambda_1 < \lambda_2$. Soit (R_1, D_1) la solution au problème $\min\{D + \lambda_1 R\}$ et (R_2, D_2) la solution de $\min\{D + \lambda_2 R\}$. Alors, on a $R_1 \geq R_2$.*

Preuve: (R_1, D_1) est la solution de $\min\{D + \lambda_1 R\}$ donc $D_1 + \lambda_1 R_1 \leq D_2 + \lambda_1 R_2$. On a de même $D_2 + \lambda_2 R_2 \leq D_1 + \lambda_2 R_1$. En ajoutant ces inégalités, on obtient

$$\begin{aligned}\lambda_1 R_1 + \lambda_2 R_2 &\leq \lambda_1 R_2 + \lambda_2 R_1 \\ (\lambda_1 - \lambda_2) R_1 &\leq (\lambda_1 - \lambda_2) R_2 \\ R_1 &\geq R_2\end{aligned}$$

■

En utilisant cette propriété, on peut définir un algorithme rapide pour trouver la valeur de λ qui nous donnera le débit visé. En partant d'une valeur de départ λ , le débit $R(\lambda)$ est calculé. Selon les valeurs relatives de $R(\lambda)$ et R , la valeur de λ est modifiée. Une recherche dichotomique est particulièrement efficace dans cette situation. Il est à noter que tous ces calculs pour l'entrelacement du train binaire final arrivent après la compression des groupes et n'implique que les points de coupure en mémoire. Cette recherche n'a pas besoin de refaire la compression ni même d'accéder aux données compressées. Une fois que le λ donnant le débit désiré est trouvé, on passe à l'étape suivante pour effectuer l'entrelacement des groupes pour obtenir le train binaire final (Fig. 4.16).

4.5 Résultats

4.5.1 Données

Les données proviennent du capteur aéroport AVIRIS du JPL/NASA. Les scènes sont au départ de $614 \times 512 \times 224$ pixels et ont été réduites à $512 \times 512 \times 224$ (en abandonnant les 102 derniers échantillons). Les données sont des données en luminance, correspondant à ce qui est vu par le capteur. Pour des comparaisons plus faciles avec les travaux existants, des images connues sont traitées : en particulier la scène 3 du vol f970620t01p02_r03 sur Moffett Field, mais aussi la scène 1 du vol f970403t01p02_r03 sur Jasper Ridge et la scène 1 du vol f970619t01p02_r02 sur le site de Cuprite (Annexe D). Des images médiales à résonance magnétique (MR) et tomographie (CT) sont également utilisées. Les détails des images sont donnés dans le tableau 4.1.

L'erreur est exprimée en terme de PSNR (2.11), RMSE (2.7) et Erreur maximum (2.13) (ou MAD). Pour les données AVIRIS, le PSNR est calculé pour une dynamique de 16 bits. Toutes les erreurs sont mesurées entre l'image originale et l'image comprimée-décomprimée. Les mesures choisies facilitent les comparaisons éventuelles avec des travaux ultérieurs.

Image	Type	Dynamique	Taille
moffett_sc3	Hyperspectral	16 bits	$512 \times 512 \times 224$
jasper_sc1	Hyperspectral	16 bits	$512 \times 512 \times 224$
cuprite_sc1	Hyperspectral	16 bits	$512 \times 512 \times 224$
CT_skull	CT	8 bits	$256 \times 256 \times 192$
CT_wrist	CT	8 bits	$256 \times 256 \times 176$
MR_sag_head	MR	8 bits	$256 \times 256 \times 56$
MR_ped_chest	MR	8 bits	$256 \times 256 \times 64$

Tab. 4.1 Données utilisées : les données hyperspectrales proviennent du capteur AVIRIS, les données médicales de scanners à résonance magnétique (MR) ou tomographie (CT).

	1.0 bpppb	0.5 bpppb
Original 3D-SPIHT	75.74 dB	69.97 dB
3D-SPIHT-RA	75.65 dB	69.57 dB

Tab. 4.2 Impact des modifications sur les performances en PSNR. L'impact reste faible pour des débits élevés.

4.5.2 Performances de compression

L'algorithme combinant la progression en résolution (*Resolution Scalability*) et le codage par groupes permettant notamment l'accès aléatoire (*Random Access*) est nommé 3D-SPIHT-RARS (RARS=*Random Access with Resolution Scalability*). Les performances brutes de compression de l'algorithme 3D-SPIHT-RARS sont comparées avec les méthodes donnant les meilleures performances actuelles sans prendre en compte les flexibilités ajoutées. Les résultats pour JPEG 2000 mis en référence ici sont obtenus avec la version 5.0 de Kakadu [Tau06] en utilisant les extensions de la partie 2 de la norme : une décorrélation intercomposante à base d'ondelettes qui équivaut à ce qui est fait par notre algorithme. Les valeurs de PSNR obtenues dans ce cas sont similaires à celles publiées dans [Ruc05]. Ces performances sont également confirmées en utilisant le VM 9.1. 3D-SPIHT-RARS n'a pas pour objectif de surpasser JPEG 2000 en terme de performances brutes. La comparaison est faite pour montrer que l'augmentation de flexibilité ne conduit pas à une dégradation prohibitive des performances. Il est à noter aussi que les résultats donnés ici pour 3D-SPIHT et 3D-SPIHT-RARS n'incluent pas de codeurs entropiques.

Dans un premier temps, les performances sont comparées en termes de PSNR avec la version originale de 3D-SPIHT dans le tableau 4.2. La légère baisse de performance observée s'explique par la séparation lors du codage des différentes sous-bandes : après troncature, des bits différents vont être conservés. On rappelle une fois encore que dans le cas du codage sans pertes, à l'exception des en-têtes, SPIHT et SPIHT-RARS donnent exactement les mêmes bits dans un ordre différent. L'impact de l'ajout des nouvelles propriétés reste faible pour des forts débits.

La complexité d'un algorithme de compression n'est pas une chose facile

Algorithme de compression	temps (s)
Kakadu v5.0	20
VM 9.1	600
3D-SPIHT-RARS	130

Tab. 4.3 Temps de compression sur l'image Moffett scène 3 dans un mode de compression quasi sans perte.

Image	JPEG 2000	SPIHT-RARS
CT_skull	2.93	2.21
CT_wrist	1.78	1.31
MR_sag_head	2.30	2.42
MR_ped_chest	2.00	1.96
moffett sc3	5.14	5.47
jasper sc1	5.54	5.83
cuprite sc1	5.28	5.62

Tab. 4.4 Débit (bpppb) en codage sans pertes.

à mesurer. Une manière de le faire est de mesurer le temps nécessaire pour compresser une image, mais une mesure de ce type prend aussi en compte la qualité de l'implémentation. La version de 3D-SPIHT-RARS utilisée ici est réalisée en langage C mais n'est pas complètement optimisée. Une optimisation est difficilement compatible avec la volonté de garder un programme souple pour tester de nouvelles approches. Ce programme doit donc plus être vu comme un démonstrateur et il reste beaucoup de possibilités d'améliorations (comme le VM par exemple). Les performances présentées dans le tableau 4.3 permettent cependant de vérifier que le temps de compression reste raisonnable pour une implémentation de démonstration. D'ailleurs, la comparaison avec l'implémentation de démonstration de JPEG 2000 (le VM 9.1) le confirme. La valeur donnée ici pour 3D-SPIHT-RARS inclue les 30 s nécessaires pour le calcul de la DWT avec QccPack [Fow06]. QccPack est une bibliothèque en C proposant, entre autres, la transformée en ondelettes.

Le tableau 4.4 compare les performances, en terme de débit, entre JPEG 2000 et SPIHT-RARS pour le mode sans pertes. Dans les deux cas, la même transformée en ondelette basée sur la 5/3 entière (voir annexe A) est appliquée avec le même nombre de décompositions. Les performances sont similaires pour les images MR, SPIHT-RARS présente de meilleures performances pour les images CT et JPEG 2000 pour les images hyperspectrales.

Les tableaux 4.5 à 4.7 comparent les performances dans le cas d'un codage avec pertes utilisant l'un ou l'autre des deux algorithmes. Ces résultats confirment que l'augmentation de flexibilité de l'algorithme 3D-SPIHT-RARS n'entraîne pas une chute significative des performances. On observe moins d'un dB de différence entre les deux algorithmes. Un codeur arithmétique non contextuel appliqué directement au train binaire issu de 3D-SPIHT-RARS réduit déjà cette différence à 0.4 dB (non montré ici).

Rate (bpppb)	2.0	1.0	0.5	0.1
Kakadu v5.0	89.01	82.74	77.63	67.27
3D-SPIHT-RA	88.18	81.95	76.60	66.39

Tab. 4.5 PSNR pour différents débits pour Moffett sc3.

Rate (bpppb)	2.0	1.0	0.5	0.1
Kakadu v5.0	2.32	4.78	8.61	28.39
3D-SPIHT-RA	2.56	5.24	9.69	31.42

Tab. 4.6 RMSE pour différents débits pour Moffett sc3.

Rate (bpppb)	2.0	1.0	0.5	0.1
Kakadu v5.0	24	66	157	1085
3D-SPIHT-RA	37	80	161	1020

Tab. 4.7 MAD (ou Emax) pour différents débits pour Moffett sc3.

4.5.3 Flexibilité

Différentes résolutions et différents niveaux de qualité peuvent être extraits d'un même train binaire. Le tableau 4.8 présente différents résultats sur la scène 3 de Moffett Field en faisant varier le nombre de plans de bits et de résolutions à décoder. Pour les résultats de ce tableau, le même niveau de résolution est choisi pour les directions spatiale et spectrale, mais ce n'est pas obligatoire comme il est illustré sur la figure 4.17. L'image de référence à basse résolution pour le calcul de la distorsion finale est la sous-bande basse fréquence de la décomposition en ondelettes jusqu'au niveau désiré. Cette erreur prend en compte la non-réversibilité de l'ondelette 9/7 utilisée ici.

Pour donner une valeur correcte de la radiance, les coefficients sont correctement mis à l'échelle : compensation du gain dû aux filtres de la DWT (en fonction du niveau de résolution). L'ondelette 9/7 présente un gain de $\sqrt{2}$ dans les basses fréquences. Par exemple, lorsqu'on reconstitue l'image en abandonnant les plus hautes résolutions en spatial et en spectral, il faut compenser le gain de la DWT selon les trois directions en divisant les coefficients par $2\sqrt{2}$ pour obtenir une radiance correcte.

Le tableau 4.8 illustre la flexibilité de l'algorithme proposé. A partir du même train binaire, on peut choisir une qualité et une résolution maximales (nombre de plans de bits non décodés=0, résolution = full) ce qui conduit bien à la distorsion minimale mais également au débit maximum (5.309 bpppb) et au temps de décodage le plus important (59 s environ). À l'extrême, pour la qualité et la résolution minimales proposées ici (nombre de plans de bits non décodés=10, résolution = 1/8) un débit de 0.007 bpppb est atteint (une grande partie de la réduction provenant de la réduction de la résolution) en 3.16 s pour une qualité par rapport à l'image à basse résolution qui reste raisonnable. Entre ces extrêmes (et même au delà), l'utilisateur a

le choix parmi toutes les résolutions et les qualités possibles.

Sur la figure 4.17, on peut voir différents cubes hyperspectraux décodés du même train binaire avec différentes résolutions spatiale et spectrale. La face avant du cube est une composition colorée de différentes bandes spectrales. Les bandes sont choisies pour que la composition corresponde à celle du cube original. De légères différences peuvent être observées car les bandes spectrales des cubes à faible résolution proviennent d'une moyenne pondérée entre des bandes contigües (par la DWT). Par exemple, la composition colorée originale est obtenue en prenant la bande 90 pour le rouge, la bande 40 pour le vert et la bande 20 pour le bleu. Dans le cube sous-resolu d'un facteur 4 en spectral, on prendra alors les bandes 22 pour le rouge, 10 pour le vert et 5 pour le bleu (sur le total de 56 bandes), la bande 22 correspondant à une moyenne pondérée de la bande originale 90 avec ses voisines. . .

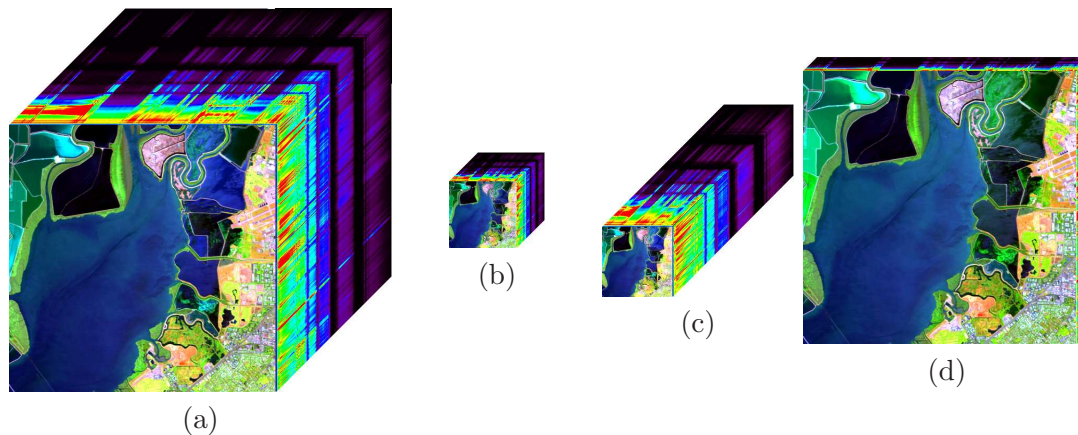


Fig. 4.17 Exemple de cube hyperspectral (moffett par AVIRIS) avec différentes résolution spatiale et spectrale à partir du même train binaire. (a) est l'image originale. (b) est obtenue en décodant l'image à 1/4 de la résolution spatiale et 1/4 de la résolution spectrale. (c) est à pleine résolution spectrale et 1/4 de résolution spatiale. (d) est à pleine résolution spatiale et 1/8 en résolution spectrale.

4.5.4 Codage de régions d'intérêt

L'intérêt principal de l'algorithme présenté ici est sa flexibilité. Le train binaire obtenu peut être décodé avec une résolution différente en spectral et en spatial pour chaque groupe. Cela est fait en lisant, ou en transmettant, un nombre minimum de bits. Chaque zone de l'image peut être décodée jusqu'à un niveau choisi de résolution spatiale, de résolution spectrale et de plans de bits. Cette propriété est illustrée sur la figure 4.18. La plus grande partie du fond de l'image (zone 1) est décodée à faible résolution spatiale et spectrale, réduisant de façon très importante la quantité de bits. Des parties spécifiques de l'image sont décodées de manière plus détaillée et ont, soit une

Nombre de plans de bits non décodés : 0				
Résolution	Full	1/2	1/4	1/8
bits lus	311 746 175	92 139 803	14 518 394	2 205 947
débit lu (bpppb)	5.309	1.569	0.247	0.038
PSNR (dB)	106.57	105.58	108.27	114.77
RMSE	0.31	0.34	0.25	0.12
Temps (s)	59.43	21.82	7.17	3.54
Nombre de plans de bits non décodés : 2				
Résolution	Full	1/2	1/4	1/8
bits lus	167 789 950	58 098 983	11 632 839	1 952 778
débit lu (bpppb)	2.857	0.989	0.198	0.033
PSNR (dB)	91.89	99.45	104.43	109.54
RMSE	1.67	0.70	0.39	0.22
Temps (s)	42.33	18.03	6.86	3.62
Nombre de plans de bits non décodés : 4				
Résolution	Full	1/2	1/4	1/8
bits lus	59 635 559	27 880 219	7 762 184	1 580 672
débit lu (bpppb)	1.016	0.475	0.132	0.027
PSNR (dB)	82.03	90.16	97.99	103.52
RMSE	5.18	2.03	0.82	0.44
Temps (s)	18.18	10.05	5.34	3.45
Nombre de plans de bits non décodés : 6				
Résolution	Full	1/2	1/4	1/8
bits lus	19 194 147	11 923 839	4 614 163	1 187 735
débit lu (bpppb)	0.327	0.203	0.079	0.020
PSNR (dB)	74.02	80.11	87.61	95.68
RMSE	13.05	6.47	2.73	1.08
Temps (s)	7.70	6.14	4.40	3.36
Nombre de plans de bits non décodés : 8				
Résolution	Full	1/2	1/4	1/8
bits lus	6 105 213	4 532 855	2 280 970	780 788
débit lu (bpppb)	0.104	0.077	0.039	0.013
PSNR (dB)	66.72	70.77	76.74	84.34
RMSE	30.23	18.97	9.53	3.98
Temps (s)	4.51	4.26	3.68	3.26
Nombre de plans de bits non décodés : 10				
Résolution	Full	1/2	1/4	1/8
bits lus	1 766 805	1 453 714	912 844	426 963
débit lu (bpppb)	0.030	0.025	0.016	0.007
PSNR (dB)	59.50	62.39	66.81	73.04
RMSE	69.41	49.76	29.92	14.60
Temps (s)	3.47	3.45	3.29	3.16

Tab. 4.8 Bits lus, équivalent en débit, PSNR, RMSE et temps de décodage pour différentes résolutions et qualité pour l'image *Moffett sc3*. La compression n'est faite qu'une seule fois. La résolution est la même pour les directions spatiales et spectrale mais elle pourrait être différente comme on peut le voir sur la figure 4.17.

résolution spectrale complète (zone 2), soit une résolution spatiale complète (zone 3) ou les deux (zone 4). L'image de la figure 4.18 est obtenue en ne lisant que 16 907 kbits du train binaire original de 311 598 kbits.

Les régions d'intérêt peuvent aussi être sélectionnées durant le codage en ajustant le nombre de plans de bits à coder pour chaque groupe. Dans un contexte de compression bord, cela pourrait permettre de réduire le débit de manière plus importante. On peut par exemple imaginer un module de détection de nuages qui ajusterait les paramètres du compresseur pour réduire la résolution et/ou la précision sur ces zones pour réduire le débit de ces groupes.

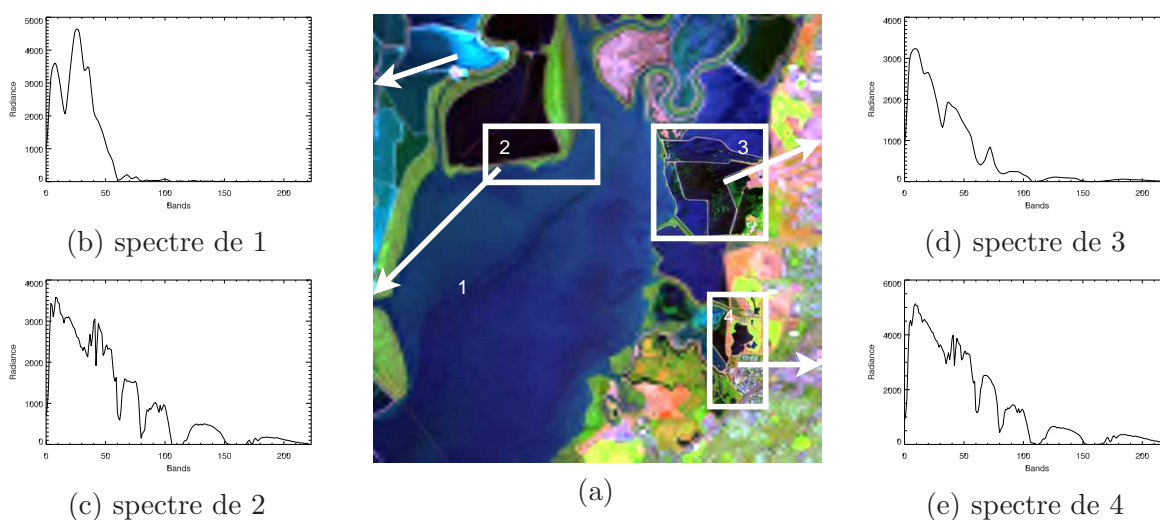


Fig. 4.18 Exemple d'image décompressée avec différentes résolutions spectrale et spatiale pour différentes parties de l'image. Le fond (zone 1) a une faible résolution spatiale et spectrale comme on peut le voir sur son spectre (b). La zone 2 a une faible résolution spatiale et une forte résolution spectrale (c). La zone 3 a une forte résolution spatiale, mais une faible résolution spectrale. Et finalement, la zone 4 a une forte résolution spatiale et spectrale. Cette image décompressée a été obtenue à partir d'un train binaire contenant toutes les informations en lisant un minimum de bits.

4.6 Conclusions

Une adaptation de l'algorithme 3D-SPIHT permettant un codage progressif en résolution ainsi qu'un accès aléatoire a été définie. Le codage progressif en résolution est déterminé de manière indépendante pour les directions spatiales et spectrales. Le codage par groupes permet l'accès aléatoire ainsi que le codage de régions d'intérêt tout en réduisant la quantité de mémoire nécessaire pour le codage. Grâce à l'optimisation débit-distorsion, cela est fait sans réduire les performances de compression.

Cet algorithme va dans le sens des tendances actuelles pour les algorithmes de compression : *Compress once, decompress many ways*. Le même train binaire peut servir à une large diversité d'utilisateurs avec des besoins différents. Cet algorithme permet également la distribution d'images par réseaux, on peut penser notamment à des applications de type Google Earth ou Geoportail de l'IGN : uniquement la partie utile de l'image est transmise et décompressée.

Chapitre 5

Retour sur les critères qualité

5.1 Représentation des critères qualité	132
5.1.1 Représentation	132
5.1.2 Robustesse par rapport à l'amplitude de la dégradation	132
5.1.3 Robustesse par rapport à l'image utilisée.	132
5.2 Validation	135
5.2.1 Une distance entre dégradations	135
5.2.2 Variation de niveau de dégradation (1er cas)	136
5.2.3 Dégradation inconnue (2ème cas)	137
5.2.4 Image différente (3ème cas)	137
5.2.5 Intérêt par rapport au PSNR	138
5.3 Impact de SPIHT-RA.	140
5.4 Aller plus loin	140



NOUS allons maintenant évaluer l'impact du système de compression nouvellement défini sur la qualité des images à partir des critères sélectionnés dans le chapitre 2. Une représentation combinant les cinq critères est d'abord exposée, son intérêt graphique est montré. Ensuite une méthode numérique est définie pour produire une identification plus rigoureuse des dégradations ainsi que pour démontrer l'avantage de cette méthode par rapport à une mesure de l'erreur classique par PSNR. Enfin la méthode est appliquée au nouvel algorithme de compression pour qualifier ses dégradations.

5.1 Représentation des critères qualité

5.1.1 Représentation

Il est délicat de trouver une représentation efficace pour visualiser la complémentarité des cinq critères sélectionnés précédemment. Un bon moyen est d'utiliser un diagramme en étoile (Fig 5.1). Les cinq axes du diagramme correspondent aux cinq critères qualité. L'échelle sur toutes les figures suivantes est la même. Pour le MAD, le MAE et le RRMSE, l'origine représente la valeur 0 (aucune dégradation). L'extrémité correspond à 5000 pour le MAD, 40 pour le MAE et 15 pour le RRMSE. Pour F_λ et $Q_{(x,y)}$ l'origine est la valeur 1 (aucune dégradation), le minimum étant 0.9 pour F_λ et 0.6 pour $Q_{(x,y)}$. La forme du diagramme est caractéristique du type de dégradation.

5.1.2 Robustesse par rapport à l'amplitude de la dégradation

Cette représentation est robuste concernant l'amplitude de la dégradation. La forme reste la même pour une dégradation donnée, il y a uniquement une différence dans le facteur d'échelle (Fig. 5.1 à 5.6). On peut également observer que les combinaisons sont possibles : le diagramme correspondant à la combinaison de flou spatial et spectral (Fig. 5.4) est logiquement la combinaison de celui du flou spatial (Fig. 5.2) et de celui du flou spectral (Fig. 5.3).

5.1.3 Robustesse par rapport à l'image utilisée

Il faut également vérifier si la représentation est robuste par rapport à l'image traitée. Cependant, certaines dégradations sont elle-même sensibles à l'image. Il est aisé de comprendre qu'un flou spatial aura une influence plus forte sur une image contenant de la ville que sur une image plutôt uniforme. La représentation est robuste pour la plupart des dégradations comme on peut le voir pour les images *moffett3*, *moffett4* et *harvard1* (Fig. 5.7 et 5.8).

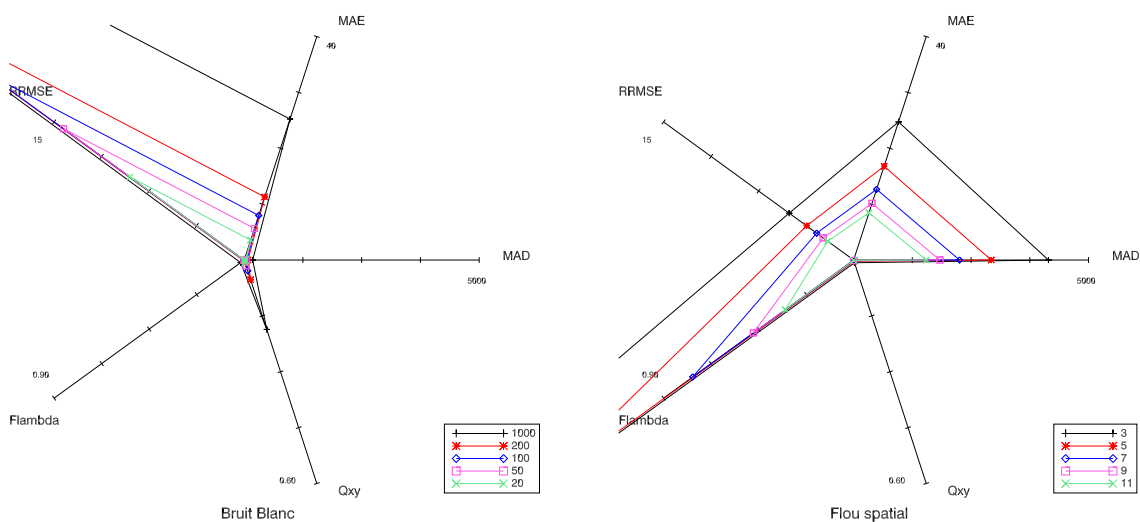


Fig. 5.1 Critères qualité pour différentes valeurs de bruit blanc sur l'image *moffett4*.

Fig. 5.2 Critères qualité pour différentes valeurs de flou spatial sur l'image *moffett4*.

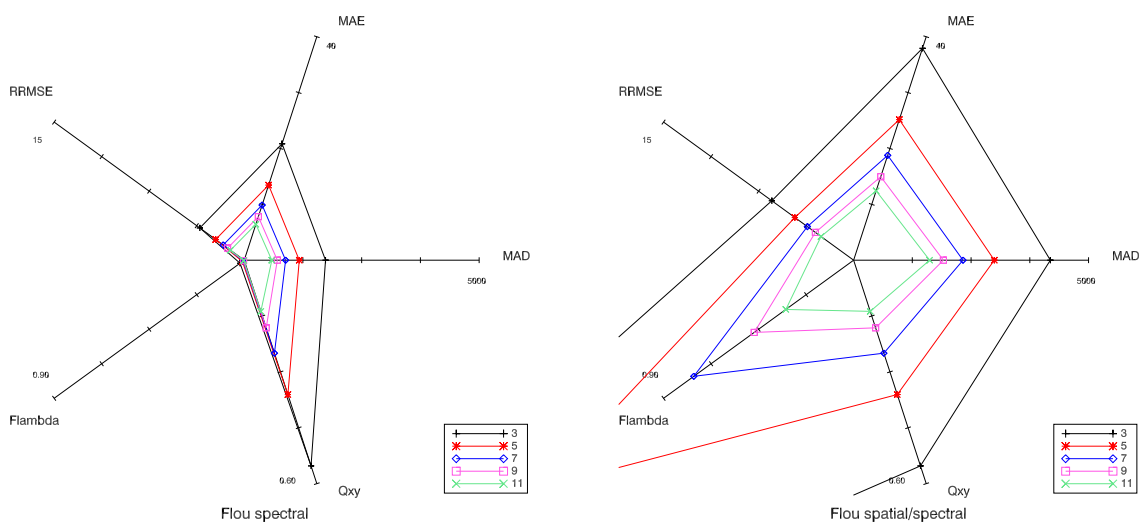


Fig. 5.3 Critères qualité pour différentes valeurs de flou spectral sur l'image *moffett4*.

Fig. 5.4 Critères qualité pour différentes valeurs de flou spatial et spectral sur l'image *moffett4*.

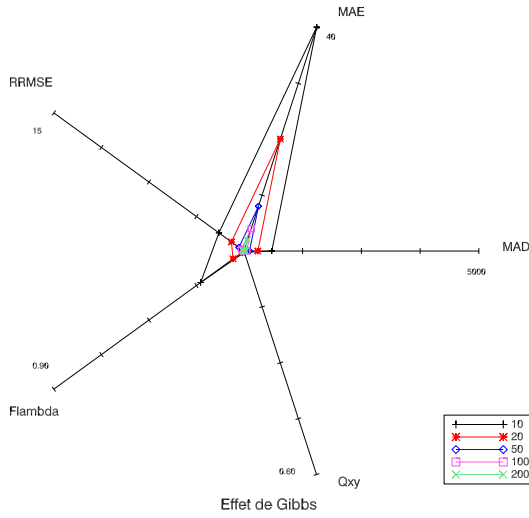


Fig. 5.5 Critères qualité pour différentes amplitudes d'effet de Gibbs sur l'image *moffett4*.

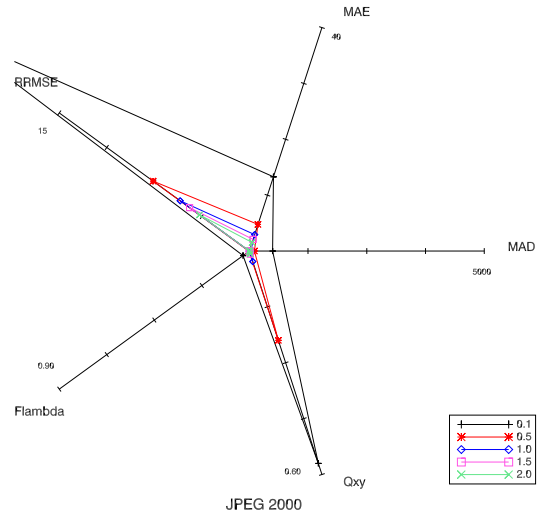


Fig. 5.6 Critères qualité pour différents débits avec JPEG 2000 sur l'image *moffett4*.

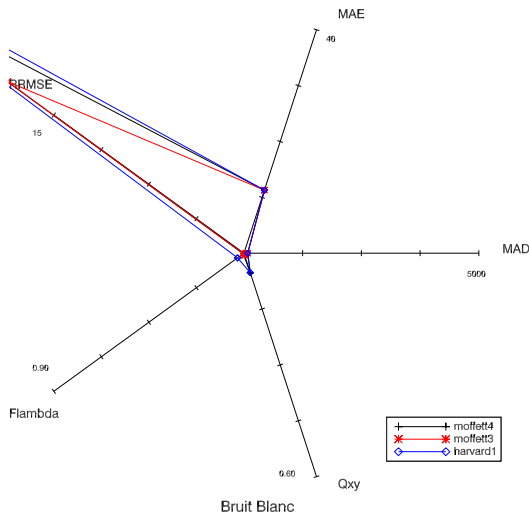


Fig. 5.7 Critères qualité pour le bruit blanc sur différentes images.

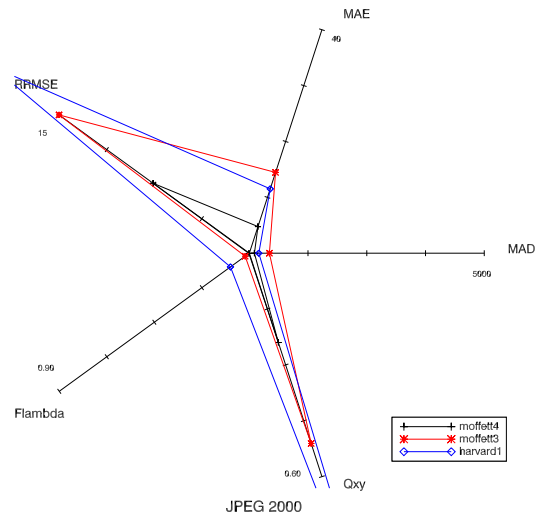


Fig. 5.8 Critères qualité pour JPEG 2000 sur différentes images.

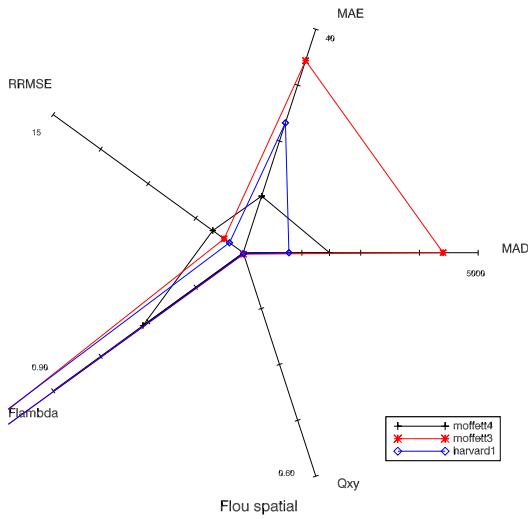


Fig. 5.9 Critères qualité pour le flou spatial sur différentes images.

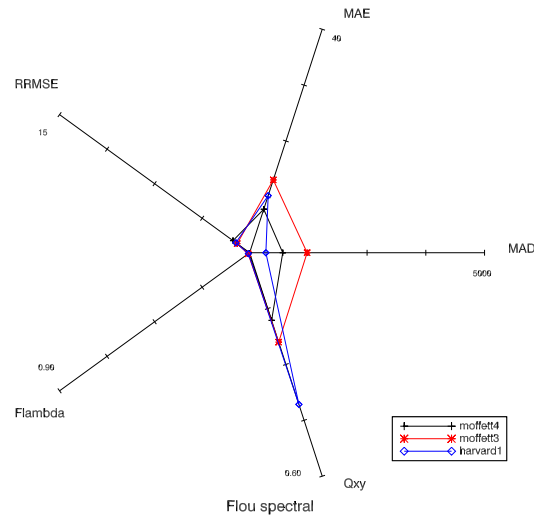


Fig. 5.10 Critères qualité pour le flou spectral sur différentes images.

Le cas des dégradations du flou est plus complexe (Fig. 5.9 et 5.10). L'amplitude de la dégradation dépend non seulement du filtre utilisé pour effectuer le flou, mais également de l'image. Si l'image contient plus de hautes fréquences, l'impact du flou va être plus important, il est donc difficile de considérer que la dégradation a la même intensité.

5.2 Validation

5.2.1 Une distance entre dégradations

On voit donc que la combinaison de ces critères de qualité fournit une bonne caractérisation de la dégradation. La situation idéale à atteindre, le *Graal* du critère qualité, serait de pouvoir prédire précisément l'impact de la dégradation sur l'application en connaissant uniquement les 5 valeurs des critères de qualité ainsi qu'une correspondance pour une certaine image et pour certaines dégradations entre valeurs des critères de qualité et dégradations sur l'application.

Il est possible de définir un protocole de validation à différents niveaux (de difficulté croissante) pour cette combinaison de critères de qualité. On considère qu'on dispose des valeurs des cinq critères qualité pour une image avec différentes dégradations et différents niveaux pour chaque dégradation (figures 5.1 à 5.6). On dispose également de la correspondance entre chacune de ces dégradations et l'impact sur une application donnée. On peut donc essayer de prédire l'impact :

- d’une dégradation connue, mais avec un niveau différent sur cette image (1er cas) ;
- d’une dégradation inconnue sur cette image (2ème cas) ;
- d’une dégradation sur une image différente (3ème cas).

On se sert dans la suite des résultats obtenus sur l’image *moffett3*. Une seule image est utilisée comme référence car il est difficile d’avoir des résultats pour plusieurs applications sur une même image (c’est une des raisons qui fait l’intérêt des critères qualité). Dans le cas où une autre image est nécessaire, l’image *moffett4* est utilisée.

Lorsqu’on observe une situation inconnue, on va chercher le diagramme connu le plus proche. Pour cette notion de proximité, il nous faut définir une distance. Une distance euclidienne classique dans un espace à 5 dimensions (les 5 critères) est la solution la plus intuitive. Il reste cependant le problème de l’échelle entre les différents critères : le domaine de variation du MAD qui atteint facilement des valeurs de 2000 n’a rien à voir avec le domaine de variation de F_λ qui reste entre 0.9 et 1. Il n’y a pas de solution idéale pour ce problème, on décide donc de se baser arbitrairement sur les échelles utilisées sur les diagrammes précédents qui permettent de bien discriminer les différents critères à l’œil. On note par $\widetilde{\cdot}$ cette valeur normalisée.

La distance entre deux diagrammes est donc définie par :

$$d = \sqrt{\widetilde{\text{MAD}}^2 + \widetilde{\text{MAE}}^2 + \widetilde{\text{RRMSE}}^2 + \widetilde{F_\lambda}^2 + \widetilde{Q_{(x,y)}}^2}. \quad (5.1)$$

Plus cette distance est faible plus les dégradations pourront être considérées comme semblables. Nous allons maintenant vérifier si cette distance donne des résultats performants dans les trois situations illustrées ci-dessous.

5.2.2 Variation de niveau de dégradation (1er cas)

Dans le cas où un bruit blanc avec une variance de 150 est appliquée à l’image, les distances entre ce diagramme et les diagrammes de référence les plus faibles sont sans surprise les celles concernant le bruit blanc d’une variance de 200 ainsi que le bruit blanc d’une variance de 100 (Tab. 5.1). On peut donc dans ce cas correctement prédire l’impact de la dégradation sur l’application. On déduira un nombre de pixels mal classifiés entre 163 et 255, ce qui est correct : le nombre de pixels mal classifiés étant 222 (sur 65536 pixels au total).

Ce cas est également illustré pour un flou spectral d’atténuation 4 appliqué à l’image (Tab. 5.2). Les distances les plus faibles sont celles avec les flous spectraux d’atténuation 3 et 5. Cela nous donne un nombre de pixels mal classifiés entre 166 et 262, la valeur réelle étant 207.

Type de dégrad.	Paramètre dégrad.	Distance	# pixels mal class. (SAM)
Bruit blanc	50	0.466217	112
Bruit blanc	100	0.202422	163
Bruit blanc	200	0.170655	255
Bruit blanc	1000	1.74537	634
Flou spectral	1	2.80421	593
Flou spectral	3	1.78054	262
Flou spectral	5	1.32591	166
Flou spectral	7	1.17346	123
Flou spatial	9	2.74443	5710
Flou spatial	11	2.09802	4871
Flou spatial	13	1.72581	4248
Flou spatial	15	1.50135	3778
Flou mixe	11	2.12895	4881
Gibbs	50	1.06826	698
Gibbs	100	1.07466	425
Gibbs	500	1.09681	70
JPEG 2000	0.1	4.08466	4376
JPEG 2000	0.5	0.859046	450
JPEG 2000	1.0	0.732228	142

Tab. 5.1 Distances pour un bruit blanc de paramètre 150 : la distance identifie bien le type de dégradation, les points les plus proches sont les bruits blancs de variance 100 et 200.

Type de dégrad.	Paramètre dégrad.	Distance	# pixels mal class. (SAM)
Bruit blanc	50	1.31986	112
Bruit blanc	100	1.40968	163
Bruit blanc	200	1.60283	255
Bruit blanc	1000	2.83994	634
Flou spectral	1	1.62052	593
Flou spectral	3	0.365524	262
Flou spectral	5	0.271982	166
Flou spectral	7	0.567515	123
Flou spatial	9	2.55066	5710
Flou spatial	11	1.90902	4871
Flou spatial	13	1.55253	4248
Flou spatial	15	1.35135	3778
Flou mixe	11	1.76783	4881
Gibbs	50	1.18216	698
Gibbs	100	1.22852	425
Gibbs	500	1.29586	70
JPEG 2000	0.1	4.20082	4376
JPEG 2000	0.5	1.01159	450
JPEG 2000	1.0	0.931696	142

Tab. 5.2 Distances pour un flou spectral de paramètre 4 : la distance identifie bien le type de dégradation, les points les plus proches sont les flous spectraux d'atténuation 3 et 5.

5.2.3 Dégradation inconnue (2ème cas)

Pour traiter ce cas, on retire les dégradations avec JPEG 2000 de notre ensemble de situations connues, on la traite en dégradation inconnue et on regarde quelle est la situation la plus proche pour essayer de prédire le nombre de pixels mal classifiés.

Les distances obtenues sont présentées dans le tableau 5.3. La dégradation causée par JPEG 2000 est identifiée comme un mélange de bruit blanc de variance 50 et de flou spectral d'atténuation 7. Cette identification correspond à l'intuition qu'on pouvait avoir en observant les diagrammes en étoile. Les valeurs de pixels mal classifiés prédites sont de l'ordre de 112 ou 123, la valeur réelle étant 142.

5.2.4 Image différente (3ème cas)

Dans ce cas, on utilise les résultats obtenus sur *moffett3* pour déduire les dégradations probables sur *moffett4*. Dans le cas d'un bruit blanc avec une variance de 100, la distance utilisée identifie correctement la dégradation comme un bruit blanc (Tab. 5.4). Cette distance interprète le bruit blanc de variance 100 sur *moffett4* comme ayant le même effet qu'un bruit blanc de variance 20 sur *moffett3*. Le nombre de pixels mal classifiés prédit par cette méthode est de 71 alors qu'en réalité la valeur est 91.

On applique également la méthode pour une compression JPEG 2000 sur *moffett4* pour un débit de 0.5 bpppb. Les distances obtenues sont présentées

Type de dégradation	Paramètre dégradation	Distance	Nombre pixels mal classifiés (SAM)
Bruit blanc	50	0.501280	112
Bruit blanc	100	0.599453	163
Bruit blanc	200	0.865206	255
Bruit blanc	1000	2.33358	634
Flou spectral	1	2.43324	593
Flou spectral	3	1.24618	262
Flou spectral	5	0.733441	166
Flou spectral	7	0.578098	123
Flou spatial	9	2.65673	5710
Flou spatial	11	1.97538	4871
Flou spatial	13	1.56823	4248
Flou spatial	15	1.31138	3778
Flou mixe	11	1.94497	4881
Gibbs	50	0.723429	698
Gibbs	100	0.722423	425
Gibbs	500	0.742236	70

Tab. 5.3 Distances pour JPEG 2000 à 1 bpppb (sans inclure JPEG 2000 dans les situations connues) : les distances les plus faibles correspondent à un bruit blanc de paramètre 50 ou un flou spectral de paramètre 7.

dans le tableau 5.5. La distance identifie bien la dégradation comme étant une compression JPEG 2000. L'image *moffett4*, plus uniforme, est plus facile à compresser. Cette propriété explique que les dégradations sur *moffett4* à 0.5 bpppb sont semblables à *moffett3* à 1.0 bpppb. Le nombre de pixels mal classifiés estimés par cette méthode est 142, la valeur réelle étant 82. Il y a donc un manque de fiabilité dans ce cas.

On voit donc que dans le cas où la méthode est appliquée à une image différente, il est difficile d'évaluer précisément l'impact de la dégradation sur l'application. Toutefois la méthode permet d'identifier avec succès le type de dégradation subie par l'image. Il serait intéressant de préciser ces résultats sur un nombre d'images et d'applications beaucoup plus important.

5.2.5 Intérêt par rapport au PSNR

Le PSNR, ainsi que les MSE ou SNR, ne permettent pas de rendre compte de la nature de la dégradation et donc ne permet pas de prévoir l'impact sur les applications. Ce fait est illustré sur la figure 5.11. Les diagrammes sont tracés pour différentes dégradations donnant le même PSNR. Les 5 types de dégradations sont appliquées à l'image *moffett3* de telle manière que le PSNR soit égal dans tous les cas à 63 dB. On voit bien sur le diagramme que bien que ces dégradations donnent le même PSNR, les caractéristiques de ces dégradations sont très différentes.

Ainsi, pour une image possédant un PSNR de 63 dB, on ne pourra pas différencier entre

Type de dégrad.	Paramètre dégrad.	Distance	# pixels mal class. (SAM)	Type de dégrad.	Paramètre dégrad.	Distance	# pixels mal class. (SAM)
Bruit blanc	20	0.0566873	71	Bruit blanc	50	0.414287	112
Bruit blanc	50	0.269541	112	Bruit blanc	100	0.548444	163
Bruit blanc	100	0.531203	163	Bruit blanc	200	0.848581	255
Bruit blanc	200	0.903320	255	Bruit blanc	1000	2.35509	634
Flou spectral	1	2.80002	593	Flou spectral	1	2.53412	593
Flou spectral	3	1.59900	262	Flou spectral	3	1.33787	262
Flou spectral	5	0.990605	166	Flou spectral	5	0.795692	166
Flou spectral	7	0.722567	123	Flou spectral	7	0.603463	123
Flou spatial	9	2.62982	5710	Flou spatial	9	2.65744	5710
Flou spatial	11	1.92666	4871	Flou spatial	11	1.97040	4871
Flou spatial	13	1.49509	4248	Flou spatial	13	1.55665	4248
Flou spatial	15	1.21260	3778	Flou spatial	15	1.29278	3778
Flou mixe	11	1.97445	4881	Flou mixe	11	1.95845	4881
Gibbs	50	0.405655	698	Gibbs	50	0.654444	698
Gibbs	100	0.371836	425	Gibbs	100	0.642532	425
Gibbs	500	0.369275	70	Gibbs	500	0.651480	70
JPEG 2000	0.1	4.63819	437	JPEG 2000	0.1	4.33610	4376
JPEG 2000	0.5	1.06898	450	JPEG 2000	0.5	0.715630	450
JPEG 2000	1.0	0.489916	142	JPEG 2000	1.0	0.117513	142

Tab. 5.4 Distances pour un bruit blanc de variance 100 sur l'image *moffett4* par rapport aux dégradations sur *moffett3* : la dégradation est correctement identifiée comme un bruit blanc. La prédiction en terme de pixels mal classifiés est bien la plus proche (71 au lieu de 91 en réalité).

Tab. 5.5 Distances pour une compression JPEG 2000 à débit de 0.5 bpppb sur l'image *moffett4* par rapport aux dégradations sur *moffett3*. La dégradation est correctement identifiée comme étant une compression JPEG 2000. La prédiction en nombre de pixels mal classifiés est assez proche (142 au lieu de 82 en réalité).

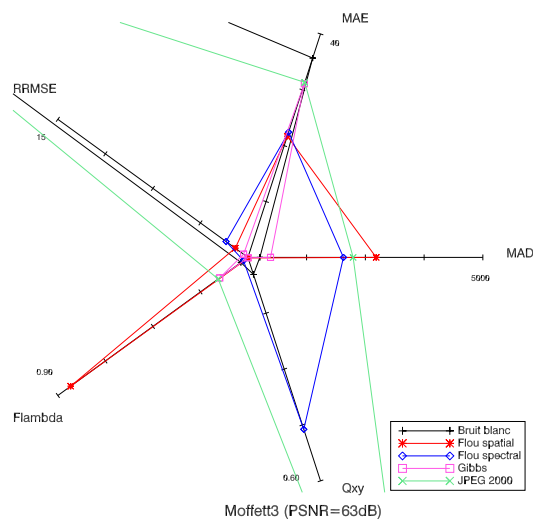


Fig. 5.11 Diagramme pour les 5 types de dégradations à PSNR constant sur l'image *moffett3* (63 dB).

- un bruit blanc de variance 2000 : 1054 pixels mal classifiés ;
- un flou spectral d'atténuation 4 : 207 pixels mal classifiés ;
- un flou spatial d'atténuation 15 : 3778 pixels mal classifiés ;
- un phénomène de Gibbs de paramètre 20 : 2029 pixels mal classifiés ;
- une compression JPEG 2000 à 0.2 bpppb : 1772 pixels mal classifiés.

Ces dégradations, qui donnent toutes un PSNR d'environ 63 dB, ont un impact sur l'application qui va varier de 207 pixels mal classifiés à 3778 pixels mal classifiés. Le PSNR échoue donc à rendre compte de l'impact d'une dégradation sur la qualité d'une image hyperspectrale pour l'utilisateur.

5.3 Impact de SPIHT-RA

On peut réaliser une comparaison des diagrammes en étoile entre JPEG 2000 et SPIHT-RA (Fig. 5.12). La version progressive en qualité de l'algorithme (SPIHT-RA) convient mieux pour des comparaisons avec JPEG 2000 (on raisonne en terme de débit visé), c'est donc cette version qui est utilisée ici. Les diagrammes présentent de grandes similarités ce qui signifie que les deux algorithmes de compression vont causer des dégradations similaires. On voit qu'alors que JPEG 2000 est légèrement plus performant en terme de PSNR, ce n'est plus le cas pour l'ensemble des critères qualité sélectionnés. Notamment en terme de MAD (ou E_{max}), sur la figure 5.12 à droite, on voit que SPIHT-RA donne un MAD plus faible que JPEG 2000 à 0.2 ou 0.4 bpppb. Le calcul des distances entre JPEG 2000 et SPIHT-RA est également réalisé. Pour deux débits différents de 0.5 bpppb et 0.2 bpppb (Tab. 5.6 et 5.7), le diagramme le plus proche est à chaque fois un JPEG 2000. Il est intéressant de noter que la dégradation causée par la compression SPIHT-RA à 0.2 bpppb est plus proche de la compression JPEG 2000 à 0.3 bpppb que celle à 0.2 bpppb. SPIHT-RA semble donc plus performant à bas débit.

5.4 Aller plus loin

Les résultats présentés ici semblent prometteurs, tant du point de vue des performances de l'algorithme de compression défini que de la fiabilité des critères qualité. Cependant, ces résultats restent à confirmer pour une base de test plus grande : plus d'images et plus d'applications.

L'utilisation de la combinaison des cinq critères de qualité est faite ici simplement en les regroupant pour définir une distance. On a vu que cette méthode donne des résultats positifs. Il serait intéressant de raffiner cette méthode en intégrant des méthodes plus évoluées : utiliser les cinq critères comme une entrée à un réseau de neurones ayant pour sortie le nombre de pixels mal classifiés semble une chose naturelle à faire, les méthodes basées

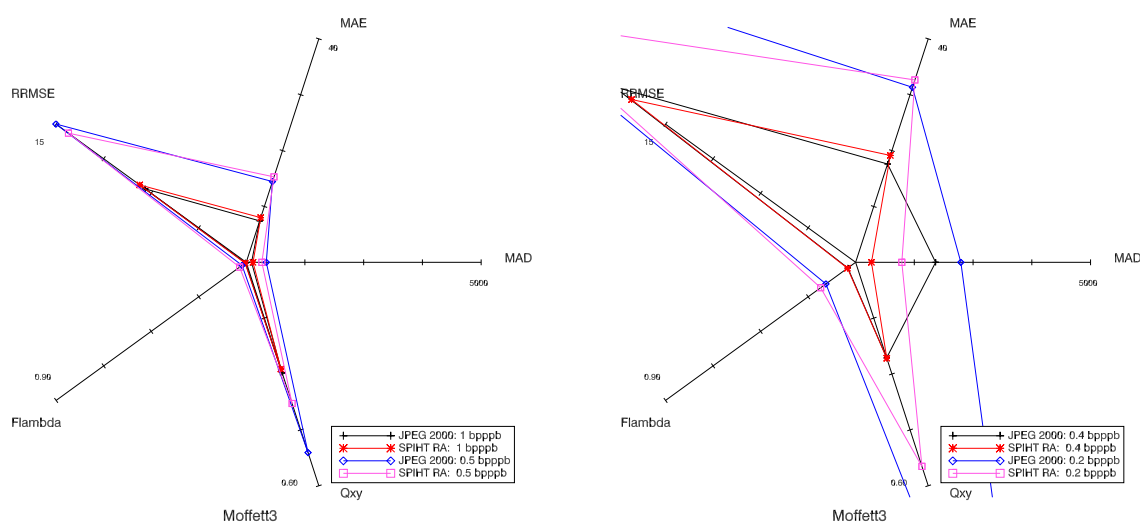


Fig. 5.12 Comparaison entre JPEG 2000 et SPIHT-RA.

Type de dégrad.	Paramètre dégrad.	Distance	# pixels mal class. (SAM)
Bruit blanc	50	0.746697	112
Bruit blanc	100	0.658787	163
Bruit blanc	200	0.703785	255
Bruit blanc	1000	1.95047	634
Flou spectral	1	2.25732	593
Flou spectral	3	1.23109	262
Flou spectral	5	0.915794	166
Flou spectral	7	0.890367	123
Flou spatial	9	2.69881	5710
Flou spatial	11	2.06138	4871
Flou spatial	13	1.70004	4248
Flou spatial	15	1.48704	3778
Flou mixe	11	1.99473	4881
Gibbs	50	1.12947	698
Gibbs	100	1.14863	425
Gibbs	500	1.18502	70
JPEG 2000	0.1	3.80149	4376
JPEG 2000	0.4	0.483630	623
JPEG 2000	0.5	0.231539	450
JPEG 2000	1.0	0.472943	142

Tab. 5.6 Distances pour une compression SPIHT-RA à un débit de 0.5 bpppb : le plus proche est le JPEG 2000 à 0.5 bpppb.

Type de dégrad.	Paramètre dégrad.	Distance	# pixels mal class. (SAM)
Bruit blanc	50	1.61881	112
Bruit blanc	100	1.42359	163
Bruit blanc	200	1.19242	255
Bruit blanc	1000	1.38219	634
Flou spectral	1	2.14931	593
Flou spectral	3	1.66292	262
Flou spectral	5	1.67821	166
Flou spectral	7	1.76372	123
Flou spatial	9	2.94032	5710
Flou spatial	11	2.44976	4871
Flou spatial	13	2.21407	4248
Flou spatial	15	2.09935	3778
Flou mixe	11	2.32759	4881
Gibbs	50	2.05463	698
Gibbs	100	2.08964	425
Gibbs	500	2.13918	70
JPEG 2000	0.2	1.76344	1772
JPEG 2000	0.3	0.329868	973
JPEG 2000	0.4	0.802000	623
JPEG 2000	1.0	1.45331	142

Tab. 5.7 Distances pour une compression SPIHT-RA à un débit de 0.2 bpppb : le plus proche est le JPEG 2000 à 0.3 bpppb.

sur les *Support Vector Machine* (SVM) peuvent également être appliquées.

Conclusion

CE document aborde le problème de la compression des images hyperspectrales de bout en bout :

Acquisition : étude du principe d'acquisition, des instruments.

Données : propriétés statistiques, mise en évidence d'une anisotropie : cette propriété importante des images hyperspectrales n'est pas trop exploitée dans la littérature.

Transformée : définition d'une transformée en ondelette quasi-optimale pour un bon compromis performances/complexité. On a montré dans un premier temps l'intérêt d'une transformée en ondelette optimale au sens d'un critère débit-distortion par rapport à une transformée classique isotropique. Le gain est de l'ordre de 8 dB en terme de PSNR par rapport à une décomposition isotropique classique, pour des débits compris entre 0.1 et 4 bits par pixel par bande (bpppb). Dans un deuxième temps, on a proposé une décomposition par ondelettes anisotropique fixe, quelle que soit l'image et le débit visé. Cette décomposition est ensuite qualifiée de *quasi-optimale* dans le sens où elle approche à moins d'un dB la décomposition optimale sur les courbes de PSNR fonction du débit.

Codage : structures d'arbres de zéros, adaptation des algorithmes EZW et SPIHT. Une fois la décomposition anisotropique quasi-optimale proposée, nous nous sommes intéressés au codage des coefficients d'ondelettes. Nous avons proposés des adaptations des algorithmes EZW et SPIHT en incluant une structure d'arbre basée sur la décomposition 3D des images. Sur l'ensemble des images, l'algorithme de compression proposé permet d'atteindre des PSNR de l'ordre de 70 dB pour des débits de 0.5 bpppb. L'utilisation de la notation binaire signée est également proposée pour renforcer l'attrait des arbres de zéros.

Utilisation : souci de produire des trains de bits flexibles pour une utilisation des données efficace. Nous avons ainsi développé une adaptation de l'algorithme permettant un codage progressif en résolution ainsi qu'un accès aléatoire, sans réduire les performances de compression. L'utilisateur peut ainsi décompresser les données hyperspectrales avec une résolution plus faible spectralement ou juste une portion de l'image avec des ressources limitées.



Qualité : définition et sélection de cinq critères de qualité complémentaires et définition d'une méthode originale pour les exploiter. En particulier, nous montrons que le PSNR n'est pas le bon critère pour prédire la qualité de l'application sur les images hyperspectrales après compression, en particulier dans le cadre d'une classification. Nous proposons un ensemble de critères qui apparaissent complémentaires et qui permettent de qualifier le type de dégradation appliquée à l'image et de donner un ordre d'idée des performances de l'application sur les images après compression (par exemple, le nombre de pixels qui seront mal classifiés).

Même si ce travail est consacré aux images hyperspectrales, la plupart des résultats restent valides pour d'autres types de données. En particulier, certaines images médicales, acquises *par tranches*, sont très semblables. La décomposition quasi-optimale fonctionne également très bien, l'algorithme de compression proposant accès aléatoire, codage progressif en qualité garde tout son intérêt.

Il reste bien entendu beaucoup de possibilités d'améliorations, d'idées nouvelles à essayer, de points à préciser... Un travail plus exhaustif est à faire sur les critères de qualité, en particulier sur la façon de les combiner et d'en exploiter les résultats. Le dernier chapitre de cette thèse a donné quelques pistes mais l'étude n'est pas complète.

Le codage des coefficients d'ondelettes proposé mériterait aussi qu'on s'intéresse un peu plus à l'implantation de SPIHT utilisant de véritables arbres 3D, nécessitant une réponse au problème du *tree-crossing* évoqué dans le chapitre 3.

Enfin, l'implémentation matérielle de l'algorithme de compression est à chiffrer et son optimisation à réaliser. Celle-ci est à envisager conjointement avec le processeur spatialisable du moment.

Annexe A

DWT

Il existe de nombreux tutoriaux concernant la théorie et l'utilisation des ondelettes. Cette annexe n'a pas la prétention de les remplacer, elle regroupe les informations utiles concernant les transformées en ondelettes discrètes (DWT) utilisées dans cette étude pour référence.

A.1 Implémentation générale

Deux approches principales existent pour la transformée en ondelette, la convolution et le *lifting scheme*. La convolution correspond à la vision traditionnelle du filtrage. Le *lifting scheme* [Dau98] permet une implémentation plus rapide et surtout les transformées d'entier en entier. On se place ici dans le contexte des ondelettes discrètes biorthogonales.

A.1.1 Approche convolutive

À l'origine, les ondelettes ont été définies sous forme de filtres [Mal89]. L'étape d'analyse correspond à la décomposition en ondelettes du signal et la synthèse à la reconstitution du signal à partir des coefficients d'ondelettes. On note h_a le filtre passe-bas d'analyse, g_a le filtre passe-haut d'analyse et h_s et g_s les filtres de synthèse, comme illustré sur la figure A.1.

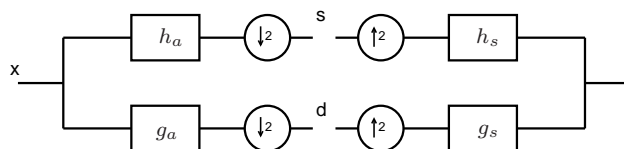


Fig. A.1 Décomposition en ondelettes : filtrage.



L'ondelette 9/7 (9 et 7 étant la longueur des filtres passe-bas et passe-haut) est aussi appelée (4,4) (les valeurs correspondant aux nombres de moments nuls des filtres passe-haut et passe-bas). Cette ondelette est quasi-orthogonale et donc particulièrement appréciée dans le cas de la compression avec pertes, c'est d'ailleurs l'ondelette par défaut pour la compression avec pertes de JPEG 2000. Les valeurs numériques des coefficients des filtres sont présentées dans le tableau A.1.

	0	± 1	± 2	± 3	± 4
h_a	0.85269868	0.37740286	-0.11062440	-0.023849465	0.037828456
g_a	0.78848562	-0.41809227	-0.04068942	0.064538883	
h_s	0.78848562	0.41809227	-0.04068942	-0.064538883	
g_s	0.85269868	-0.37740286	-0.11062440	0.023849465	0.037828456

Tab. A.1 Filtres utilisés pour la 9/7.

L'ondelette 5/3 (5 et 3 étant la longueur des filtres passe-bas et passe-haut) est aussi appelée (2,2) (les valeurs correspondant ici aux nombres de moments nuls des filtres passe-haut et passe-bas). Cette ondelette semble donner de meilleurs performances pour la compression sans pertes, c'est l'ondelette par défaut du standard JPEG 2000 pour la compression réversible. Les valeurs numériques des coefficients des filtres sont présentées dans le tableau A.2.

	0	± 1	± 2
h_a	6/8	2/8	-1/8
g_a	1	-1/2	
h_s	1	1/2	
g_s	6/8	-2/8	-1/8

Tab. A.2 Filtres utilisés pour la 5/3.

A.1.2 Approche lifting

Il est montré que la transformée en ondelettes peut se factoriser sous la forme d'étapes de prédiction et d'*update* [Dau98, Swe97]. La 9/7 peut ainsi se mettre sous cette forme.

En prenant

$$\begin{aligned}
 \alpha &= -1.586134342 \\
 \beta &= -0.05298011854 \\
 \gamma &= 0.8829110762 \\
 \delta &= 0.4435068522 \\
 \zeta &= 1.149604398
 \end{aligned}$$

La DWT 9/7 est effectuée par :

$$\begin{aligned}
 s_0(i) &= x(2i) \\
 d_0(i) &= x(2i + 1) \\
 d_1(i) &= d_0(i) + \alpha(s_0(i) + s_0(i + 1)) \\
 s_1(i) &= s_0(i) + \beta(d_1(i) + d_1(i - 1)) \\
 d_2(i) &= d_1(i) + \gamma(s_1(i) + s_1(i + 1)) \\
 s_2(i) &= s_1(i) + \delta(d_2(i) + d_2(i - 1)) \\
 s(i) &= \zeta s_2(i) \\
 d(i) &= d_2(i)/\zeta
 \end{aligned}$$

La DWT 5/3 correspond à la suite d'opérations :

$$\begin{aligned}
 s_0(i) &= x(2i) \\
 d_0(i) &= x(2i + 1) \\
 d_1(i) &= d_0(i) + 1/2(s_0(i) + s_0(i + 1)) \\
 s_1(i) &= s_0(i) + 1/4(d_1(i) + d_1(i - 1)) \\
 s(i) &= s_2(i)\sqrt{2} \\
 d(i) &= d_2(i)/\sqrt{2}
 \end{aligned}$$

Le lifting scheme permet une implémentation moins coûteuse de la transformée en ondelette. Cette approche est particulièrement intéressante au niveau de l'utilisation de la mémoire puisqu'elle permet à la transformée en ondelette d'être réalisée en utilisant uniquement une taille mémoire légèrement supérieure à la taille des données.

A.2 La transformée entière

Un autre point intéressant est la facilité avec laquelle on peut implémenter des transformées d'entier en entier pour permettre la compression sans pertes [Cal98]. On obtient ainsi une transformée en entier correspondant à la DWT 9/7 par :

$$\begin{aligned}
s_0(i) &= x(2i) \\
d_0(i) &= x(2i + 1) \\
d_1(i) &= d_0(i) + \lfloor \alpha(s_0(i) + s_0(i + 1)) + 1/2 \rfloor \\
s_1(i) &= s_0(i) + \lfloor \beta(d_1(i) + d_1(i - 1)) + 1/2 \rfloor \\
d(i) &= d_1(i) + \lfloor \gamma(s_1(i) + s_1(i + 1)) + 1/2 \rfloor \\
s(i) &= s_1(i) + \lfloor \delta(d_2(i) + d_2(i - 1)) + 1/2 \rfloor
\end{aligned}$$

et pour la DWT 5/3 :

$$\begin{aligned}
s_0(i) &= x(2i) \\
d_0(i) &= x(2i + 1) \\
d(i) &= d_0(i) + \lfloor 1/2(s_0(i) + s_0(i + 1)) + 1/2 \rfloor \\
s(i) &= s_0(i) + \lfloor 1/4(d_1(i) + d_1(i - 1)) + 1/2 \rfloor
\end{aligned}$$

Si on veut conserver une transformée unitaire, il faut ajouter un coefficient d'échelle. Des recherches ont été faites dans [Xio03] pour rendre ce facteur d'échelle compatible avec une transformée sur des entiers.

Dans notre cas, on utilise la transformée d'entier en entier uniquement dans le cas de la compression sans perte, le facteur d'échelle n'a pas d'importance.

A.3 Notation des sous-bandes

Il existe plusieurs notations pour les sous-bandes de la transformée en ondelettes. La figure A.2 présente la notation classique pour la décomposition multirésolution. On reprend cette notation en 3D dans notre cas, principalement pour référence à la sous-bande de plus basse fréquence appelée LLL.

Un exemple de la notation plus générale présentée dans le chapitre 3 appliqué à la décomposition multirésolution est donné sur la figure A.3.

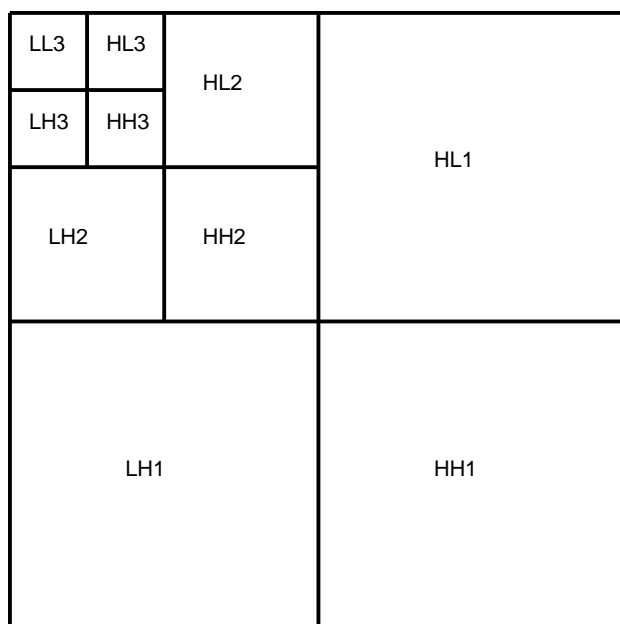


Fig. A.2 Décomposition multirésolution et appellation des sous-bandes.

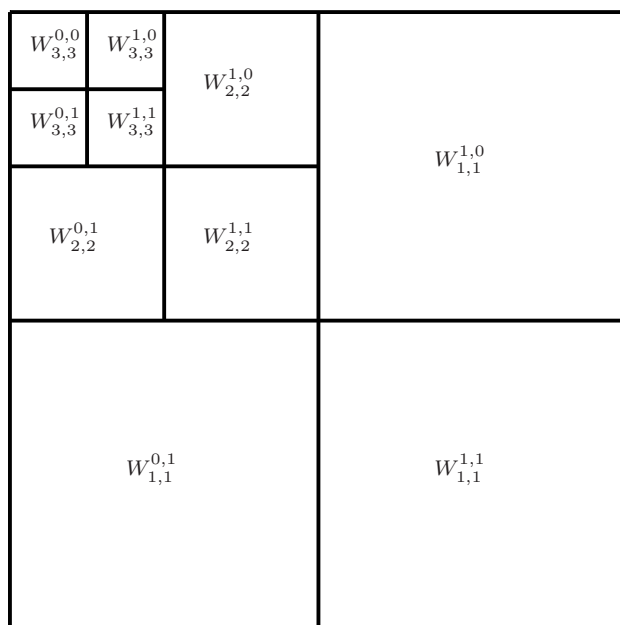


Fig. A.3 Décomposition multirésolution et appellation des sous-bandes (généralisée).

Annexe B

Exemples pour EZW et SPIHT

LES algorithmes EZW et SPIHT sont populaires et utilisés dans beaucoup de travaux, néanmoins, pour en comprendre les subtilités il est utile d'appliquer *à la main* ces algorithmes sur des données simples. Pour permettre de mieux comprendre ces deux algorithmes (décrits dans le chapitre 3 ainsi que dans [Sha93] et [Sai96]) et mettre en évidence leurs différences, on applique ici la première passe de ces deux algorithmes sur un exemple concret.

L'exemple original de l'article [Sha93] est appliqué également à SPIHT [Sai99]. Ces deux exemples sont repris ici sur les mêmes données (Fig. B.1) pour faciliter la compréhension des algorithmes. Rappelons que le codage est fait par plans de bits successifs.

63	-34	49	10	7	13	-12	7
-31	23	14	-13	3	4	6	-1
15	14	3	-12	5	-7	3	9
-9	-7	-14	8	4	-2	3	2
-5	9	-1	47	4	6	-2	2
3	0	-3	2	3	-2	0	4
2	-3	6	-4	3	6	3	6
5	11	5	6	0	3	-4	4

Fig. B.1 Exemple de coefficients issus de la transformée en ondelettes.

B.1 Déroulement sur EZW

Pour EZW, rappelons également qu'un coefficient est déclaré significatif (symbole POS ou NEG suivant son signe) si sa valeur absolue est supérieure au seuil T_k du plan de bits. Sinon, soit il s'agit d'un zéro isolé (IZ) si au



Sous-bande	Valeurs du coefficient	Symbole	
LL3	63 (0,0)	POS	(1)
HL3	-34 (1,0)	NEG	
LH3	-31 (0,1)	IZ	(2)
HH3	23 (1,1)	ZTR	(3)
HL2	49 (2,0)	POS	
HL2	10 (3,0)	ZTR	
HL2	14 (2,1)	ZTR	
HL2	-13 (3,1)	ZTR	
LH2	15 (0,2)	ZTR	
LH2	14 (1,2)	IZ	
LH2	-9 (0,3)	ZTR	
LH2	-7 (1,3)	ZTR	
HL1	7 (4,0)	Z	(4)
HL1	13 (5,0)	Z	
HL1	3 (4,1)	Z	
HL1	4 (5,1)	Z	
HL1	-1 (2,4)	Z	
HL1	47 (3,4)	POS	(5)
HL1	-3 (2,5)	Z	
HL1	-2 (3,5)	Z	

Tab. B.1 Déroulement de la première passe de l'algorithme EZW (seuil 32) sur les données de la figure B.1.

moins un de ses descendants est significatif, soit d'un arbre de zéros (ZTR) si aucun de ses descendants n'est significatif.

Le déroulement de l'algorithme EZW sur les données de la figure B.1 est décrit dans le tableau B.1. Quelques précisions sont données pour certaines étapes, les numéros correspondants sont données dans la dernière colonne du tableau B.1 :

1. Le premier coefficient a la valeur 63, il est supérieur au seuil courant (32), le symbole POS est émis.
2. -31 est non significatif, mais il possède un descendant qui est significatif (47) c'est un zéro isolé, IZ est émis.
3. 23 est non significatif et tous ses descendants sont non significatifs, il s'agit donc d'un arbre de zéros, ZTR est émis.
4. 7 est non significatif et comme la sous-bande HL1 n'a pas de descendant, les symboles IZ et ZTR peuvent être regroupés en un seul symbole Z.
5. 47 est significatif, le symbole POS est émis. Comme ce symbole sera traité pendant la passe de raffinement, il pourra être considéré comme étant 0 pour les passes suivantes (le symbole 14 de la LH2 sera alors une racine d'un arbre de zéros).

En supposant que EZW utilise 2 bits pour coder les symboles POS, NEG, ZTR et IZ et 1 bit pour coder le symbole Z, on obtient un débit de $26+7=33$ bits.

B.2 Déroulement sur SPIHT

L'algorithme SPIHT maintient trois listes de coefficients : la liste des coefficients significatifs (List of Significant Pixels ou LSP), la liste des coefficients non significatifs (List of Insignificant Pixels ou LIP) et la liste des ensembles non significatifs (List of Insignificant Sets ou LIS). $\mathcal{O}(x, y, \lambda)$ est l'ensemble des enfants de (x, y, λ) (un seul niveau de descendance), $\mathcal{D}(x, y, \lambda)$ est l'ensemble de tous les descendants et $\mathcal{L}(x, y, \lambda) = \mathcal{D}(x, y, \lambda) - \mathcal{O}(x, y, \lambda)$ est l'ensemble des descendants à l'exception des enfants. La fonction $S_k(x, y, \lambda)$ est égale à 0 si tous les descendants de (x, y, λ) sont en dessous du seuil T_k (arbre de zéros) et 1 dans le cas contraire. SPIHT considère deux types d'arbres de zéros : le type A où tous les descendants ne sont pas significatifs (arbre de degré 1) et le type B où tous les descendants, à l'exception d'au moins un des enfants, ne sont pas significatifs (arbre de degré 2).

Le déroulement de l'algorithme SPIHT sur les données de la figure B.1 est décrit dans le tableau B.2. Quelques précisions sont données pour certaines étapes :

1. Initialisation des listes : la LSP est vide, la LIP contient les 4 coefficients de plus basse fréquence et la LIS contient la même chose que la LIP à part le coefficient (0,0) (valeur 63) qui n'a pas de descendants.
2. (0,0) qui a la valeur 63 est significatif, il est mis dans la LSP. Le signe de cette valeur est également émis, et noté +.
3. Un des descendants de (1,0) est significatif (49) on teste donc les 4 enfants de (1,0).
4. Comme un des enfants de (1,0) était significatif, il est changé de type (B) et mis à la fin de la LIS.
5. Tous les descendants de (1,1) sont en dessous du seuil, on ne fait donc rien de particulier, on passe à l'élément suivant dans la LIS.
6. (1,0) est de type B, on regarde donc $\mathcal{L}(1,0)$ qui n'est pas significatif
7. (0,1) est de type B, on regarde donc $\mathcal{L}(0,1)$ qui est significatif (présence de la valeur 47), on partage donc l'ensemble et on ajoute (0,2), (1,2), (0,3), (1,3) à la fin de la LIS comme élément de type A et on enlève (0,1) de la LIS.
8. Comme (1,2) n'a pas de petits-enfants, au lieu de le passer en élément de type B comme on aurait du le faire, on le retire de la LIS.
9. Ces listes constituent les listes d'entrées pour le passage suivant où le seuil sera de 16.

La taille de la sortie de cette première passe est de 29 bits.

Coeff. ou ens. testé	Sortie	Action	Liste	
			LIS = $\{(1,0)A, (0,1)A, (1,1)A\}$ LIP = $\{(0,0), (1,0), (0,1), (1,1)\}$ LSP = \emptyset	(1)
(0,0)	1+	(0,0) dans LSP	LIP = $\{(1,0), (0,1), (1,1)\}$ LSP = $\{(0,0)\}$	(2)
(1,0)	1-	(1,0) dans LSP	LIP = $\{(0,1), (1,1)\}$ LSP = $\{(0,0), (1,0)\}$	
(0,1)	0	rien		
(1,1)	0	rien		
$\mathcal{D}(1,0)$	1	tester les descendants		(3)
(2,0)	1+	(2,0) dans LSP	LSP = $\{(0,0), (1,0), (2,0)\}$	
(3,0)	0	(3,0) dans LIP	LIP = $\{(0,1), (1,1), (3,0)\}$	
(2,1)	0	(2,1) dans LIP	LIP = $\{(0,1), (1,1), (3,0), (2,1)\}$	
(3,1)	0	(3,1) dans LIP	LIP = $\{(0,1), (1,1), (3,0), (2,1), (3,1)\}$	
		chgt type	LIS = $\{(0,1)A, (1,1)A, (1,0)B\}$	(4)
$\mathcal{D}(0,1)$	1	tester les descendants		
(0,2)	0	(0,2) dans LIP	LIP = $\{(0,1), (1,1), (3,0), (2,1), (3,1), (0,2)\}$	
(1,2)	0	(1,2) dans LIP	LIP = $\{(0,1), (1,1), (3,0), (2,1), (3,1), (0,2), (1,2)\}$	
(0,3)	0	(0,3) dans LIP	LIP = $\{(0,1), (1,1), (3,0), (2,1), (3,1), (0,2), (1,2), (0,3)\}$	
(1,3)	0	(1,3) dans LIP	LIP = $\{(0,1), (1,1), (3,0), (2,1), (3,1), (0,2), (1,2), (0,3), (1,3)\}$	
		chgt type	LIS = $\{(1,1)A, (1,0)B, (0,1)B\}$	
$\mathcal{D}(1,1)$	0	rien		(5)
$\mathcal{L}(1,0)$	0	rien		(6)
$\mathcal{L}(0,1)$	1	ajouter des ensembles	LIS = $\{(1,1)A, (1,0)B, (0,2)A, (1,2)A, (0,3)A, (1,3)A\}$	(7)
$\mathcal{D}(0,2)$	0	rien		
$\mathcal{D}(1,2)$	1	tester les descendants		
(2,4)	0	(2,4) dans LIP	LIP = $\{(0,1), (1,1), (3,0), (2,1), (3,1), (0,2), (1,2), (0,3), (1,3), (2,4)\}$	
(3,4)	1+	(3,4) dans LSP	LSP = $\{(0,0), (1,0), (2,0), (3,4)\}$	
(2,5)	0	(2,5) dans LIP	LIP = $\{(0,1), (1,1), (3,0), (2,1), (3,1), (0,2), (1,2), (0,3), (1,3), (2,4), (2,5)\}$	
(3,5)	0	(3,5) dans LIP	LIP = $\{(0,1), (1,1), (3,0), (2,1), (3,1), (0,2), (1,2), (0,3), (1,3), (2,4), (2,5), (3,5)\}$	
$\mathcal{L}(1,2)=\emptyset$		enlever (1,2)	LIS = $\{(1,1)A, (1,0)B, (0,2)A, (0,3)A, (1,3)A\}$	(8)
$\mathcal{D}(0,3)$	0	rien		
$\mathcal{D}(1,3)$	0	rien		
			LIS = $\{(1,1)A, (1,0)B, (0,2)A, (0,3)A, (1,3)A\}$ LIP = $\{(0,1), (1,1), (3,0), (2,1), (3,1), (0,2), (1,2), (0,3), (1,3), (2,4), (2,5), (3,5)\}$ LSP = $\{(0,0), (1,0), (2,0), (3,4)\}$	(9)

Tab. B.2 Déroulement de la première passe de l'algorithme SPIHT (seuil 32) sur les données de la figure B.1.

Annexe C

Utilisation de JPEG2000

UTILISER JPEG 2000 n'est pas évident au premier abord. Cette annexe décrit donc les lignes de commandes utilisées pour les comparaisons des chapitres 3 et 4. Le Verification Model qui constitue une implémentation de référence pour la norme contient de nombreuses options dont une grande partie est détaillée dans [JBI01]. Kakadu possède des options semblables, mais utilise souvent une syntaxe différente.

C.1 Correction préliminaire

La version initialement obtenue pour le vérification model ne fonctionnait pas dans le cas multicomposantes, renvoyant une erreur systématique lors de la décompression :

```
Uniform quantization
ERROR
  Attempting to access data from a non-existent marker element! The
  requested element has the description string, "Type of quantization
  (DZQ,MASK,TCQ,...)".
```

Il semble que le décompresseur cherche un marqueur QCD pour chaque composante, alors que d'après la norme, il ne doit pas y avoir plus d'un marqueur QCD par tuile [ISO02]. Dans le fichier *info/std_reverse_info.c* en remplaçant à la ligne 1131 dans la fonction *retrieve_nz_info_from_stream* :

```
type = stream->get_marker_val(stream,tnum,MARKER_QCD_WHICH,instance,0);

par

if (instance == 0){
  type = stream->get_marker_val(stream,tnum,MARKER_QCD_WHICH,instance,0);
}
```

On peut alors bien décoder le fichier multicomposantes obtenu.



C.2 Décorrélation interbandes par DWT

La quantité d'options disponibles pour les programmes ne rend pas forcément son usage très facile. Durant les simulations, les lignes de commande et les options utilisées étaient :

```
vm9_compress_32 -i input.img -o out.bits -rcb 256 256 224 -do bip -dt short
                -rate 224 -Flra -Mtdt -low_rate_tol 0.01 -hi_rate_tol 0.01
```

Le format d'entrée utilisé est une image brute sans indication d'en-tête, c'est pour cela que le format de l'image doit être précisé. L'option `-rcb` indique la taille de l'image : 256 pixels par 256 pixels sur 224 composantes, `-do` indique l'ordre des données, ici en *bit interleave by pixel*, `-dt` indique que les données sont codées sur 16 bits signés (short).

Les autres paramètres servent à donner les options de compression : `-rate` précise le débit voulu (à noter que c'est le débit par pixel spatial et non pas le débit par valeur numérique, une valeur de 224 correspond donc à un bpp de 1), `-Mtdt` demande au codeur de faire d'abord une décorrélation des bandes par une transformation en ondelettes, `-Flra` demande une adaptation du débit par lagrangien (sans cette option, le débit visé pour les images hyperspectrales n'est pas souvent atteint) et enfin `-low_rate_tol` `-hi_rate_tol` précisent la tolérance accordé pour le débit demandé.

Et pour la décompression :

```
vm9_expand_32 -i out.bits -o tmp.raw -do bip
```

Si les images sont codées en *big endian* et traitées sur un PC ou en *little endian* et traitées sur un système Unix, l'option `-swap` est à rajouter lors de la compression et la décompression.

C.3 Décorrélation interbandes par KLT

Pour effectuer une décorrélation par KLT, il faut créer des fichiers annexes :

- klt.fcb
[0-223] -16
- klt.fcc
DECORREL
[0-223] [0-223] [1,1]
DEPEND
[0-223] [0-223] [0,0]
- klt.fct
0
0
1
1 224
2
FILE
matmean

```

1
224 224
2
FILE
matklt
• klt.icb
  [0-223] -16
• klt.icc
  DECORREL
  [0-223] [0-223] [1,1]
  DEPEND
  [0-223] [0-223] [0,0]
• klt.ict
0
0
1
1 224
2
FILE
matmean
1
224 224
2
FILE
matiklt

```

Trois autres fichiers sont nécessaires et référencés par ces premiers :

- `matmean` : contient la valeur moyenne de chaque bande spectrale. Il y a donc 224 valeurs pour des données AVIRIS.
- `matklt` : contient la matrice de transformation directe. Il y a donc 224×224 valeurs pour des données AVIRIS.
- `matiklt` : contient la matrice de transformation inverse. Il y a donc 224×224 valeurs pour des données AVIRIS.

Les données sont enregistrées en binaire au format *float* (32 bits). Ces matrices de passage doivent être calculées par ailleurs.

La ligne de commande à utiliser pour la compression est alors :

```

vm9_compress_32 -i input.img -o out.bits -rcb 256 256 224 -do bip -dt short
  -rate 224 -Flra -Mlin klt -Fguard_bits 5 -low_rate_tol 0.01 -hi_rate_tol 0.01

```

La commande pour la décompression ne change pas.

C.4 Utilisation de Kakadu

Une autre implémentation populaire, également écrite en grande partie par Taubman est Kakadu [Tau06]. Kakadu ne correspond plus à une version de développement, il est donc particulièrement optimisé au niveau de l'utilisation de la mémoire et de la vitesse d'exécution. Les dernières versions commencent à inclure les spécifications de la partie 2 du standard et donc permettent l'utilisation d'une transformée intercomposantes.

L'image doit être enregistrée au format BSQ (cf. annexe D). La ligne de commande utilisée pour effectuer la compression d'une image hyperspectrale à un débit de 1.0 bpppb est :

```
kdu_compress -i moffett512.rawl*2240524288 -o out.jpj -jpx_layers \* -jpx_space sLUM
  Creversible=no Sdims=\{512,512\} Mcomponents=224 Msigned=yes Mprecision=16
  Mstage_inputs:I25=\{0,223\} Mstage_outputs:I25=\{0,223\} Mstage_collections:I25=\{224,224\}
  Mstage_xforms:I25=\{DWT,0,4,5,0\} Mvector_size:I4=224 Mvector_coeffs:I4=0 Mnum_stages=1
  Mstages=25 Sprecision=16 Ssigned=yes Qstep=0.00001 -rate 224.0
```

Pour attendre le débit visé, il est obligatoire de donner explicitement la valeur du Qstep, la valeur par défaut ne correspondant pas.

Le décodage ne peut pas encore se faire en une seule ligne de commande, on est donc obligé dans ce cas d'écrire un script pour décodifier les bandes une à une. Le temps d'exécution est alors beaucoup plus long. Ce dernier point risque probablement d'évoluer dans les versions ultérieures.

```
#!/bin/bash
rm out.rawl
touch out.rawl
for i in `seq 0 223`; do
    echo component: $i
    kdu_expand -i out.jpj -o tmp.rawl -jpx_layer $i
    cat tmp.rawl >> out.rawl
done
rm tmp.rawl
```

Annexe D

Données hyperspectrales

D.1 Images utilisées

SUIVANT les publications, différentes images hyperspectrales sont utilisées. Cependant, quelques jeux de données semblent assez populaires. Nous avons essayé de nous y conformer pour faciliter les comparaisons. Cette annexe détaille les références et la localisation des images utilisées. Ces images proviennent du JPL/NASA pour les images AVIRIS et de la NASA pour Hyperion. Elles sont calibrées et certaines sont corrigées géométriquement pour compenser les mouvements du capteurs. Selon les simulations, ce ne sont pas toujours les mêmes images qui ont été utilisées.

D.2 Format des données

Les données hyperspectrales sont généralement fournies sous la forme de données brutes. Chaque valeur est codée sur 16 bits et les valeurs sont enregistrées les unes à la suite des autres. Il existe différents ordres de progression dans ces valeurs. La progression peut être en *Bits Interleaved by Pixel* (BIP), en *Band SeQuential* (BSQ) ou en *Bits Interleaved by Line* (BIL). Les différents modes de progression sont illustrés sur la figure D.8. Les images AVIRIS sont généralement fournies en BIP, tandis que les images Hyperion sont en BIL. Ces deux ordres correspondent mieux au mode d'acquisition du capteur qui va progresser dans les lignes en fonction du déplacement du capteur. L'ordonnement en BSQ est plus adapté à une visualisation des plans spectraux.

Il faut également prêter attention à l'ordre des octets dans les données fournies. Les machines Unix sont généralement en *big endian*, l'octet de poids fort en premier, tandis que les architectures de type x86 (PC) sont en *little endian*. Les données sont généralement fournies en *big endian* (format



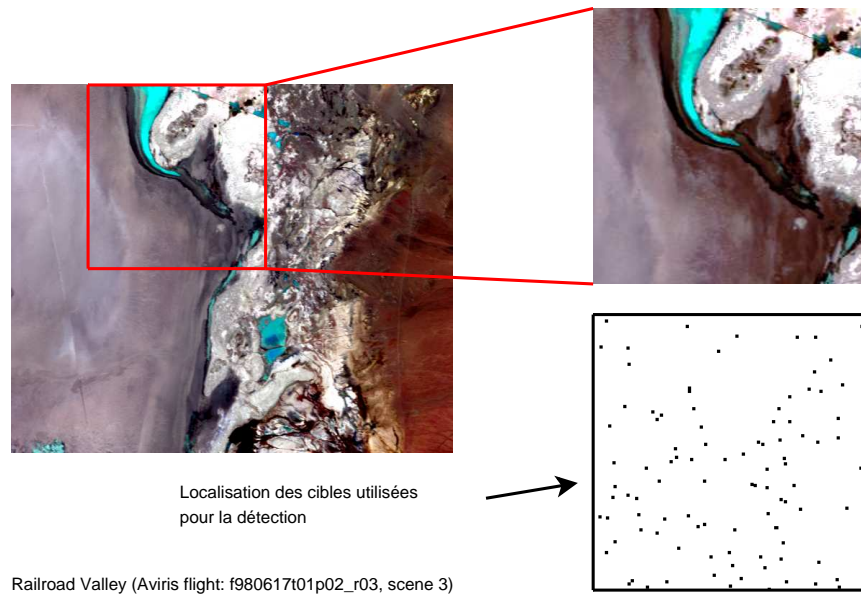


Fig. D.1

IEEE), il faut donc être prudent si on veut lire directement l'image avec un programme C.

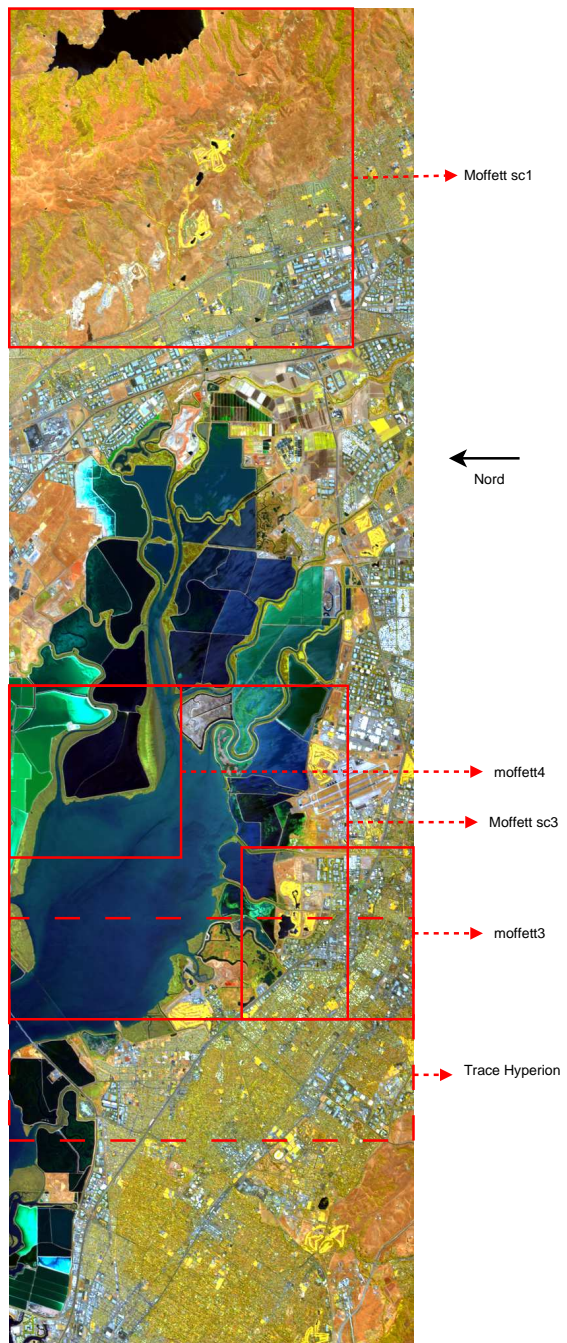
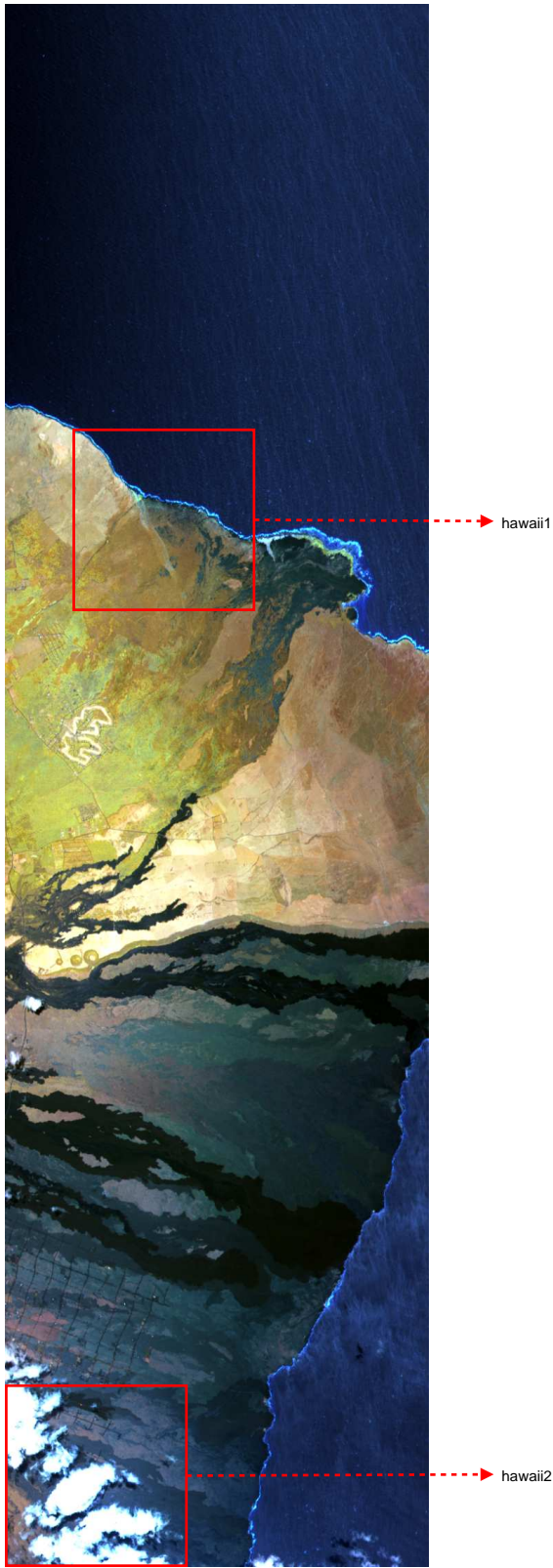


Fig. D.2

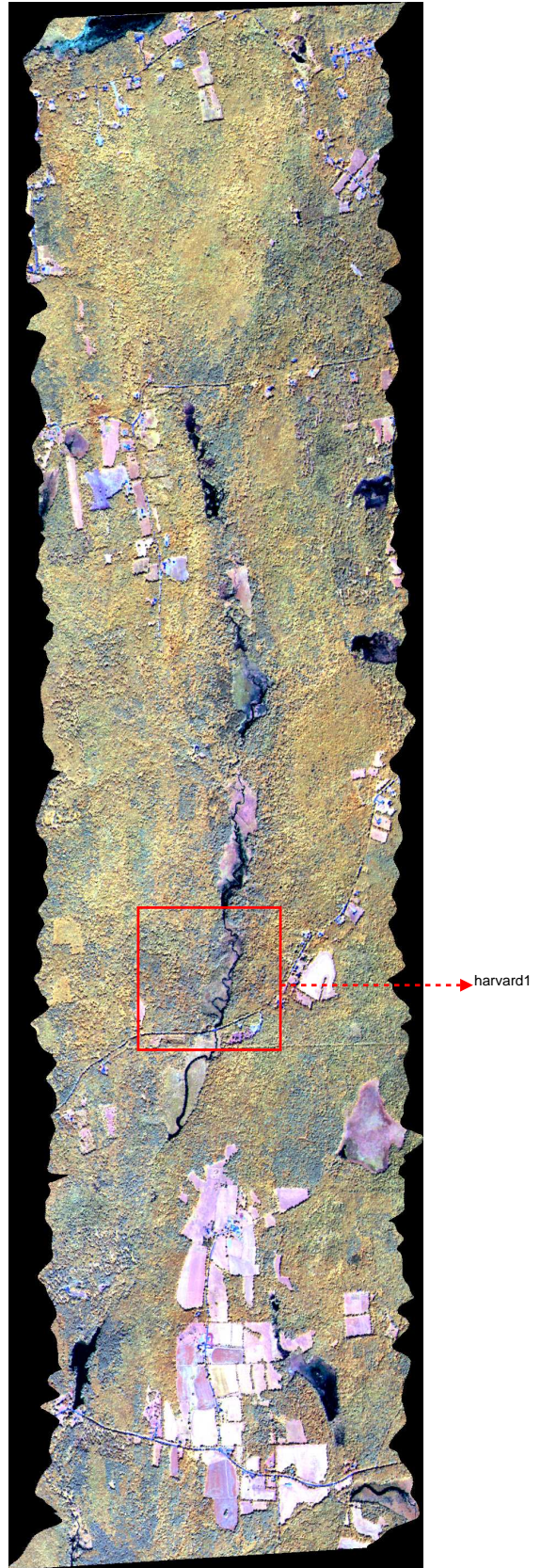


Fig. D.3



Hawaii (AVIRIS flight: f000414t01p03_r08)

Fig. D.4



Harvard Forest (AVIRIS flight: f010903t01p01_r03)

Fig. D.5

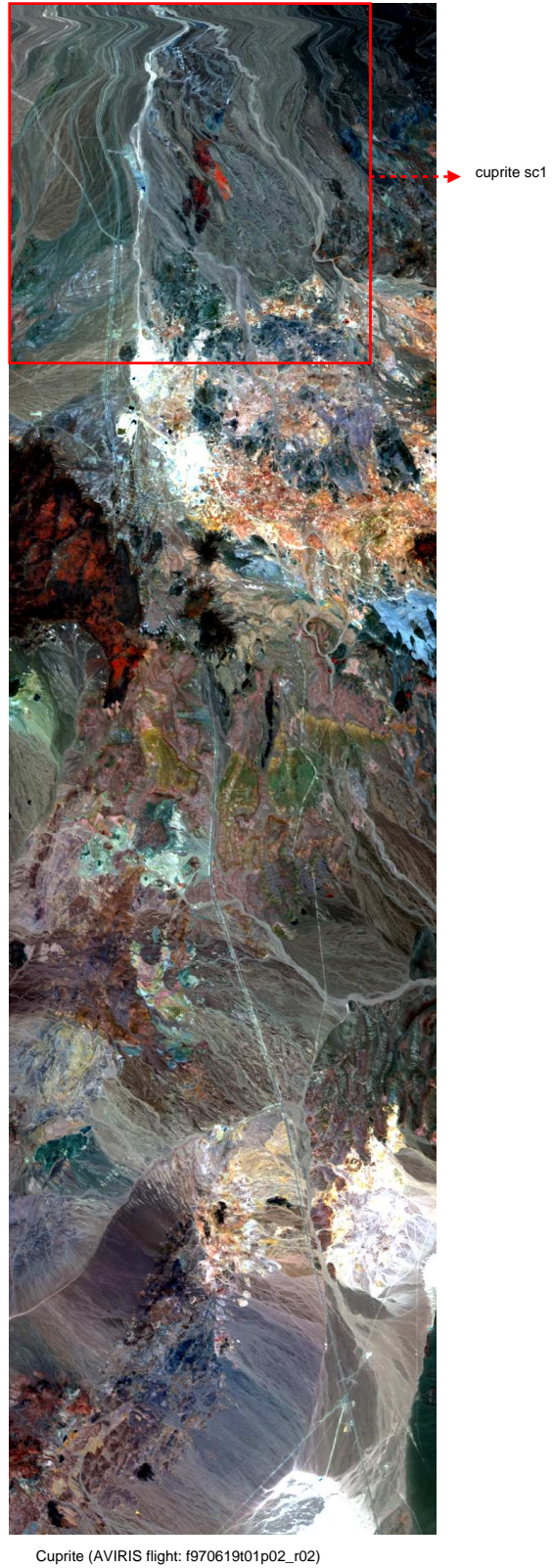
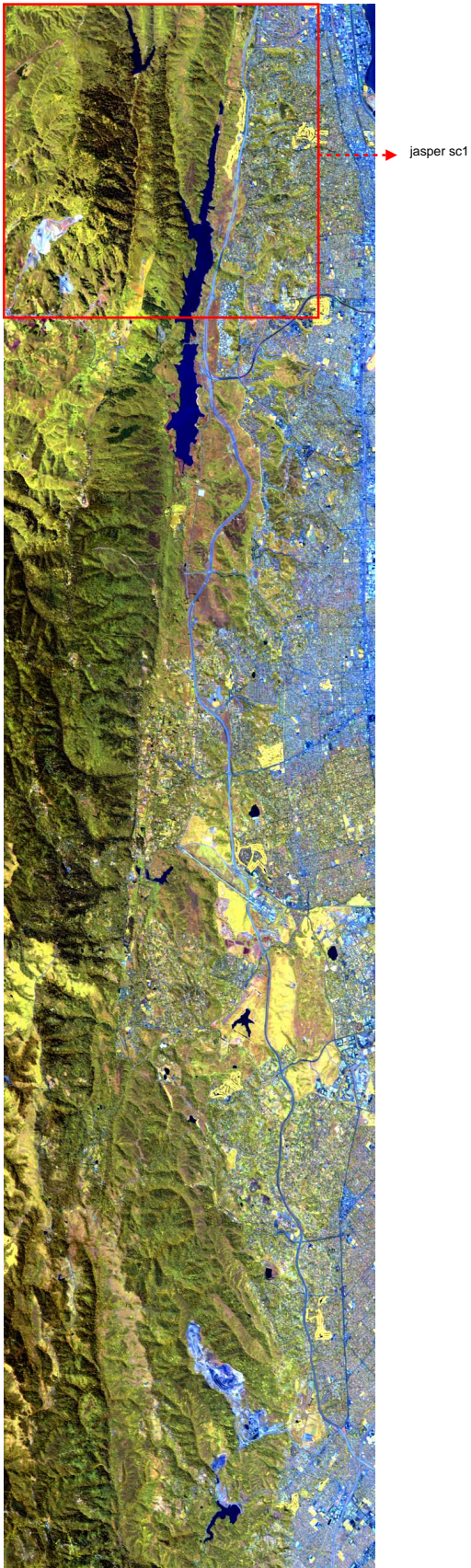


Fig. D.6

Fig. D.7

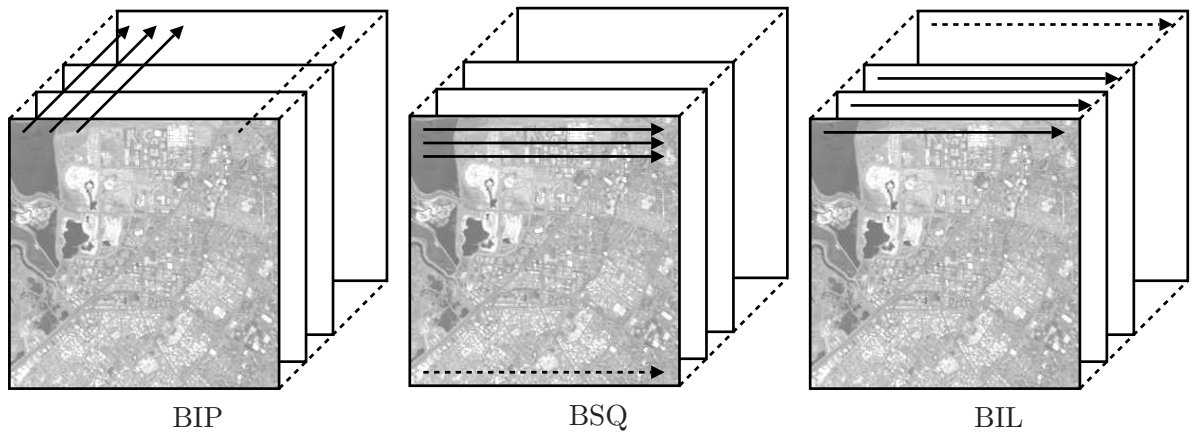


Fig. D.8 Différents ordres courants d'organisation des pixels pour les images hyperspectrales. Les images AVIRIS sont généralement fournies en BIP, les images Hyperion en BIL et le format le plus rapide pour visualiser les images est le BSQ.

Annexe E

Instruments

MALGRÉ la complexité de la réalisation et le concept de l'hyperspectral qui est assez récent, de nombreuses sociétés proposent des instruments hyperspectraux. Cette annexe présente les principales caractéristiques de quelques instruments hyperspectraux en fonctionnement actuellement. Ces instruments sont soit spatiaux, soit aéroportés. Dans tous les cas, ce sont des instruments possédant une bonne résolution spectrale. Ce sont tous des instruments fonctionnant dans le domaine réflectif. Les caractéristiques dans le domaine du visible seront plus particulièrement étudiées. Cette étude n'est pas exhaustive, les instruments hyperspectraux étant très nombreux principalement dans le domaine aéroporté.

Ces caractéristiques serviront à définir les caractéristiques d'un instrument hyperspectral spatial "type" sur lequel seront effectuées les simulations de la thèse. L'idée est de faire des simulations aussi réalistes que possible.

Un inventaire des images disponibles sur lesquelles il serait possible de travailler est également réalisé. Les caractéristiques de ces images sont précisées.

E.1 Démarche

Les instruments mentionnés ici sont bien connus, et largement utilisés par la communauté scientifique. Il existe parfois plusieurs versions de ces instruments (surtout dans le cas des instruments commerciaux), selon les besoins des utilisateurs, mais aussi selon les évolutions technologiques. Les principaux constructeurs d'imageurs hyperspectraux dans un but commercial sont GER (DAIS et EPS), Integrated Spectronics (Hymap), ITRES Research Limited (CASI), Kestrel Corporation (IrCam, HyperCam), SAIC (COIS), Sensytech (MIVIS), TRW (Hyperion). Des constructeurs non commerciaux existent aussi, par exemple JPL pour la NASA (AVIRIS). Les instruments présentés ici proviennent de constructeurs divers, afin de donner une bonne



idée des différentes spécifications et des différentes technologies utilisées.

Les spécifications assez complètes pour les différentes versions de CASI étaient également disponibles. Les deux principales versions sont présentées ici, ce qui permet de voir l'évolution, et les points principaux que le constructeur a amélioré. Ces améliorations sont dictées par les besoins des utilisateurs, c'est donc un bon indicateur pour la tendance des futurs instruments hyperspectraux.

Les premiers instruments présentés sont des instruments aéroportés, ils sont à un stade de fonctionnement commercial. Le passage au spatial n'est pas évident à cause des contraintes spécifiques du domaine. Les instruments présentés à la fin sont des instruments spatiaux souvent envoyés comme démonstrateurs. Ils peuvent servir à valider des technologies, mais aussi les applications possibles à partir de ces instruments. Il est également intéressant de comparer les spécifications des domaines aéroporté et spatial. Le dernier capteur présenté est le capteur Spectra de l'ESA qui devrait être envoyé en 2008.

E.2 Les capteurs actuels

E.2.1 AVIRIS

Les spécifications datent de 1987 pour un instrument aéroporté. Aviris a été réalisé pour simuler les performances des futurs instruments hyperspectraux spatiaux. D'autre part, les images sont facilement disponibles et beaucoup d'endroits sont couverts. De plus, la documentation technique disponible est très complète, et toutes les spécifications sont disponibles sur le site.

Référence : [Por87]

Nom	Airborne Visible / Infrared Imaging Spectrometer
Plateforme	ER-2 (aéroporté haute altitude) autre porteur (aéroporté basse altitude)
Pays	USA
Constructeur	NASA/JPL
Opérateur	NASA/Ames Research Center
Date d'exploitation	1987
Site internet	http://aviris.jpl.nasa.gov

Instrument

FOV	30°
IFOV	1 mrad
Focale effective	19.76 cm
Diamètre effectif de la pupille	14.5 cm
Fibre Optique	Silice (A,B) Verre fluoré (C,D)
Diamètre	200 μm
Ouverture numérique	0.45
Type de scanner	Whiskbroom
Altitude	20 km
Vitesse/sol	735 km/h
Enregistrement	Metrum VLDS
Masse	340 kg
Puissance	1148 W (28 VDC, 41 A)
Dimensions	84 cm \times 160 cm \times 117 cm
Température de fonctionnement	0 à 30°C

Capteur

Spectromètre	A	B	C	D
Spectre (nm)	380-690	670-1270	1260-1880	1880-2500
Nombre de bandes	32	64	64	64
Largeur de bande	9.7 nm	9.5 nm	10.0 nm	12 nm
Réseau (lignes/mm)	117.65	128.2	124.2	128.6
Type	Line array	Line array	Line array	Line array
Nombre d'éléments	32	64	64	64
Matériaux	Silicon	InSb	InSb	InSb
Temps d'intégration	87 μs	87 μs	87 μs	87 μs
Surface active (μm)	200 \times 200	200 \times 200	200 \times 200	200 \times 200
Espace inter-éléments	30 μm	30 μm	30 μm	30 μm

Acquisition des données

Largeur de champ	11 km
Délai de revisite	aéroporté
Nombre de bandes	224
Résolution spectrale	~10 nm
Pixels par ligne	614
Taux d'échantillonnage	12 lignes/s
Échantillonnage spatial	20 m × 20 m (à 20 km) 4 m × 4 m (à 4 km)
Quantification	12 bits
Débit	16 bits après corrections radiométriques 20.4 Mbit/s
SNR A	150 :1
SNR B	140 :1
SNR C	70 :1
SNR D	30 :1
Remarque	Le SNR a été amélioré dans les versions successives de l'instrument, il est de l'ordre de 600 à 1000 pour la version 3

Données disponibles

Scène	
Largeur	1.9 km à 11 km
Longueur	10-100 km
Taille fichier	max 10 Go par vol (850 km)

E.2.2 CASI, CASI-2

CASI est un instrument hyperspectral commercial assez répandu et largement utilisé. C'est l'appareil commercial leader au niveau de l'acquisition d'images hyperspectrales dans le domaine du visible.

Nom	Compact Airborne Spectrographic Imager
Plateforme	Avions légers
Pays	Canada
Constructeur	ITRES Research
Opérateur	ITRES
Date d'exploitation	1989 (CASI-2 2000)
Site internet	http://www.itres.com

Instrument

FOV	37.8°
IFOV	1.3 mrad
Ouverture	f/2.8 à f/11
Élément dispersif	Réseau réflectif
Type de scanner	Pushbroom
Altitude	de 400 m à 8000 m
Vitesse/sol	~ 400 km/h
Enregistrement	Exabyte digital recording (8 mm tapes) 1.1 Go Digital helical scan drive (40 Gbit) ou Disque (9 Gbit) (CASI-2)
Masse	55 kg (dont tête 6 kg)
Puissance	400 W (3.9 A à 110 VAC) ou 560 W (20 A à 28 V)

Capteur

Spectre (nm)	430-940
Nombre de bandes	jusqu'à 288
Largeur de bande	1.9 nm
Résolution spectrale	2.2 nm
Type	EEV UT104 array
Nombre d'éléments	612 × 576
Matériaux	Si (CCD)
Temps d'intégration	30 ms (spatial mode) 100 ms (spectral mode)

Modes d'acquisition

Différents modes d'acquisition sont proposés, principalement pour des raisons de taille de données.

Mode	1	2	3	4
Bandes	288	144	96	72
Pixels par ligne	101	203	405	511

Acquisition des données

Résolution spatiale	de 0.5 à 10 m
Quantification	12 bits
Débit	840 Ko/s à 1 Mo/s
SNR	420 :1 peak
Précision radiométrique	470-800 nm ±2% absolu 430-870 nm ±5% absolu

La taille des données est très variable selon les demandes (résolution, fauchée,...).

E.2.3 CASI-3

CASI-3 est l'évolution, lancée en 2002, des précédents CASI. Il peut être intéressant de comparer les différences entre cette version et la version précédente pour dégager les principales tendances. Ces évolutions répondant à des besoins formulés de la part des utilisateurs.

Le principal avantage de CASI-3 par rapport aux versions précédentes est la plus grande fauchée possible (environ 3 fois celle de CASI-2), ce qui permet de réduire les temps de vol nécessaires pour couvrir la même zone d'environ 60%. Cette amélioration de la fauchée a été obtenue en améliorant la résolution du capteur, ce qui permet à l'avion de voler à plus haute altitude, et donc d'augmenter la fauchée.

CASI-3 a été spécialement conçu pour éviter les distortions au niveau spectral : pas de contamination par les points adjacents (pas de *keystoning*), et les points d'une image à une longueur d'onde donnée, sont uniquement à cette longueur d'onde (pas de *smile*).

Références : [ITR02a], [ITR02b]

Nom	Compact Airborne Spectrographic Imager
Plateforme	Avions légers
Pays	Canada
Constructeur	ITRES Research
Opérateur	ITRES
Date d'exploitation	2002
Site internet	http://www.itres.com

Instrument

FOV	39.5°
IFOV	1.3 mrad
Ouverture	f/3.5 à f/19
Élément dispersif	Réseau réflectif
Type de scanner	Pushbroom
Altitude	jusqu'à 3048 m (non pressurisé)
Vitesse/sol	~ 400 km/h
Enregistrement	
Masse	70 kg
Dimensions	31.8 × 41.7 × 81.3 cm (tête) 18.4 × 48.3 × 52.3 cm (unité de contrôle)
Température de fonctionnement	15 à 30°C (optimal)
Humidité relative	20% à 80%

Capteur

Spectre (nm)	400-1050
Nombre de bandes	jusqu'à 288
Largeur de bande	2.2 nm en moyenne
Type	CCD array
Nombre d'éléments	1480 × 288

Acquisition des données

Résolution spatiale	de 0.5 à 10 m
Taux d'échantillonnage	333 lignes/s
Quantification	14 bits
Débit	2 Mo/s
SNR	480 :1 peak
Précision radiométrique	470-800 nm $\pm 2\%$ absolu 430-870 nm $\pm 5\%$ absolu

E.2.4 MIVIS

Instrument principal de la société Sensytech, MIVIS ne possède que 20 bandes dans le domaine du visible. Néanmoins, ses quatre spectromètres couvrant le domaine spectral du visible à l'infrarouge thermique, peuvent fournir des données utiles pour beaucoup d'applications. Cependant, le bruit est très irrégulier en fonction des différents bandes.

Références : [Bia95]

Nom	Multispectral Infrared and Visible Spectrometer
Plateforme	CASA-212 (aéroporté)
Pays	USA
Constructeur	Daedalus Enterprise (Sensytech maintenant)
Opérateur	CNR-LARA (Italie) et d'autres
Date d'exploitation	1993
Site internet	http://ltpwww.gsfc.nasa.gov/ISSSR-95/mivisair.htm http://www.sensytech.com/Imaging/MIVIS.html

Instrument

FOV	71.059°
IFOV	2.0 mrad
Élément dispersif	Prisme
Type de scanner	Whiskbroom
Altitude	max 4600 m
Vitesse/sol	max 450 km/h
Enregistrement	VHS cassette (10.2 Go)
Masse	220 kg
Puissance	2240 W (28 \pm 3 VDC, 80 A)
Dimensions	67 \times 52 \times 71 cm (tête) 103 \times 48 \times 64 cm (électronique)
Température de fonctionnement	5 à 40°C
Humidité relative	20% à 80%

Capteur

Spectromètre	1 VIS	2 NIR	3 SWIR	4 TIR
Spéctre (nm)	433-833	1115-1550	1983-2478	8180-12700
Nombre de bandes	20	8	64	10
Largeur de bande	20 nm	50 nm	9 nm	340-540 nm

Acquisition des données

Largeur de champ	3.8 km
Délai de revisite	aéroporté
Nombre de bandes	102
Résolution spectrale	20 nm
Pixels par ligne	755
Taux d'échantillonnage	25, 16.7, 12.5, 8.3, 6.25 scan/s
Échantillonnage spatial	5 m × 5 m (à 3000 m)
Quantification	12 bits
Débit	jusqu'à 23 Mbit/s
SNR	
1	55-438
2	715-1125
3	27-198
4	7-80

Données disponibles

Scène	Selon spécifications
Level 0	Données brutes et information pour la calibration radiométrique et les corrections géométriques.
Level 1	Données calibrées radiométriquement
Level 2	Corrections atmosphériques
Level 3	Corrections géométriques

E.2.5 Hymap

Considéré comme un bon modèle pour les futurs capteurs hyperspectraux, cet instrument fournit un spectre complet de 450 à 2480 nm sauf dans les bandes d'absorption de l'eau autour de 1400 et 1900 nm. Il y a environ 5 exemplaires en utilisation actuellement. Certaines différences existent entre les versions (bandes TIR sur certaines versions). L'instrument peut être adapté en fonction des besoins des utilisateurs. HyVista, un opérateur, possède un de ces instruments, et se charge de vendre les images.

NB : La première version d'Hymap en 96 bandes n'est pas décrite ici.

Références : [Coc98], [Kru00]

Nom	Hyperspectral Mapping
Plateforme	Cessna (et autres)
Pays	Australie
Constructeur	Integrated Spectronics
Opérateur	Integrated Spectronics / DLR / HyVista
Date d'exploitation	1996
Site internet	http://www.intspec.com

Instrument

FOV	60°
Ifov	2.5 mrad along track 2.0 mrad across track
Diamètre effectif de la pupille	100 mm
Type de scanner	Whiskbroom
Altitude	1500-5000 m
Vitesse/sol	200-330 km/h
Enregistrement	Exabyte Mammoth (80 Go)

Capteur

Spectromètre	1 VIS	2 NIR	3 SWIR1	4 SWIR2
Spéctre (nm)	450-890	890-1350	1400-1800	1950-2480
Nombre de bandes	32	32	32	32
Largeur de bande (nm)	15-16	15-16	15-16	18-20
Échantillonnage (nm)	15	15	13	17
Matériaux	Si	InSb	InSb	InSb

Acquisition des données

Largeur de champ	2.3 km à Ifov 5 m 4.6 km à Ifov 10 m
Nombre de bandes	128
Résolution spectrale	~ 16 nm
Pixels par ligne	512
Résolution spatiale (Ifov)	5 × 5 m à 2 km 10 × 10 m à 4 km
Quantification	12-16 bits
Débit	2.5 Mo/s
SNR	> 500 :1
Registration interbande	< 1/10 pixel

Données disponibles

Scène	
Largeur	512 pixels
Longueur	variable
Taille fichier	260 Mo (2.5 × 10 km)

E.2.6 Hyperion

Premier capteur à transmettre des données hyperspectrales depuis l'espace, Hyperion a été envoyé pour démontrer la faisabilité des applications spatiales hyperspectrales. Le SNR n'est pas très bon comparé à des instruments aéroportés arrivés à maturité (AVIRIS par exemple).

Références : [Bei02, Bar01, Pea00, Pea01a, Kru02]

Nom	Hyperion
Plateforme	EO-1 (satellite)
Pays	USA
Constructeur	TRW
Opérateur	NASA GSFC (Goddard Space Flight Center)
Date d'exploitation	21/11/2000
Site internet	http://eo1.gsfc.nasa.gov/Technology/Hyperion.html

Instrument

FOV	0.624°
IFOV	42.55 μ rad
VNIR FTM à 630 nm	0.23 - 0.27
SWIR FTM à 1650 nm	0.28
Élément dispersif	Réseau convexe
Type de scanner	Pushbroom
Altitude	705 km
Vitesse/sol	30000 km/h
Masse	49 kg
Puissance	51 W
Dimensions	39 cm \times 75 cm \times 66 cm

Capteur

Spectromètre	VNIR	SWIR
Spectre (nm)	400-1000	900-2500
Nombre de bandes	60	160
Largeur de bande	10 nm	10 nm
Type	Matrice	Matrice
Nombre d'éléments	70 \times 256	172 \times 256
Matériaux	silicon CCD	HgCdTe (120K)
Surface active	60 μ m	60 μ m

Acquisition des données

Largeur de champ	7.5 km
Délai de revisite	8 jours
Nombre de bandes	196 (calibrées)
Résolution spectrale	10 nm
Pixels par ligne	256
Taux d'échantillonnage	223.4 Hz
Résolution spatiale	30 m × 30 m
Quantification	12 bits
Débit	105 Mb/s
Compression	non
Stockage à bord	40 Gbit
VNIR SNR (550-700 nm)	140-190
SWIR SNR (~ 1225 nm)	96
SWIR SNR (~ 2125 nm)	38
Précision radiométrique	3.4%
Co-Reg spatiale	
VNIR	18% (Pix #126)
SWIR	21% (Pix #131)

Données disponibles

Scène	256 × 6072 × 242
VNIR	bandes 1-70 (toutes ne sont pas calibrées)
SWIR	bandes 71-242 (toutes ne sont pas calibrées)
Largeur	7.8 km (7.6 km utilisable)
Longueur	185 km
Taille fichier	565 Mo
	Les données de calibration sont également fournies

E.2.7 CHRIS-PROBA

Capteur satellitaire embarqué à bord de Proba, lancé en octobre 2001 et opérationnel depuis avril 2002, il possède différents modes d'acquisition de données, favorisant soit la résolution spatiale, soit la résolution spectrale. Ce compromis est nécessaire à cause de la capacité limitée de transmission.

L'objectif de cet instrument est de collecter des BRDF (*Bidirectional Reflectance Distribution Function*) afin de mieux comprendre les réflectances spectrales. Au niveau technologique, CHRIS sert aussi à explorer les capacités des spectroimageurs sur des petits satellites maniables.

Références : [ESA99]

Nom	Compact High Resolution Imaging Spectrometer
Plateforme	PROBA
Pays	UK
Constructeur	Sira Electro-Optics Ltd
Opérateur	ESA
Date d'exploitation	2002
Site internet	http://www.chris-proba.org.uk/

Instrument

FOV	1.3°
Focale	746 mm
Diamètre ouverture	120 mm (f/6)
Élément dispersif	Prisme
Type de scanner	Pushbroom
Altitude	~ 615 km
Vitesse/sol	~ 30000 km/h
Masse	15 kg
Puissance	< 9 W
Dimensions	790 × 260 × 200 mm

Capteur

Spectromètre	VNIR
Spectre (nm)	400-1050
Nombre de bandes	max 63
Largeur de bande	entre 1.3 et 12 nm
Type	CCD
Nombre d'éléments	770 × 576

Modes d'acquisition

Mode	1	2	3	4	5
Spatial	36 m	18 m	18 m	18 m	18 m
Spectral	63 bandes	18 bandes	18 bandes	18 bandes	37 bandes
Fauchée	Complète	Complète	Complète	Complète	Demi
Utilisation	Toutes	Eau	Terre/aérosols	Chlorophylle	Terre

La disposition exacte des bandes pour chacun des modes est donnée sur le site de CHRIS-PROBA.

Acquisition des données

Largeur de champ	14 km
Délai de revisite	7 jours
Nombre de bandes	19 (mode 3) 63 (mode 1)
Résolution spectrale	1.3 nm à 410 nm 12 nm à 1050nm
Pixels par ligne	Dépendant du mode de fonctionnement
Résolution spatiale	18 m
Quantification	12 bits
Débit	5 images/12 h (1 Mb/s)
Compression	a priori non
Stockage à bord	~ 700 Mbit
SNR	200
Précision radiométrique	
Pixel registration	>5% (spectral et spatial)

Données disponibles

Scène	
Largeur	14 km (748 pix)
Longueur	14 km (748 pix)
Taille fichier	131 Mo

E.2.8 COIS-NEMO

NEMO a des applications civiles et militaires. Il a été conçu pour l'observation des zones côtières (eaux peu profondes).

Références : [Kra02]

Nom	Coastal Ocean Imaging Spectrometer
Plateforme	NEMO
Pays	USA
Constructeur	NRL/SAIC
Opérateur	Navy
Date d'exploitation	2003
Site internet	

Instrument

FOV	2.86°
IFOV	
Focale effective	36 cm, f/2.4
Diamètre ouverture	15 cm
Élément dispersif	Réseau
Type de scanner	Pushbroom
Altitude	605 km
Vitesse/sol	~ 30000 km/h
Masse	61 kg

Capteur

Spectromètre	VNIR	SWIR
Spectre (nm)	400-1000	1000-2500
Nombre de bandes	60	150
Largeur de bande	10 nm	10 nm
Type	Si CCD	MCT
Nombre d'éléments	1024 × 1024	1024 × 1024
Surface active (μm)	18	18

Acquisition des données

Largeur de champ	30 km
Délai de revisite	2.5 jours
Nombre de bandes	210
Résolution spectrale	10 nm
Pixels par ligne	
Taux d'échantillonnage	
Résolution spatiale	30/60 m
Quantification	12 bits
Débit	131 Mb/s (bande X) 1.024 Mb/s (bande S)
Compression	>10 :1 (Orasis)
Stockage à bord	48 Gbit 9 images par orbites (30 m) 227 Gbit
VNIR SNR	200 :1
SWIR SNR	100 :1

Données disponibles

Scène	
Largeur	30 km
Longueur	200 km

Le système ORASIS (Optical Real-time Adaptive Signature Identification System), développé par NRL, réduit significativement les besoins en stockage à bord et en transmission. ORASIS réalise une identification de signatures spectrales sans

connaissances a priori ni supervision. La méthode d'identification est basée sur des techniques de projections orthogonales.

E.2.9 Spectra

Spectra est actuellement en développement par l'ESA. Toutes les caractéristiques ne sont pas encore fixées, mais il est intéressant de voir les objectifs visés. Les données mentionnées ici ne concernent que la bande 450-2350 nm, la bande 10.3-12.3 μm n'est pas détaillée.

Références : [Eur01]

Nom	Surface Processes and Ecosystem Changes Through Response Analysis
Plateforme	Spectra
Pays	Europe
Constructeur	ESA
Opérateur	ESA
Date d'exploitation	lancement 2008
Site internet	http://www.esa.int/esaLP/spectra.html

Instrument

Les spécifications ne sont pas encore terminées, et plusieurs hypothèses sont encore à l'étude concernant le système optique.

FOV	4.3°
Élément dispersif	
Type de scanner	Pushbroom
Altitude	± 670 km
Vitesse/sol	~ 30000 km/h
Masse	Selon les choix techniques : 410 kg ou 250 kg
Puissance	320 W
Dimensions	
Température de fonctionnement	

Capteur

Spectromètre	VNIR	SWIR
Spectre (nm)	450-2350	
Nombre de bandes	60 (sélectionnables)	
Largeur de bande	12	
Nombre d'éléments	1250 \times 132	1250 \times 256
Matériaux	Silicon CCD	HgCdTe

Acquisition des données

Largeur de champ	50 km
Délai de revisite	14 jours (3 avec dépointage)
Nombre de bandes	62 (dont 2 TIR)
Résolution spectrale	10 nm
Calibration spectrale	1 nm
Pixels par ligne	1250
Échantillonnage spatial	< 50 m
Résolution spatiale	
Quantification	14 bits
Débit	> 100 Mbit/s (bande X) 8.5 Gbit/orbite
Compression	1.8 (lossless)
Stockage à bord	> 100 Gbit
SNR	
Précision radiométrique	2 à 5%
Déréglage inter-bandes	< 0.2 pixels
Déréglage entre spectromètre	< 4 pixels
Déréglage spectrale	1.5 nm

Données disponibles

Scène	~ 50 × 50 km max 1250 × 1000 pixels
Taille fichier	602 Mbit

E.2.10 Autres capteurs

MERIS

MERIS (Medium Resolution Imaging Spectrometer) est un capteur spatial européen de l'ESA. Ce n'est pas réellement un capteur hyperspectral, car il ne transmet que 15 bandes sélectionnées, on n'a plus la continuité spectrale. De plus, la résolution spatiale est assez faible (1200 à 300 m). Une étude exhaustive ne sera donc pas réalisée pour ce capteur.

Hydice

Capteur développé par Hughes Danbury Optical Systems et opérationnel depuis 1994, il a été spécifié pour démontrer l'utilité des techniques hyperspectrales dans les domaines du renseignement, du militaire, mais aussi du civil. Sa configuration est assez similaire à celle de CASI, mais il dispose d'une plus grande résolution spatiale (1-4 m), mais pour une fauchée plus faible (10° environ). A cause de sa nature militaire, les images ne sont pas disponibles.

E.2.11 Résumé

Capteur	λ (nm)	Δ_x (m)	n_λ	Δ_λ (VIS)	Remarques
Aviris	380-2500	4-20	224	~ 10 nm	whiskbroom
CASI3	400-1050	0.5-10	288 max	~ 2 nm	
MIVIS	433..12700	~ 5	102	20 nm	
DAIS7915	400..12600	3-20	79	15-30 nm	
Hymap	450-2480	5-10	128	15 nm	
					Pas de données dans les bandes d'absorption de l'eau, whiskbroom
Hyperion	400-2500	30	196	10 nm	SNR faible
Chris	400-1050	18/36	63 max	1.3-12 nm	
Cois	400-2500	30/60	210	10 nm	Pas encore lancé
Spectra	450-2500	50	62 (sel.)	10 nm	

Bibliographie

- [Aia01] B. Aiazzi, L. Alparone et S. Baronti. “Near-lossless compression of 3-D optical data”. *IEEE Transactions on Geoscience and Remote Sensing.*, vol. 39, n° 11, p. 2547–2557, nov. 2001.
- [Aia03a] B. Aiazzi, L. Alparone, S. Baronti, C. Lastrì, L. Santurri et M. Selva. “Spectral distortion evaluation in lossy compression of hyperspectral imagery”. Dans *IEEE International Geoscience and Remote Sensing Symposium, IGARSS’03*, vol. 3, p. 1817–1819. juil. 2003.
- [Aia03b] B. Aiazzi, L. Alparone, S. Baronti, F. Lotti, L. Santurri et M. Selva. “Quality issues for hyperspectral data dissemination : radiometric and spectral distortions in lossy compression”. Dans *Sensors, Systems, and Next-Generation Satellites VI*, vol. 4881, p. 701–708. SPIE, 2003.
- [Alc06] “Implementation du décorellateur multispectral - R&T Compression”. Rap. tech. 100137101A, Alcatel Alenia Space, Nov 2006.
- [Alp03] L. Alparone, S. Baronti, A. Garzelli et F. Nencini. “A new method for quality assessment of pan-sharpened very high-resolution multispectral images”. Dans *Tyrrhenian International Workshop on Remote Sensing*. CNIT, sept. 2003.
- [Ant92] M. Antonini, M. Barlaud, P. Mathieu et I. Daubechies. “Image coding using wavelet transform”. *IEEE Transactions on Image Processing*, vol. 1, n° 2, p. 205–220, Apr. 1992.
- [Arn93] S. Arno et F. S. Wheeler. “Signed digit representations of minimal hamming weight”. *IEEE Transactions on Computers*, vol. 42, n° 8, p. 1007–1010, août 1993.
- [Bai01] P. Baizert, M. Pickering et M. Ryan. “Compression of hyperspectral data by spatial/spectral discrete cosine transform”. Dans *IEEE International Geoscience and Remote Sensing Symposium, IGARSS’01*, vol. 4, p. 1859–1861. juil. 2001.
- [Bar01] P. Barry. “Introduction to the Hyperion instrument and data processing”. Dans *Hyperion and ALI data users workshop*, p. 3.1–3.41. TRW Space and Electronics Group, nov. 2001.
- [Bei02] D. Beiso. “Overview of Hyperion on-orbit instrument performance, stability, and artifacts”. Dans *Applied Imagery Pattern Recognition Workshop, 2002. Proceedings. 31st*, p. 95–101. oct. 2002.
- [Bia95] R. Bianchi, R. M. Cavalli, C. M. Marino et S. Pignatti. “MIVIS airborne hyperspectral remote sensing in Europe”. Dans *ISSSR-95 proceedings*. CNR, LARA Project, nov. 1995.



- [Bil00] A. Bilgin, G. Zweig et M. W. Marcellin. “Three-dimensional image compression with integer wavelet transforms”. *Applied optics*, vol. 39, n° 11, p. 1799–1814, avr. 2000.
- [Bol03] J. F. Bolton. “Full spectral imaging : a revisited approach to remote sensing”. Dans *Sensors, Systems, and Next-Generation Satellites VII*, vol. 5234, p. 243–251. SPIE, sept. 2003.
- [Bow00] J. Bowles, D. Gillis et P. Palmadesso. “New improvements in the ORASIS algorithm”. *Aerospace Conference Proceedings*, vol. 3, p. 293–298, 2000.
- [Bri96] S. Briles. “Evaluation of onboard hyperspectral-image compression techniques for a parallel push-broom sensor”. Dans *Algorithms for Ultraspectral and Hyperspectral Imagery II*, vol. 2758, p. 332–341. SPIE, 1996.
- [Bri04] M. H. Brill, J. Lubin, P. Costa, S. Wolf et J. Pearson. “Accuracy and cross-calibration of video quality metrics : new methods from ATIS/T1A1”. *Signal Processing : Image Communication*, vol. 19, p. 101–107, fév. 2004.
- [Bur94] M. Burrows et D. Wheeler. “A block-sorting lossless data compression algorithm”. Rap. tech. 124, System Research Center, Digital Equipment Corporation, 1994.
- [Cal98] A. R. Calderbank, I. Daubechies, W. Sweldens et B.-L. Yeo. “Wavelet transforms that map integers to integers”. *Journal of Applied and Computational Harmonic Analysis*, vol. 5, p. 332–369, 1998.
- [Car03] M. Carnec, P. L. Callet et D. Barba. “Full reference and reduced reference metrics for image quality assessment”. Dans *Proceedings of Signal Processing and its Applications*, vol. 1, p. 477–480. IEEE, juil. 2003.
- [Chi01] S.-S. Chiang et C.-I. Chang. “Unsupervised target detection in hyperspectral image using projection pursuit”. *IEEE Transactions on Geoscience and Remote Sensing*, vol. 39, n° 7, p. 1380–1391, juil. 2001.
- [Cho03] S. Cho et W. A. Pearlman. “Error resilient video coding with improved 3-D SPIHT and error concealment”. Dans *Image and Video Communications and Processing*, vol. 5022, p. 125–136. SPIE, jan. 2003.
- [Cho05a] S. Cho et W. A. Pearlman. “Multilayered protection of embedded video bitstreams over binary symmetric and packet erasure channels”. *Journal of Visual Communication and Image Representation*, vol. 16, n° 3, p. 359–378, juin 2005.
- [Cho05b] Y. Cho. *Resolution scalable and random access decodable image coding with low time complexity*. Thèse de doctorat, Rensselaer Polytechnic Institute, Troy, New York, USA, juil. 2005.
- [Cho05c] Y. Cho, W. A. Pearlman et A. Said. “Low complexity resolution progressive image coding algorithm : Progres (progressive resolution decompression)”. Dans *IEEE International Conference on Image Processing*, vol. 3, p. 49–52. sept. 2005.
- [Chr00] C. Chrysafis, A. Said, A. Drukarev, A. Islam et W. A. Pearlman. “SBHP - a low complexity wavelet coder”. Dans *IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2000*, vol. 4, p. 2035–2038. 2000.

- [Chr05a] E. Christophe, D. Léger et C. Mailhes. “Comparison and evaluation of quality criteria for hyperspectral imagery”. Dans *Image Quality and System Performance II*, vol. 5668, p. 204–213. SPIE, jan. 2005.
- [Chr05b] E. Christophe, D. Léger et C. Mailhes. “Images hyperspectrales et critères qualité”. Dans *Colloque annuel des Doctorants EDIT’05*, p. 132–136. UPS, Toulouse, avril 2005.
- [Chr05c] E. Christophe, D. Léger et C. Mailhes. “Quality criteria benchmark for hyperspectral imagery”. *IEEE Transactions on Geoscience and Remote Sensing*, vol. 43, n° 09, p. 2103–2114, Sept. 2005.
- [Chr06a] E. Christophe, C. Mailhes et P. Duhamel. “Best anisotropic 3-D wavelet decomposition in a rate-distortion sense”. Dans *IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP’06*, vol. 2, p. II–17 – II–20. mai 2006.
- [Chr06b] E. Christophe, C. Mailhes et P. Duhamel. “Décorrélation des images hyperspectrales avec une décomposition 3D en ondelettes”. Dans *Workshop on Transform Based on Independent Component Analysis for Audio, Video and Hyperspectral Images Data Reduction and Coding*. juil. 2006.
- [Chr06c] E. Christophe, C. Mailhes et P. Duhamel. “Hyperspectral image compression : Adapting SPIHT and EZW to anisotropic 3D wavelet coding”. *IEEE Transactions on Image Processing*, 2006. (submitted).
- [Chr06d] E. Christophe et W. A. Pearlman. “Three-dimensional SPIHT coding of hyperspectral images with random access and resolution scalability”. Dans *Fortieth Annual Asilomar Conference on Signals, Systems, and Computers*. oct. 2006. (invited paper).
- [Chr06e] E. Christophe et W. A. Pearlman. “Three-dimensional SPIHT coding of volume images with random access and resolution scalability”. *IEEE Transaction on Image Processing*, 2006. (submitted).
- [Chr07a] E. Christophe, P. Duhamel et C. Mailhes. “Adaptation of zerotrees using signed binary digit representations for 3 dimensional image coding”. *EURASIP Journal on Image and Video Processing*, 2007. (to appear).
- [Chr07b] E. Christophe, P. Duhamel et C. Mailhes. “Signed binary digit representation to simplify 3D-EZW”. Dans *IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP’07*. 2007. (to appear).
- [Cle84] J. G. Cleary et I. H. Witten. “Data compression using adaptive coding and partial string matching”. *IEEE Transactions on Communications*, vol. 32, n° 4, p. 396–402, avr. 1984.
- [CNE05] CNES. *Spacecraft Techniques and Technology*. Space Technology Course. 2005.
- [Coc98] T. Cocks, R. Jenssen, A. Stewart, I. Wilson et T. Shields. “The HYMAP airborne hyperspectral sensor : the system, calibration and performance”. Dans *1st EARSEL workshop on imaging spectroscopy, Zurich*. oct. 1998.
- [Coi90] R. R. Coifman, Y. Meyer, S. Quake et M. V. Wickerhauser. “Signal processing and compression with wavelet packets”. Rap. tech., Numerical Algorithms Research Group, Yale University, New Haven, 1990.
- [Coi92] R. R. Coifman et M. V. Wickerhauser. “Entropy-based algorithms for best basis selection”. *IEEE Transactions on Information Theory*, vol. 38, n° 2, p. 713–718, Mar. 1992.

- [Dan03] H. Danyali et A. Mertins. “Fully spatial and SNR scalable, SPIHT-based image coding for transmission over heterogenous networks”. *Journal of Telecommunications and Information Technology*, n° 2, p. 92–98, 2003.
- [Dau98] I. Daubechies et W. Sweldens. “Factoring wavelet transforms into lifting steps”. *J. Fourier Anal. Appl.*, vol. 4, n° 3, p. 247–269, 1998.
- [ESA99] ESA Scientific Campaign Unit, ESTEC. “Exploitation of CHRIS data from the PROBA mission for science and applications experimenters handbook”, oct. 1999.
- [Esk93] A. M. Eskicioglu et P. S. Fisher. “A survey of quality measures for gray scale image compression”. Dans *9th Computing in Aerospace Conference*, 93-4514. AIAA, oct. 1993.
- [Esk95] A. M. Eskicioglu et P. S. Fisher. “Image quality measures and their performance”. *IEEE Transactions on Communications*, vol. 43, n° 12, p. 2959–2965, déc. 1995.
- [Eur01] European Space Agency. “SPECTRA - Surface Processes and Ecosystem Changes Through Response Analysis”. Dans *The Five Candidate Earth Explorer Core Missions*. ESA, sept. 2001.
- [Fow06] J. E. Fowler. “QccPack - Quantization, Compression, and Coding Library”. <http://qccpack.sourceforge.net/>, 2006.
- [Fra01] J. Fraieu et D. Lebedeff. “Compression multispectrale : Synthèse de l’étude et proposition d’évolutions”. Rap. tech., Alcatel Space Industries, nov. 2001.
- [Ger92] A. Gersho et R. M. Gray. *Vector Quantization and Signal Compression*. Kluwer Academic Publisher, 1992.
- [Guo06] J. Guo, S. Mitra, B. Nutter et T. Karp. “A fast and low complexity image codec based on backward coding of wavelet trees”. Dans *Data Compression Conference*, p. 293–301. mars 2006.
- [Hal99] W. H. Hallidy Jr. et M. Doerr. “Hyperspectral image compression”. Rap. tech., Systems and Processes Engineering Corporation, 1999.
- [He03] C. He, J. Dong et Y. F. Zheng. “Optimal 3-D coefficient tree structure for 3-D wavelet video coding”. *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, n° 10, p. 961–972, 2003.
- [Hsi01] S.-T. Hsiang. “Embedded image coding using zeroblocks of sub-band/wavelet coefficients and context modeling”. Dans *Data Compression Conference*, p. 83–92. mars 2001.
- [Huf52] D. A. Huffman. “A method for the construction of minimum-redundancy codes”. Dans *Proceedings of the Institute of Radio Engineers*, vol. 40, p. 1098–1101. sept. 1952.
- [Hug68] G. F. Hughes. “On the mean accuracy of statistical pattern recognizers”. *IEEE Transactions on Information Theory*, vol. 14, n° 1, 1968.
- [Isl99] A. Islam et W. A. Pearlman. “An embedded and efficient low-complexity hierarchical image coder”. Dans *Visual Communications and Image Processing '99*, vol. 3653, p. 294–305. SPIE, jan. 1999.
- [ISO02] “Information technology – jpeg 2000 image coding system : Core coding system”, 2002.

- [ISO04] “Information technology – jpeg 2000 image coding system : Extensions”, 2004.
- [ITR02a] ITRES Research Limited. “CASI-3 specifications”. <http://www.itres.com/docs/casi3.html>, oct. 2002.
- [ITR02b] ITRES Research Limited. “Comparison of CASI-3 and CASI-2”. <http://www.itres.com/docs/casi3.html>, oct. 2002.
- [JBI01] JBIG et JPEG. “Jpeg2000 verification model 9.1 (technical description)”. Rap. tech. N2165, ISO/IEC JTC1/SC29/WG1, juin 2001.
- [Joy00] M. Joye et S.-M. Yen. “Optimal left-to-right binary signed-digit recording”. *IEEE Transactions on Computers*, vol. 49, n° 7, p. 740–748, juil. 2000.
- [Kat99] A. Kattnig, D. Leger, V. Michau, L. Mugnier, M. Duplaa, P. Henry et C. Valorge. “Pôle optique spatiale, groupe de proposition et d’exécution, sous-groupe : qualité et restauration d’image état des lieux”. Rap. tech., CNES / Onera, nov. 1999.
- [Kes02] N. Keshava et J. F. Mustard. “Spectral unmixing”. *IEEE Signal Processing Magazine*, vol. 19, n° 1, p. 44–57, jan. 2002.
- [Kim00] B.-J. Kim, Z. Xiong et W. A. Pearlman. “Low bit-rate scalable video coding with 3-D set partitioning in hierarchical trees (3-D SPIHT)”. *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 10, n° 8, p. 1374–1387, déc. 2000.
- [Kim03] H. Kim et C. Choe, Jand Lee. “Fast implementation of 3-D SPIHT using tree information matrix”. Dans *IEEE International Geoscience and Remote Sensing Symposium, IGARSS’03*, vol. 6, p. 3586–3588. juil. 2003.
- [Kra02] H. J. Kramer. *Observation of the Earth and its Environment : Survey of missions and sensors*. Springer, 4^e éd., 2002.
- [Kru00] F. Kruse, J. Boardman, A. Lefkoff et J. Young. “The AIG/HyVista 1999 USA HyMap group shoot : overview and analysis examples”. Dans *IEEE International Geoscience and Remote Sensing Symposium, IGARSS’2000.*, vol. 4, p. 1373–1375. juil. 2000.
- [Kru02] F. A. Kruse, J. Boardman et J. Huntington. “Comparison of airborne and satellite hyperspectral data for geologic mapping”. Dans *Algorithms and technologies for multispectral, hyperspectral and ultraspectral imagery VIII*, vol. 4725, p. 128–139. SPIE, 2002.
- [Lam95] B. Lamiscarre et D. Léger. “Nouveaux critères pour l’évaluation globale de la qualité des images ; applications aux images satellitaires restaurées ou comprimées”. Rap. tech., CERT/DERO, sept. 1995.
- [Lan99] D. Landgrebe. *Information Processing for Remote Sensing*, chap. Information Extraction Principles and Methods for Multispectral and Hyperspectral Image Data, Chapter 1. World Scientific Publishing Co., Inc., River Edge, NJ, USA, 1999.
- [Lan00] Y. Langevin et O. Forni. “Image and spectral image compression for four experiments on the ROSETTA and Mars Express missions of ESA”. Dans *Applications of Digital Image Processing XXIII*, vol. 4115, p. 364–373. SPIE, 2000.

- [Lur94] J. Lurie, B. Evans, B. Ringer et M. Yeates. “Image quality measures to assess hyperspectral compression techniques”. Dans *International Society for Optical Engineering*, vol. 2313. SPIE, sept. 1994.
- [Ma01] J. Ma, Q. Tong, L. Zheng et B. Zhang. “An approach to assessing hyperspectral data quality”. Dans *Multispectral and hyperspectral image acquisition and processing*, vol. 4548, p. 244–247. SPIE, 2001.
- [Mal89] S. Mallat. “A theory for multiresolution signal decomposition : The wavelet representation”. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, n° 7, p. 674–693, juil. 1989.
- [Mal97] S. Mallat. *A Wavelet Tour of Signal Processing*. Academic Press, 1997.
- [Mar01] M. W. Marcellin et A. Bilgin. “Quantifying the parent-child coding gain in zero-tree-based coders”. *IEEE Signal Processing Letters*, vol. 8, n° 3, p. 67–69, mars 2001.
- [Mof98] A. Moffat, R. Neal et I. H. Witten. “Arithmetic coding revisited”. *ACM Transactions on Information Systems*, vol. 16, n° 3, p. 256–294, 1998.
- [Mot03] G. Motta, F. Rizzo et J. A. Storer. “Compression of hyperspectral imagery”. Dans *Data Compression Conference, DCC*, vol. 8, p. 333–342. IEEE, mars 2003.
- [Mot06] G. Motta, F. Rizzo et J. A. Storer (réds.). *Hyperspectral Data Compression*. Springer, 2006.
- [Oke04] K. Okeya, K. Schmidt-Samoa, C. Spahn et T. Takagi. *Lecture Notes in Computer Science : Advances in Cryptology - CRYPTO 2004*, chap. Signed Binary Representations Revisited, p. 123–139. Spinger-Verlag, 2004.
- [Oli98] J.-L. Olives. *Optimisation globale de système imageur à l'aide de critère de qualité visuelle*. Thèse de doctorat, École nationale supérieure de l'aéronautique et de l'espace, sept. 1998.
- [Oli03] J. Oliver et M. P. Malumbres. “Fast and efficient spatial scalable image compression using wavelet lower trees”. Dans *Data Compression Conference*, p. 133–142. mars 2003.
- [Par03] C. Parisot. *Allocations basées modèles et transformée en ondelettes au fil de l'eau pour le codage d'images et de vidéos*. Thèse de doctorat, Université de Nice, Sophia Antipolis, jan. 2003.
- [Pea98] W. A. Pearlman et A. Said. “Data compression using set partitioning in hierarchical trees”. United States Patent 5,764,807, 9 juin 1998.
- [Pea00] J. Pearlman, C. Segal, L. Liao, S. Carman, M. Folkman, B. Browne, L. Ong et S. Ungar. “Development and operations of the EO-1 Hyperion imaging spectrometer”. Rap. tech., TRW Space and Electronics Group, août 2000.
- [Pea01a] J. Pearlman, S. Carman, C. Segal, P. Jarecke, P. Clancy et W. Browne. “Overview of the Hyperion imaging spectrometer for the NASA EO-1 mission”. Dans *IEEE International Geoscience and Remote Sensing Symposium, IGARSS'01*, vol. 7, p. 3036–3038. juil. 2001.
- [Pea01b] W. A. Pearlman. “Trends of tree-based, set-partitioning compression techniques in still and moving image systems”. Dans *Proceedings Picture Coding Symposium 2001 (PCS 2001)*, p. 1–8. avr. 2001. (invited, keynote paper).

- [Pea04] W. A. Pearlman, A. Islam, N. Nagaraj et A. Said. “Efficient, low-complexity image coding with a set-partitioning embedded block coder”. *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, n° 11, p. 1219–1235, nov. 2004.
- [Pen06] B. Penna, T. Tillo, E. Magli et G. Olmo. “Progressive 3-D coding of hyperspectral images based on JPEG 2000”. *IEEE Geoscience and Remote Sensing Letters*, vol. 3, n° 1, p. 125–129, jan. 2006.
- [Pic00] M. Pickering et M. Ryan. “Compression of hyperspectral data using vector quantisation and the discrete cosine transform”. Dans *IEEE International Conference on Image Processing, ICIP 2000*, vol. 2, p. 195–198. sept. 2000.
- [Por87] W. M. Porter et H. T. Enmark. “A system overview of the airborne visible / infrared imaging spectrometer (AVIRIS)”. Dans *Imaging Spectroscopy II Conference*. Society of Photo-Optical Instrumentation Engineers, août 1987.
- [Pro00] H. Prodinger. “On binary representations of integers with digit -1, 0, 1”. *Integers Electronic Journal of Combinatorial Number Theory*, vol. 0, 2000.
- [Qia00] S.-E. Qian, A. Hollinger, D. Williams et D. Manak. “Vector quantization using spectral index-based multiple subcodebooks for hyperspectral data compression”. *IEEE Transactions on Geoscience and Remote Sensing*, vol. 38, n° 3, p. 1183–1190, mai 2000.
- [Qia03] S. Qian, M. Bergeron, C. Serele, I. Cunningham et A. Hollinger. “Evaluation and comparison of JPEG2000 and vector quantization based onboard data compression algorithm for hyperspectral imagery”. Dans *IEEE International Geoscience and Remote Sensing Symposium, 2003. IGARSS '03.*, vol. 3, p. 1820–1822. juil. 2003.
- [Qia04] S.-E. Qian. “Hyperspectral data compression using a fast vector quantization algorithm”. *IEEE Transactions on Geoscience and Remote Sensing*, vol. 42, n° 8, p. 1791–1798, août 2004.
- [Ram93] K. Ramchandran et M. Vetterli. “Best wavelet packet bases in a rate-distortion sense”. *IEEE Transactions on Image Processing*, vol. 2, n° 2, p. 160–175, avr. 1993.
- [Ris79] J. Rissanen et G. G. L. Jr. “Arithmetic coding”. *IBM Journal of Research and Development*, vol. 23, n° 2, p. 149–162, mars 1979.
- [Ruc05] J. T. Rucker, J. E. Fowler et N. H. Younan. “JPEG2000 coding strategies for hyperspectral data”. Dans *IEEE International Geoscience and Remote Sensing Symposium, IGARSS'05*, vol. 1, p. 128–131. juil. 2005.
- [Rup01] S. Rupert, M. Sharp, J. Sweet et E. Cincotta. “Noise constrained hyperspectral data compression”. Dans *IEEE International Geoscience and Remote Sensing Symposium, IGARSS'01*, vol. 1, p. 94–96. juil. 2001.
- [Rya97] M. J. Ryan et J. F. Arnold. “Lossy compression of hyperspectral data using vector quantization”. *Remote Sensing of Environment*, vol. 61, p. 419–436, 1997.
- [Rya98] M. Ryan et J. Arnold. “A suitable distortion measure for the lossy compression of hyperspectral data”. Dans *IEEE International Geoscience*

- and Remote Sensing Symposium, IGARSS'98*, vol. 4, p. 2056–2058. juil. 1998.
- [Rya00] M. Ryan et M. Pickering. “An improved M-NVQ algorithm for the compression of hyperspectral data”. Dans *IEEE International Geoscience and Remote Sensing Symposium, IGARSS 2000.*, vol. 2, p. 600–602. juil. 2000.
- [Sai96] A. Said et W. A. Pearlman. “A new, fast, and efficient image codec based on set partitioning in hierarchical trees”. *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, n° 3, p. 243–250, juin 1996.
- [Sai99] A. Said. “Example of application for image compression”. http://www.cipr.rpi.edu/~pearlman/papers/ex_spiht-ewz.pdf, 1999.
- [Sha49] C. E. Shannon et W. Weaver. *The Mathematical Theory of Communication*. Illini Books, 1949.
- [Sha93] J. M. Shapiro. “Embedded image coding using zerotrees of wavelet coefficients”. *IEEE Transactions on Signal Processing*, vol. 41, n° 12, p. 3445–3462, déc. 1993.
- [She98] S. S. Shen et B. S. Beard. “Effects of hyperspectral compression on nonliteral exploitation”. Dans *Imaging Spectrometry IV*, vol. 3438, p. 191–199. SPIE, juil. 1998.
- [Shk02] D. Shkarin. “PPM : one step to practicality”. Dans *Data Compression Conference*. 2002.
- [Sho88] Y. Shoham et A. Gersho. “Efficient bit allocation for an arbitrary set of quantizers”. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 36, n° 9, p. 1445–1453, sept. 1988.
- [Sim97] R. E. Simmons et B. V. Brower. “Data characterization for hyperspectral image compression”. Dans *Multispectral imaging for terrestrial application II*, vol. 3119, p. 172–183. SPIE, 1997.
- [Sko01] A. Skodras, C. Christopoulos et T. Ebrahimi. “The JPEG 2000 still image compression standard”. *IEEE Signal Processing Magazine*, vol. 18, n° 5, p. 36–58, sept. 2001.
- [Sub00] S. Subramanian, N. Gat, A. Ratcliff et M. Eismann. “Real-time hyperspectral data compression using principal components transformation”. Dans *Aviris Proceedings*. JPL Publication, 2000.
- [Swe97] W. Sweldens. “The lifting scheme : a construction of second generation wavelets”. *SIAM Journal on Mathematical Analysis*, vol. 29, p. 511–546, 1997.
- [Swe01] J. N. Sweet, M. Sharp et J. Granahan. “Hyperspectral analysis toolset”. Dans *Sensors, systems, and next-generation satellites IV*, vol. 4169, p. 396–407. SPIE, 2001.
- [Tan03] X. Tang, W. A. Pearlman et J. W. Modestino. “Hyperspectral image compression using three-dimensional wavelet coding”. Dans *Image and Video Communications and Processing*, vol. 5022, p. 1037–1047. SPIE, jan. 2003.
- [Tau00] D. Taubman. “High performance scalable image compression with EBCOT”. *IEEE Transactions on Image Processing*, vol. 9, n° 7, p. 1158–1170, juil. 2000.

- [Tau02] D. S. Taubman et M. W. Marcellin. *JPEG2000 Image Compression Fundamentals, Standards and Practice*. Kluwer Academic Publishers, Boston, MA, 2002.
- [Tau06] D. Taubman. “Kakadu Software v 5.0”. <http://www.kakadusoftware.com/>, 2006.
- [Thi06] C. Thiebaut, D. Lebedeff, C. Latry et Y. Bobichon. “On-board compression algorithm for satellite multispectral images”. Dans *Data Compression Conference*. 2006.
- [VB05] D. Van Buren. “A high-rate JPEG2000 compression system for space”. Dans *IEEE Aerospace Conference*, p. 1–7. IEEE, mars 2005.
- [VQE03] VQEG. “Final report from the video quality experts group on the validation of objective models of video quality assessment, phase II”. Rap. tech., VQEG, août 2003. [Http://www.vqeg.org](http://www.vqeg.org).
- [Wan02a] Z. Wang et A. C. Bovik. “A universal image quality index”. *IEEE Signal Processing Letters*, vol. 9, n° 3, p. 81–84, mars 2002.
- [Wan02b] Z. Wang, A. C. Bovik et L. Lu. “Why is image quality assessment so difficult ?” Dans *IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP’02*, vol. 4, p. 3313–3316. mai 2002.
- [Wan04a] Y. Wang, J. T. Rucker et J. E. Fowler. “Three-dimensional tarp coding for the compression of hyperspectral images”. *IEEE Geoscience and Remote Sensing Letters*, vol. 1, n° 2, p. 136–140, Apr. 2004.
- [Wan04b] Z. Wang, L. Lu et A. C. Bovik. “Video quality assessment based on structural distortion measurement”. *Signal Processing : Image Communication*, vol. 19, p. 121–132, 2004.
- [wg101] “JPEG2000 verification model 9.1 (technical description)”, juin 2001.
- [wg104] “JP3D working draft v5.0”, nov. 2004.
- [Wit87] I. H. Witten, R. M. Neal et J. G. Cleary. “Arithmetic coding for data compression”. *Communications of the ACM*, vol. 30, n° 6, p. 520–540, déc. 1987.
- [Xie05] G. Xie et H. Shen. “Highly scalable, low-complexity image coding using zeroblocks of wavelet coefficients”. *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 15, n° 6, p. 762–770, juin 2005.
- [Xio03] Z. Xiong, X. Wu, S. Cheng et J. Hua. “Lossy-to-lossless compression of medical volumetric data using three-dimensional integer wavelet transforms”. *IEEE Transactions on Medical Imaging*, vol. 22, n° 3, p. 459–470, mars 2003.
- [Xu03] D. Xu et M. N. Do. “Anisotropic 2D wavelet packets and rectangular tiling : theory and algorithms”. Dans *SPIE, Wavelets : Applications in Signal and Image Processing X.*, vol. 5207, p. 619–630. Nov 2003.
- [Yeh05] P.-S. Yeh, P. Armbruster, A. Kiely, B. Masschelein, G. Moury, C. Schaefer et C. Thiebaut. “The new CCSDS image compression recommendation”. Dans *IEEE Aerospace Conference*. IEEE, mars 2005.
- [Ziv77] J. Ziv et A. Lempel. “A universal algorithm for sequential data compression”. *IEEE Transactions on Information Theory*, vol. IT-23, n° 3, p. 337–343, mai 1977.

Résumé :

Les images hyperspectrales présentent des caractéristiques spécifiques qui demandent à être exploitées par un algorithme de compression efficace. Cette thèse se consacre à la définition d'un système complet de compression pour les images hyperspectrales. En compression, les ondelettes ont montré une bonne efficacité sur des données diverses tout en conservant une complexité raisonnable. De plus, le codage des coefficients d'ondelettes est souvent assuré par des algorithmes basés sur le principe des arbres de zéros qui sont, à l'heure actuelle, parmi les plus efficaces. C'est pourquoi ce travail de thèse s'est intéressé à définir dans un premier temps une décomposition en ondelettes quasi-optimale au sens d'un critère débit-distorsion pour les images hyperspectrales. Dans un deuxième temps, nous proposons une adaptation des méthodes de codage par arbres (EZW, SPIHT), associées à la décomposition obtenue. Les performances sont comparées à une adaptation de JPEG 2000 pour les images hyperspectrales, démontrant l'intérêt des méthodes proposées.

D'autre part, les utilisateurs des images hyperspectrales sont souvent intéressés uniquement par certaines caractéristiques de l'image (résolution ou zone) en fonction de l'application. Une adaptation de l'algorithme précédent est réalisée pour permettre l'accès aléatoire afin de décoder uniquement une partie de l'image (en spatial ou en spectral). La progression en résolution est également disponible permettant de décoder des images à différentes résolutions à partir du même train binaire tout en lisant un minimum de bits. Il est également possible de spécifier de manière indépendante les résolutions spatiales et spectrales souhaitées.

Enfin, on ne peut parler de compression (avec pertes) sans définir au préalable un critère de distorsion adapté. Nous définissons ainsi un groupe de cinq critères de qualité présentant une bonne complémentarité. Ces cinq critères ont été choisis afin de pouvoir s'assurer que l'algorithme de compression n'entraîne pas des dégradations compromettant l'intérêt des données. Une nouvelle méthode d'utilisation de ces cinq critères de qualité montre une bonne aptitude à distinguer différentes dégradations produites sur l'image. Cette méthode, appliquée au nouvel algorithme de compression montre que les dégradations restent faibles pour des débits autour de 1 bit par pixel par bande.

Mots clés : compression, hyperspectral, satellite, EZW, SPIHT, critère qualité.

Abstract :

Hyperspectral images present some specific characteristics that should be used by an efficient compression system. This thesis focuses on the definition and the optimization of a full wavelet compression system for hyperspectral images. In compression, wavelets have shown a good adaptability to a wide range of data, while being of reasonable complexity. Zerotree based compression algorithms are among the best for image compression. Therefore, in this work, efficient compression methods based on zerotree coding (EZW, SPIHT) are adapted on a near-optimal wavelet decomposition for hyperspectral images. Performances are compared with the adaptation of JPEG 2000 for hyperspectral images.

End users of hyperspectral images are often interested only in some specific features of the image (resolution, location) which depend on the application. A further adaptation of the proposed hyperspectral image compression algorithm is presented to allow random access to some part of the image, whether spatial or spectral. Resolution scalability is also available, enabling the decoding of different resolution images from the compressed bitstream of the hyperspectral data while reading a minimum amount of bits from the coded data. Final spatial and spectral resolutions are chosen independently.

Finally, any lossless compression method cannot be characterized without the definition of a distortion measure. Therefore, a group of five quality criteria presenting a good complementarity is defined. The purpose is to make sure the compression algorithm does not impact significantly the data quality. A new method using these five criteria shows a good ability to discriminate between different degradations. Application of this method to the newly defined algorithm shows that the degradation remains low for compression rate around 1.0 bit per pixel per band.

Keywords : compression, hyperspectral, satellite, EZW, SPIHT, quality criteria.